Adaptively Solving the Local-Minimum Problem for Deep Neural Networks

Huachuan Wang and James Ting-Ho Lo
Department of Mathematics and Statistics, University of Maryland Baltimore County, USA.

1 Abstract

This paper aims to overcome a fundamental problem in the theory and application of deep neural networks (DNNs). We propose a method to solve the local minimum problem in training DNNs directly. Our method is based on the cross-entropy loss criterion's convexification by transforming the cross-entropy loss into a risk averting error (RAE) criterion. To alleviate numerical difficulties, a normalized RAE (NRAE) is employed. The convexity region of the cross-entropy loss expands as its risk sensitivity index (RSI) increases. Making the best use of the convexity region, our method starts training with an extensive RSI, gradually reduces it, and switches to the RAE as soon as the RAE is numerically feasible. After training converges, the resultant deep learning machine is expected to be inside the attraction basin of a global minimum of the cross-entropy loss. Numerical results are provided to show the effectiveness of the proposed method.

2 Introduction

The problem, called the local minimum problem in training DNNs, has plagued the DNN community since the 1980s[1,2]. DNNs trained with backpropagation are extensively utilized to solve various tasks in artificial intelligence fields for decades[3–7]. The computing power of DNNs is derived through its particularly distributed structure and the capability to learn and generalize. However, the application and further development of DNNs have been impeded by the local minimum problem and have attracted much attention for a very long time. DNNs has recently been extensively studied to represent high-level abstractions. Training DNNs involves significantly intricate difficulties, prompting the development of unsupervised layer-wise pre-training and many ingenious heuristic techniques. Although such methods and techniques have produced impressive results in solving famous machine learning tasks, more serious problems originated from the essence of local minimum in high-dimensional non-convex optimization remain[8–11].

A primary difficulty of solving the local minimum problem lies in the intrinsic nonconvexity of the training criteria of the DNNs[12–16], which usually contain a large number of non-global local minima in the weight space of the DNNs. As the standard optimization methods perform a local search in the parameter (e.g., weight) space, they cannot consistently guarantee the resultant DNN's satisfactory performance even with many training sessions. Although an enormous amount of solutions have been developed to optimize the free parameters of the objective function for consistently achieving a better optimum, these methods or algorithms cannot solve the local minimum problem essentially with the intricate presence of the non-convex function[17–22].

The most standard approach to optimize DNNs is Stochastic Gradient Descent (SGD). There are many variants of SGD, and researchers and practitioners typically choose a particular variant empirically. While nearly all DNNs optimization algorithms in popular use are gradient-based, recent work has shown that more advanced second-order methods such as L-BFGS and Saddle-Free Newton (SFN) approaches can yield better results for DNN tasks[2315]. Second-order derivatives can be addressed by GPUs or batch methods when dealing with massive data, SGD still provides a robust default choice for optimizing DNNs. Instead of modifying the network structure or optimization techniques for DNNs, we focused on designing a new error function to convexify the error space. The convexification approach has been studied in the optimization community for decades but has never been seriously applied within deep learning. A well-known methods is the LiuFloudas convexification method[1,24]. LiuFloudas convexification can be applied to optimization problems where the error criterion is twice continuously differentiable, although determining the weight α of the added quadratic function for convexifying the error criterion involves significant computation when dealing with massive data and parameters. Following the same name employed for deriving robust controllers and filters[24–26].

To alleviate the local minimum problem's fundamental difficulty in training DNNs, this paper proposes a series of methodologies by applying convexification and deconvexification to avoid non-global local minima and achieve the global or near-global minima with satisfactory optimization and generalization performances. These methodologies are developed based on a normalized risk-averting error (NRAE) criterion. The use of this criterion removes the practical difficulty of computational overflow and ill-initialization that existed in the risk-averting error criterion, which was the predecessor of the NRAE criterion Furthermore, it has benefits to effectively handle non-global local minima by convexifying the non-convex error space. The method's effectiveness based on the NRAE criterion is evaluated in training multilayer perceptrons (MLPs) for function approximation tasks, demonstrating the optimization advantage compared to training with the standard mean squared error criterion. Moreover, numerical experiments also illustrate that the NRAE-based training methods applied to train DNNs, such as convolutional neural networks and deep MLPs, to recognize handwritten digits in the MNIST dataset achieve better optimization generalization results than many benchmark performances. Finally, to enhance the generalization of the DNNS obtained with the NRAE-based training, a statistical pruning method that prunes redundant connections of the DNNS is implemented and confirmed for further improving the generalization ability of the DNNs trained by the NRAE criterion.

3 Risk-Averting Error

Given training samples $X, y = (x_1, y_1), (x_2, y_2), ..., (x_m, y_m)$, the function $f(x_i, W)$ is the learning model with parameters W. The cross-entropy loss function $J(\boldsymbol{W})$ is defined as:

$$J\left(f\left(\boldsymbol{x}_i, \boldsymbol{W}\right), y_i\right) = \frac{1}{m}\sum_{i=1}^{m}\left[y_i \log f\left(\boldsymbol{x}_i, \boldsymbol{W}\right) + \left(1 - y_i\right)\log\left(1 - f(\boldsymbol{x}_i, \boldsymbol{W})\right)\right] \tag{1}$$

The Risk-Averting Error criterion (RAE) corresponding to the $L(\boldsymbol{W})$ is defined by

$$RAE_p\left(f\left(\boldsymbol{x}_i,\boldsymbol{W}\right),y_i\right)=\frac{1}{m}\sum_{i=1}^{m}e^{\lambda^p\left[y_i\log f(\boldsymbol{x}_i,\boldsymbol{W})+(1-y_i)\log\left(1-f(\boldsymbol{x}_i,\boldsymbol{W})\right)\right]} \tag{2}$$

$$=\frac{1}{m}\sum_{i=1}^{m}\left[f\left(\boldsymbol{x}_i,\boldsymbol{W}\right)^{y_i}\left(1-f(\boldsymbol{x}_i,\boldsymbol{W})\right)^{(1-y_i)}\right]^{\lambda^p} \tag{3}$$

λ is the convexity index. It controls the size of the convexity region.

Because RAE has the sum-exponential form, its Hessian matrix is tuned exactly by the convexity index λ^p. Given the Risk-Averting Error criterion $RAE_p(p\in\mathcal{N}^+$, which is twice continuous differentiable. $J_p(W)$ and $H_p(W)$ are the corresponding Jacobian and Hessian matrix. As $\lambda\to\infty$, the convexity region monotonically expands to the entire parameter space except for the subregion $S:=\{W\in R^n|rank(H_p(W))<n,H_p(W<0)\}$.

Intuitively, the use of the RAE was motivated by its emphasizing large individual deviations in approximating functions and exponentially optimizing parameters, thereby avoiding such large individual deviations and achieving robust performances. When the convexity index λ increases to infinity, the convexity region in the parameter space of RAE expands monotonically to the entire space except for the intersection of a the finite number of lower-dimensional sets. The number of sets increases rapidly as the number m of training samples increases. Roughly speaking, larger λ and m cause the size of the convexity region to grow larger respectively in the error space of RAE[27].

When $\lambda\to\infty$, the error space can be perfectly stretched to be strictly convex, thus avoid the local optimum to guarantee a global optimum. Although RAE works well in theory, it is not bounded and suffers from the exponential magnitude and arithmetic overflow when using gradient descent in implementations.

4 Normalized Risk-Averting Error

$$\begin{aligned}&NRAE_p\left(f\left(\boldsymbol{x}_i,\boldsymbol{W}\right),y_i\right)\\&=\frac{1}{\lambda^p}\log RAE_p\left(f\left(\boldsymbol{x}_i,\boldsymbol{W}\right),y_i\right)\\&=\frac{1}{\lambda^p}\log\frac{1}{m}\sum_{i=1}^{m}\left[f\left(\boldsymbol{x}_i,\boldsymbol{W}\right)^{y_i}\left(1-f(\boldsymbol{x}_i,\boldsymbol{W})\right)^{(1-y_i)}\right]^{\lambda^p}\end{aligned} \tag{4}$$

If $RAE_p(f(x_i,W),y_i)$ is convex, it is quasiconvex. $\log$ function is monotonically increasing, so the composition $\log RAE_p(f(x_i,W),y_i)$ is quasi-convex. $\log$ is a strictly monotone function and $NRAE_p(f(x_i,W),y_i)$ is quasi-convex, so it shares the same local and global minimizer with $RAE_p(f(x_i,W),y_i)$[27].

The convexity region of NRAE is consistent with RAE. To interpret this statement in another perspective, the log function is a strictly monotone function. Even if RAE is not strictly convex, NRAE still shares the same local and global optimum with RAE. If we define the mapping function $f : RAE \rightarrow NRAE$, it is easy to see that f is bijective and continuous. Its inverse map f^{-1} is also continuous, so that f is an open mapping. Thus, it is easy to prove that the mapping function f is a homeomorphism to preserve all the topological properties of the given space. The above theorems state the consistent relations among NRAE, RAE and cross-entropy loss. It is proven that the greater the convexity index λ, the larger is the convex region is. Intuitively, increasing λ creates tunnels for a local-search minimization procedure to travel through to a good local optimum. However, we care about the justification on the advantage of NRAE.

Given training samples $\{\boldsymbol{X}, y\} = \left\{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), ..., (\boldsymbol{x_m}, y_m)\right\}$ and the model $f(x_i, \boldsymbol{W})$ with parameters W. If $\lambda^p \geq 1, p \in \mathcal{N}^+$, then both $RAE_p(f(x_i, W), y_i)$ and $NRAE_p(f(x_i, W), y_i)$ always have the higher chance to find a better local optimum than the cross-entropy error due to the expansion of the convexity region.

Because $NRAE_p(f(x_i, W), y_i)$ is quasi-convex, sharing the same local and global optimum with $RAE_p(f(x_i, W), y_i)$, the above conclusions are still valid.

Roughly speaking, NRAE always has a larger convexity region than the cross-entropy error in terms of their Hessian matrix when $\lambda \geq 1$. This property guarantees the higher probability to escape poor local optima using NRAE. In the worst case, NRAE will perform as good as standard cross-entropy error if the convexity region shrinks as λ decreases or the local search deviates from the ?tunnel? of convex regions. More specifically, $NRAE_p(f(x_i, W), y_i)$ approaches the standard Lp-norm error as $\lambda_p \rightarrow 0$ and approaches the minimax error criterion in $f_W \alpha_{max}(W)$ as $\lambda_p \rightarrow \infty$.

5 Learning Methods

Under some regularity conditions, the convexity region of $J(\boldsymbol{W})$ expands monotonically as λ increases. To make advantage of a larger convexity region of $J(\boldsymbol{W})$ at a greater λ and avoid computer overflow, we are tempted to minimize $NRAE_p\left(f\left(\boldsymbol{x}_i, \boldsymbol{W}\right), y_i\right)$ at a λ as large as possible. However, at a very large λ, the training process is extremely slow or grinds to a halt, which phenomenon is called training stagnancy:

minimizing $f_W \alpha_{max}(W)$ for $\lambda \gg 1$ minimizes virtually the largest $f_W \alpha_{max}(W)$. The architecture of the deep learning machine is therefore redundant for the approximation. When all the weights are adjusted to achieve the approximation, they tend to become similar or duplicated, thus causing rank deficiency, violating the regularity conditions required for convexification of $J(\boldsymbol{W})$.

In the Adaptive Normalized Risk-Avering Training (ANRAT) approach[2, 11, 27], we learn λ adaptively in error backpropagation by considering λ as a parameter instead of a hyperparameter. The learning procedure is standard batch SGD. We show it works quite well in theory and practice. The

loss function of ANRAT is

$$l(\boldsymbol{W},\lambda)=\frac{1}{\lambda^p}\log\frac{1}{m}\sum_{i=1}^{m}\left[f\left(\boldsymbol{x}_i,\boldsymbol{W}\right)^{y_i}\left(1-f(\boldsymbol{x}_i,\boldsymbol{W})\right)^{(1-y_i)}\right]^{\lambda^p}+a\|\lambda\|^{-q} \tag{5}$$

We also use a penalty term a $\|\lambda\|^{-q}$ to control the changing rate of λ. While minimize the NRAE score, small λ is penalized to regulate the convexity region. a is a hyperparameter to control the penalty index. $\alpha_i(\mathbf{W})=f(\mathbf{x}_i,\mathbf{W})^{y_i\lambda^p-1}$, $\beta_i(\mathbf{W})=(1-f(\mathbf{x}_i,\mathbf{W}))^{\lambda^p-\lambda^p y_i-1}$ and $\gamma_i(\mathbf{W})=f\left(\boldsymbol{x}_i,\boldsymbol{W}\right)^{y_i}\left(1-f(\boldsymbol{x}_i,\boldsymbol{W})\right)^{(1-y_i)}$. The first order derivatives on weight and λ are

$$\begin{aligned}
\frac{dl(W,\lambda)}{dW}&=\frac{\sum_{i=1}^{m}\alpha_i(\mathbf{W})\beta_i(\mathbf{W})f(\mathbf{x}_i,\mathbf{W})(1-f(\mathbf{x}_i,\mathbf{W}))\lambda^p(y_i-f(\mathbf{x}_i,\mathbf{W}))\frac{\partial f(\boldsymbol{x}_i,\boldsymbol{W})}{\partial\boldsymbol{W}}}{\sum_{i=1}^{m}\alpha_i(\mathbf{W})\beta_i(\mathbf{W})}\\
\frac{dl(W,\lambda)}{d\lambda}&=\frac{-p}{\lambda^{p+1}}\log\frac{1}{m}\sum_{i=1}^{m}\left[f\left(\boldsymbol{x}_i,\boldsymbol{W}\right)^{y_i}\left(1-f(\boldsymbol{x}_i,\boldsymbol{W})\right)^{(1-y_i)}\right]^{\lambda^p}\\
&\quad+\frac{1}{\lambda^p}\frac{\sum_{i=1}^{m}p\lambda^{p-1}\log\gamma_i(\mathbf{W})\gamma_i(\mathbf{W})^{\lambda^p}}{\sum_{i=1}^{m}\gamma_i(\mathbf{W})^{\lambda^p}}\\
&\quad-\quad aq\lambda^{-q-1}
\end{aligned} \tag{6}$$

$$\frac{dl(W,\lambda)}{d\lambda}\approx\frac{q}{\lambda}\left(L-\text{ cross entropy error }-NRAE\right) \tag{7}$$

This training approach has more flexibility. The gradient on λ as the weighted difference between NRAE and the standard cross entropy error, enables NRAE to approach the cross entropy error by adjusting λ gradually. Intuitively, it keeps searching the error space near the manifold of the cross entropy error to find better optima in a way of competing with and at the same time relying on the standard cross entropy error space. The penalty weight a and index q control the convergence speed by penalizing small λ. Smaller a emphasizes tuning λ to allow faster convergence speed between NRAE and cross entropy error. Larger a forces larger λ for a better chance to find a better local optimum but runs the risk of plateaus and deviating far from the stable error space. q regulates the magnitude of λ and its derivatives in gradient descent.

This loss function is minimized by batch SGD without complex methods, such as momentum, adaptive/hand tuned learning rates or tangent prop. The learning rate and penalty weight a are selected in $\{1, 0.5, 0.1\}$ and $\{1, 0.1, 0.001\}$ on validation sets respectively. The initial λ is fixed at 10. We use the hold-out validation set to select the best model, which is used to make predictions on the test set. All experiments are implemented quite easily in Python and Theano to obtain GPU acceleration[22]. The MNIST dataset[?] consists of hand written digits $0-9$ which are 28×28 in size. There are $60,000$ training images and $10,000$ testing images in total. We use 10000 images in training set for validation to select the hyperparameters and report the performance on the test set.We test our method on this dataset without data augmentation.

6 Results and Discussion

On the MNIST dataset we use the same structure of LeNet5 with two convolutional max-pooling layers but followed by only one fully connected layer and a densely connected softmax layer. The first convolutional layer has 20 feature maps of size 5×5 and max-pooled by 2×2 non-overlapping windows. The second convolutional layer has 50 feature maps. with the same convolutional and max-pooling size. The fully connected layer has 500 hidden units. An $l2$ prior was used with the strength 0.05 in the Softmax layer. Trained by ANRAT, we can obtain a test set error of 0.56%, which is the best result we are aware of that does not use dropout on the pure ConvNets. We summarize the best published results on the standard MNIST dataset in Table 1.

Method	Error $\%$
ConvNets + ANRAT	0.56
ConvNets + ANRAT + dropout	0.33

Table 1: Test set misclassification rates of the best methods that utilized convolutional networks on the original MNIST dataset using single model.

7 Conclusion

Six advantages of the proposed method:

- no need for repeated trainings (or consistent performances among different training sessions with different initialization seeds);
- applicability to virtually any data fitting;
- conceptual simplicity and math-ematical justification;
- possibility of its use jointly with other training methods;
- a smaller DLM with the same or similar performance;
- a same-architecture DLM with a better performance.

References

[1] S. Zlobec. On the liu-floudas convexification of smooth programs. *Journal of Global Optimization*, 32(3):401–407, 2005. PT: J; UT: WOS:000232059300005.

[2] K. Glover and J. C. Doyle. State-space formulas for all stabilizing controllers that satisfy an h infinity-norm bound and relations to risk sensitivity. *Systems Control Letters*, 11(3):167–172, 1988. PT: J; UT: WOS:A1988Q410700001.

[3] D. H. Jacobson. Optimal stochastic linear-systems with exponential performance criteria and their relation to deterministic differential games. *Ieee Transactions on Automatic Control*, AC18(2):124–131, 1973. PT: J; UT: WOS:A1973O960800006.

[4] James Ting-Ho Lo, Yichuan Gui, and Yun Peng. Overcoming the local-minimum problem in training multilayer perceptrons by gradual deconvexification. *2013 International Joint Conference on Neural Networks (Ijcnn)*, 2013. PT: S; CT: International Joint Conference on Neural Networks (IJCNN); CY: AUG 04-09, 2013; CL: Dallas, TX; SP: Int Neural Network Soc, IEEE Computat Intelligence Soc; UT: WOS:000349557200089.

[5] Hiroshi Ninomiya. *Distributed Robust Training of Multilayer Neural Netwroks Using Normalized Risk-Averting Error*. 2014. PT: B; CT: IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain; CY: DEC 09-12, 2014; CL: Orlando, FL; SP: IEEE, IEEE Computat Intelligence Soc; UT: WOS:000380501000020.

[6] Jawes Ting-Ho Lo, Yichuan Gui, and Yun Peng. Training deep neural networks with gradual deconvexification. *2016 International Joint Conference on Neural Networks (Ijcnn)*, pages 1000–1007, 2016. PT: S; CT: International Joint Conference on Neural Networks (IJCNN); CY: JUL 24-29, 2016; CL: Vancouver, CANADA; SP: IEEE, IEEE Computat Intelligence Soc, Int Nueral Network Soc, Evolutionary Programming Soc, IET, IEEE BigData, Gulf Univ Sci Technol; UT: WOS:000399925501024.

[7] James Ting-Ho Lo, Yichuan Gui, and Yun Peng. Solving the local-minimum problem in training deep learning machines. *Neural Information Processing, Iconip 2017, Pt i*, 10634:166–174, 2017. PT: S; CT: 24th International Conference on Neural Information Processing (ICONIP); CY: NOV 14-18, 2017; CL: Guangzhou, PEOPLES R CHINA; SP: Chinese Acad Sci, Inst Automat, Guangdong Univ Technol, S China Univ Technol, Springers Lecture Notes Comp Sci, IEEE CAA Journal Automatica Sinica, Asia Pacific Neural Network Soc; PN: I; UT: WOS:000576768000018.

[8] James Ting-Ho Lo, Yichuan Gui, and Yun Peng. The normalized risk-averting error criterion for avoiding nonglobal local minima in training neural networks. *Neurocomputing*, 149:3–12, 2015. PT: J; PN: A; UT: WOS:000360028800002.

[9] Y. LeCun, L. Bottou, G. B. Orr, and K. R. Muller. Efficient backprop. *Neural Networks: Tricks of the Trade*, 1524:9–50, 1998. PT: J; UT: WOS:000083105900001.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. PT: J; UT: WOS:000402555400026.

[11] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. PT: J; UT: WOS:000237698100002.

[12] C. Poultney, S. Chopra, and Y. L. Cun. Efficient learning of sparse representations with an energy-based model. pages 1137–1144, 2006.

[13] J. Mairal, P. Koniusz, Z. Harchaoui, and et al. Convolutional kernel networks. pages 2627–2635, 2014.

[14] J. T. H Lo, Yichuan Gui, and Yun Peng. Overcoming the local-minimum problem in training multilayer perceptrons with the nrae training method. pages 440–7, 2012.

[15] Q. V. Le, J. Ngiam, and Z. Chen. Tiled convolutional neural networks. pages 1279–1287, 2010.

[16] Q. V. Le, J. Ngiam, A. Coates, and et al. On optimization methods for deep learning. In *Conference: Proc. 28th Int. Conf. Int. Conf. Mach. Learn nbsp*, pages 265–272, 2011.

[17] Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. PT: J; UT: WOS:000310345000010.

[18] Yichuan Gui, James Ting-Ho Lo, and Yun Peng. A pairwise algorithm for training multilayer perceptrons with the normalized risk-averting error criterion. *Proceedings of the 2014 International Joint Conference on Neural Networks (Ijcnn)*, pages 344–351, 2014. PT: S; CT: International Joint Conference on Neural Networks (IJCNN); CY: JUL 06-11, 2014; CL: Beijing, PEOPLES R CHINA; SP: IEEE; UT: WOS:000371465700052.

[19] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, 2010.

[20] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(6):1137–1155, 2003. PT: J; CT: Workshop on Machine Learning Methods for Text and Images; CY: 2001; CL: VANCOUVER, CANADA; UT: WOS:000186002400006.

[21] Yichuan Gui, J. T. H Lo, and Yun Peng. A pairwise algorithm for training multilayer perceptrons with the normalized risk-averting error criterion. pages 358–65, 2014.

[22] F. Bastien, P. Lamblin, R. Pascanu, and et al. Theano: new features and speed improvements. 2012.

[23] Y. N. Dauphin, R. Pascanu, and G. Gulcehre. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *In Advances in Neural Information Processing Systems*, pages 2933–2941, 2014.

[24] W. B. Liu and C. A. Floudas. A remark on the gop algorithm for global optimization. *Journal of Global Optimization*, 3(4):519–521, 1993. PT: J; UT: WOS:A1993MM93300007.

[25] J. L. Speyer, J. Deyst, and D. H. Jacobson. Optimization of stochastic linear-systems with additive measurement and process noise using exponential performance criteria. *Ieee Transactions on Automatic Control*, AC19(4):358–366, 1974. PT: J; UT: WOS:A1974T590200007.

[26] James Ting-Ho Lo. Convexification for data fitting. *Journal of Global Optimization*, 46(2):307–315, 2010. PT: J; UT: WOS:000273403000010.

[27] Zhiguang Wang, Tim Oates, and James Lo. *Adaptive Normalized Risk-Averting Training for Deep Neural Networks*. ASSOC ADVANCEMENT ARTIFICIAL INTELLIGENCE, PALO ALTO; 2275 E BAYSHORE RD, STE 160, PALO ALTO, CA 94303 USA, 2016. PT: B; CT: 30th Association-for-the-Advancement-of-Artificial-Intelligence (AAAI) Conference on Artificial Intelligence; CY: FEB 12-17, 2016; CL: Phoenix, AZ; SP: Assoc Advancement Artificial Intelligence; NR: 26; TC: 0; PG: 7; GA: BN6JG; UT: WOS:000485474202034.