Text Categorization with Fractional Gradient Descent Support Vector Machine

Dian Puspita Hapsari[1*], Imam Utoyo [2], and Santi Wulan Purnami [3]

[1]Department of Informatics Engineering, Institute of Technology Adhi Tama Surabaya, Indonesia
[2]Faculty Science & Technology, Airlangga University, Indonesia
[3]Faculty Science & Technology, Institute of Technology Sepuluh Nopember Surabaya, Indonesia

[*]dian.puspita@itats.ac.id

Abstract. Text documents on the web are an incredible resource including one example of big data, large size and so many variations that it becomes difficult for humans to choose meaningful information without the help of a computer. Text categorization job is to automatically classify text documents into standards class based on their content. The objective of this research is to implement a classifier with optimization based on the Fractional Gradient Descent in text classification. In our research, we propose using the Fractional Gradient Descent to optimize the SVM classifier so that it can increase the speed of training data. We explore a batch of different training data to compare the speed of the UCI ML text dataset training process with the SVM-SGD and SVM-FGD classifiers. This research concludes that using SVM-FGD will optimize the training time for text dataset in the activity of data classification.

1. Introduction

Data mining, also referred to as knowledge discovery in data, text mining refers to such a knowledgediscovery process when the source data is text. Data mining and text mining in general should be regarded as application-oriented interdisciplinary fields. It can be found to be closely related to disciplines such as natural language processing, computational linguistics and information retrieval, machine learning and artificial intelligence. Nowadays, the access to a virtually unlimited amount of information in digital text format, text mining is becoming a very important tool for both providing competitive services to users and extracting valuable knowledge [1].

Text mining is new field of computer science, through techniques such as categorization or classification, entity extraction, sentiment analysis, text mining extracting information that is useful for uncovering hidden knowledge in text [2]. Text mining is able to reveal insights, patterns, and trends in large volumes of unstructured data accompanied by the ability to get rid of all irrelevant material and provide fast answers. Based on the purpose of this study to improve the scalability of supervised learning techniques. An efficient learning algorithm is used to conduct a short learning on each training data sample. Machine learning methods that can be trained for millions of training examples so they can enjoy the latest huge databases. In short, we have looked for training algorithms that have a short training time property (linear scaling with training set sizes) but have high generalization accuracy. Support Vector Machines (SVMs) as one of the supervised learning technique algorithms that can be applied in many cases. That is the reason why we try to present most of our algorithms in general to facilitate the conception of derivation for large scale applications.

This research employs Farm Ads text dataset for classification with 4,143 rows, 54.877 attributes from UCI-ML repository. This data was collected from text ads found on twelve websites that deal with various farm animal related topics. Information from the ad creative and the ad landing page is included. The binary labels are based on whether or not the content owner approves of the ad. For each ad, we include the words on the ad creative and the words from the landing page. Each word from the creative is given a prefix of 'ad-'. Title and header HTML mark-up are noted in a similar way in the text of the landing page. We have already performed stemming and stop word removal. Each ad is on a single line. The first word in the line is the label of the instance. It is 1 for accepted ads and -1 for rejected ads. We have also included a straightforward bag-of-words representation of our data. We use the SVM light sparse vector format. Each of these attributes is encoded as index: value.

Moreover, we studied the performance using the Fractional Gradient Descent to optimize the SVM classifier so that it can increase the speed of training time for classification data training. Inspired by (Pu et al., 2015; Chen, 2013) [3], we apply the fractional steepest descent algorithm to train SVMs. In particular, we employ the Caputo derivative formula to compute the fractional-order gradient of the error function with respect to the objective function and obtain the deterministic convergence. The classification of large scale learning has focused on building linear classification models for large scale dataset classification tasks. In many test cases, the linear SVM classifier is an exchange between training time and classification accuracy. Shalev- Shwartz et al. [4] and Bottou et al. [5] proposed a stochastic gradient reduction algorithm for SVM (symbolized by SVM-SGD) which showed promising results for the problem of large- scale binary classification.

2. Support Vector Machine

Support-vector machines are supervised learning models that analysed data used for classification and regression analysis. The original SVM algorithm was invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. In 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create nonlinear classifiers by applying the kernel trick to maximum-margin hyperplanes Boser, Bernhard E [6].

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

For linear classifier SVMs can efficiently perform a non-linear classification using kernel trick, mapping their inputs into high-dimensional feature spaces. SVMs tries to find the hyperplane. We will call this the optimal hyperplane, and we will say that it is the one that best separates the data. The SVM optimization problem is:

$$\min \cdot \quad \frac{1}{2}\| \quad \| \qquad \cdot \quad + \quad \geq 1, \quad = 1 \ldots \qquad (1)$$

This function is convex in w. A function is convex if for every $\mathbf{!}, \mathbf{''}$ in the domain, and for every $\# \in [0,1]$ we have $\% \#\mathbf{!} + 1 - \# \mathbf{''} \leq \#\% \mathbf{!} + 1 - \# \% \mathbf{''}$. For convex functions in general the function f is $\nabla\% \quad = 0$ for x to be a minimum.

The stationary condition tells us that the selected point must be a stationary point. It is a point where the function stops increasing or decreasing. When there is no constraint, the stationary condition is just the point where the gradient of the objective function is zero. When we have constraints, we use the gradient of the Lagrangian [7].

Primal feasibility condition, looking at this condition, you should recognize the constraints of the primal problem. It makes sense that they must be enforced to find the minimum of the function under constraints. Dual feasibility condition. Similarly, this condition represents the constraints that must be respected for the dual problem.

This formulation of the SVM is called the hard margin SVM. It cannot work when the data is not linearly separable. There are several Support Vector Machines formulations. In the next chapter, we will consider another formulation called the soft margin SVM, which will be able to work when data is non-linearly separable because of outliers. Minimizing the norm of is a convex optimization problem, which can be solved using the Lagrange multipliers method. Some researchers have discovered new heuristics to improve this algorithm, and popular libraries like LIBSVM use an SMO-like algorithm. Note that even if this is the standard way of solving the SVM problem, other methods exist, such as gradient descent and stochastic gradient descent (SGD), which is particularly used for online learning and dealing with huge datasets [8,9,10].

2.1 Gradient Descent

Gradient Descent is a popular optimization technique in Machine Learning and Deep Learning and it can be used with most, if not all, of the learning algorithms. A gradient is slope of a function, the degree of change of a parameter with the amount of change in another parameter. Mathematically, it can be described as the partial derivatives of a set of parameters with respect to its inputs. Gradient Descent is a convex function [11,12].

Gradient Descent can be described as an iterative method which is used to find the values of the parameters of a function that minimizes the cost function as much as possible. The parameters are initially defined a particular value and from that, Gradient Descent is run in an iterative fashion to find the optimal values of the parameters, using calculus, to find the minimum possible value of the given cost function [13,14,15].

Gradient descent is an iterative algorithm, that starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function. This algorithm is useful in cases where the optimal points cannot be found by equating the slope of the function to 0. The general idea is to start with a random point (in our parabola example start with a random "x") and find a way to update this point with each iteration such that we descend the slope [16,17]. We are trying to minimize,

$$* = \min \quad + \quad , \; 0,1 - \quad , \qquad (2)$$

The steps of the algorithm are; First, find the slope of the objective function with respect to each parameter/feature. In other words, compute the gradient of the function. Second, pick a random initial value for the parameters. (To clarify, in the parabola example, differentiate "y" with respect to "x". If we had more features like x1, x2 etc., we take the partial derivative of "y" with respect to each of the features.) Third, update the gradient function by plugging in the parameter values. And last calculate the step sizes for each feature as: step size = gradient * learning rate. Calculate the new parameters as: new params = old params -step size. Repeat steps 3 to 5 until gradient is almost 0. The "learning rate" mentioned above is a flexible parameter which heavily influences the convergence of the algorithm. Larger learning rates make the algorithm take huge steps down the slope and it might jump across the minimum point thereby missing it. So, it is always good to stick to low learning rate such as 0.01. It can also be mathematically shown that gradient descent algorithm takes larger steps down the slope if the starting point is high above and takes baby steps as it reaches closer to the destination to be careful not to miss it and also be quick enough [18].

Journal of Physics: Conference Series **1477** (2020) 022038 doi:10.1088/1742-6596/1477/2/022038

There are a few downsides of the gradient descent algorithm. We need to take a closer look at the amount of computation we make for each iteration of the algorithm. When we have 10,000 data points and 10 features. The sum of squared residuals consists of as many terms as there are data points, so 10000 terms in our case. We need to compute the derivative of this function with respect to each of the features, so in effect we will be doing 10000 * 10 = 100,000 computations per iteration. It is common to take 1000 iterations, in effect we have 100,000 * 1000 = 100000000 computations to complete the algorithm. That is pretty much an overhead and hence gradient descent is slow on huge data [19.20].

2.2 Fractional Gradient Descent

The definition of the Caputo fractional-order derivative of order α is:

∠- % = + / Γ 12/ - 3α − 4 12/2+% 1 4 54, (3)
/

Where −- is caputo derivative operator, α is the fractional order.

When 7 ∈ 0,1 , the expression for Caputo derivative is as follows:

∠- % = + / Γ 12/ - 3α − 4 2/% 8 4 54. (4)

When 7 ∈ 0,1 and α =0, there is a visible difference in the derivation with respect to a constant between Caputo derivative and Riemann-Liouville derivative. Suppose that K is a / constant, it is easy to conclude that −- 9 = 0 and :; −- 9 = + 2/ / Γ +2 / ≠ 0. That the reason why we used Caputo derivative for fractional order gradient descent optimization [21,22].

3. Research methodology

Pre-processing is the initial phase of text classification to form unstructured texts into token representation and ready to be modelled by a classifier algorithm. Clean up Text, Cleaning is the process to remove extra characters in HTML syntax used to design website. Tokenization Tokenizing is the process of extracting serialized word in sentences to become variation of two words in many combinations, done at each sentence. For further more find special tokens, lowercases, stop words, and stemming.

Moreover we convert to Numeric for DFM (Doc Frequency Matrix) and feature selection with TF-IDF. Each dataset is randomly split into two subsets with a fixed proportion, training and testing samples are separately and we use Cross Validation with 10 fold for stratified CV. Finally we applied SVM-FGD for the dataset. To analyse the numerical performance based on different fractional-order derivatives, three performance metrics have been conducted: training accuracy, testing accuracy and training time.

4. Result

The result implementation SVM-FGD for training dataset is objective value = 2.2507 with test error = 4,367 and train error = 2,362, norm w = 395.048 for accuracy classifier = 0.9074. theoretical results of the presented algorithm in this paper, such as monotonicity and convergence. The accuracies with larger learning rates are higher than those with smaller learning rates in each sub-figure. In addition, we observe that training and testing accuracies are gradually increasing with increasing fractional-orders and then reach the peak around =-order.>

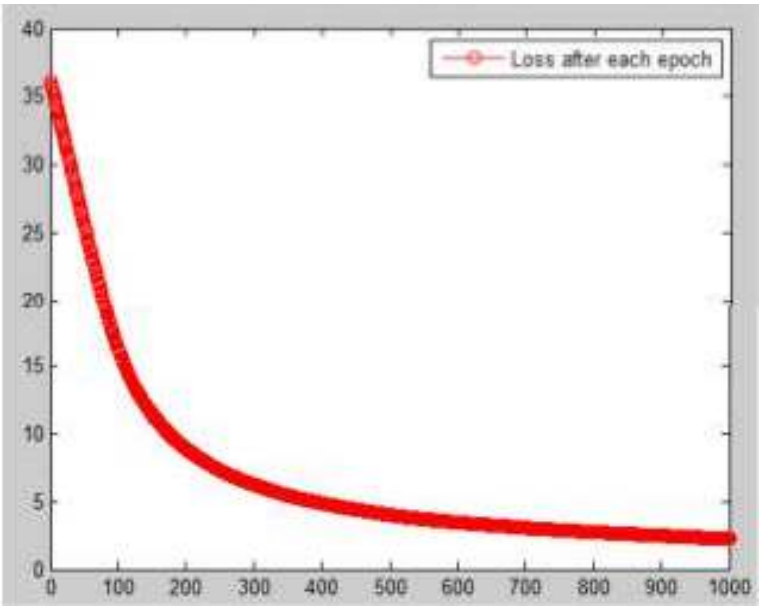

Figure 1. The result implementation SVM-FGD for training dataset is objective value = 2.2507 with test error = 4,367 and train error = 2,362, norm w = 395.048 for accuracy classifier = 0.9074

5. Conclusion

In this research, implementation text classification with support vector machine classifier and unconstraint optimization with stochastic gradient descent already done. This research tries to study implication for FGD optimizer for large scale training data. Algorithm chosen is FGD with two kernel variations to gain best result. Based on the result, it can be concluded that using FGD decreases the training time and precision of SVM. The accuracy of TF-IDF is better combined with FGD when the sample does not have specific terms, instead, it has a unique pattern in the same news portal provider. For further research, we tried to implemented fractional gradient descent to improve performance training time text dataset in large scale.

Acknowledgments

The authors would like to express their gratitude to the editors and anonymous reviewers for their valuable comments and suggestions which improve the quality of this paper. Gratitude for the supervisor of the doctoral program in the faculty of science and technology, Airlangga university.

References

[1] Zhou, L. et al. 2017. 'Machine learning on big data: Opportunities and challenges',Neurocomputing. doi: 10.1016/j.neucom.2017.01.026.
[2] Prasetijo, A.B., Isnanto,Rizal., Eridani, Dania., 2017. Hoax detection system on Indonesian news sites based on text classification using SVM and SGD.
[3] Pu, Y., Zhou, J., Zhang, Y., Zhang, N., Huang, G., Siarry, P., 2015. Fractional extreme value adaptive training method: fractional steepest descent approach. IEEE Transactions on neural networks and learning systems, 26 (4), 653–662.
[4] Auria, L. and Moro, R. A. (2011) 'Support Vector Machines (SVM) as a Technique for Solvency Analysis', SSRN Electronic Journal. doi: 10.2139/ssrn.1424949.

[5] Boser, B. E., Guyon, I. M. and Vapnik, V. N. (2004) 'A training algorithm for optimal margin classifiers', in. doi: 10.1145/130385.130401.
[6] Boser, E. et al. (1992) 'Training Algorithm Margin for Optimal Classifiers', Annual Workshop on Computational Learning Theory. doi: 10.1145/130385.130401.
[7] Boyd, Stephen, and L. V. (2004) Convex Optimization, Communication Networking. doi: 10.1016/B978-012428751-8/50019-7.
[8] C. Li, W. Deng, and D. Xu, "Chaos synchronization of the Chua system with a fractional order," Physica A, vol. 360, no. 2, pp. 171185, Feb. 2006.
[9] Chervonenkis, A. Y. (2013) 'Early history of support vector machines', in Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik. doi: 10.1007/978-3-642-41136-6_3.
[10] D. Xu, H. Zhang, and D. Mandic, "Convergence analysis of an augmented algorithm for fully complex-valued neural networks," Neural Networks, vol. 69, no. C, pp. 44-50, Sept. 2015.
[11] E. Kaslik and S. Sivasundaram, "Nonlinear dynamics and chaos in fractional-order neural networks," Neural Networks, vol. 32, no. 1, pp. 245-56, Aug. 2012.
[12] G. Anastassiou, "Fractional neural network approximation," Comput. Math Appl., vol. 64, no. 6, pp. 1655-1676, Sept. 2012.
[13] Hsieh, C.-J. et al. (2008) 'A dual coordinate descent method for large-scale linear SVM', in. doi: 10.1145/1390156.1390208.
[14] Wu, W., Wang, J., Cheng, M., Li, Z., 2011b. Convergence analysis of online gradient method for bp neural networks. Neural Networks, 24, 91–98.
[15] Wang, H., Yu, Y., Wen, G., Zhang, S., Yu, J., 2015. Global stability analysis of fractional-order hopfield neural networks with time delay. Neurocomputing, 154, 15–23.
[16] Zhang, S., Yu, Y., Wang, H., 2015. Mittag-leffler stability of fractional-order hopfield neural networks. Nonlinear Analysis: Hybrid Systems, 16, 104–121
[17] Chen, B., Chen, J., 2016. Global o(t−α) stability and global asymptotical periodicity for a non-autonomous fractional-order neural networks with time-varying delays. Neural Networks, 73, 47–57.
[18] Chen, X., 2013. Application of fractional calculus in bp neural networks. Ph.D. thesis, Nanjing Forestry University, Nanjing, Jiangsu.
[19] H. Zhang, W. Wu, F. Liu, and M. Yao, "Boundedness and convergence of online gradient method with penalty for feedforward neural networks," Neural Process Lett., vol. 20, no. 6, pp. 1050-1054, Jun. 2009.
[20] H. Zhang, D. Xu, and Y. Zhang, "Boundedness and Convergence of Split-Complex Back-Propagation Algorithm with Momentum and Penalty," Neural Process Lett., vol. 39, no. 3, pp. 297-307, Jun. 2014.
[21] Wang, J., Wen, Y., Gou, Y., Ye, Z., Chen, H.,2017. Fractional-order gradient descent learning of BP neural networks with Caputo derivative
[22] Yang,G., Zhang, B., Sang, z., Wang, J., 2017. A Caputo-Type Fractional-Order Gradient Descent Learning of BP Neural Networks.