ImageNet Classification Using the Random Oblique Stumps Algorithm on the Graphics Processing Unit

Minh-Trung Nguyen[1], Truong-Thanh Ma[1], Thanh-Nghi Do[1,2]

[1] College of Information Technology
Can Tho University, 94000-Cantho, Vietnam
[2] UMI UMMISCO 209 (IRD/UPMC)
Sorbonne University, Pierre and Marie Curie University - Paris 6, France
dtnghi@ctu.edu.vn

Abstract. In this paper, we propose a novel random oblique stumps algorithm (ROS) using the graphics processing unit (GPU) to perform the classification task of the ImageNet challenge. The ROS-GPU algorithm bases on our proposed multi-class linear discriminant analysis (MC-LDA) on GPU, with the One-Versus-All (OVA) multi-class strategy and the under-sampling technique to train in the parallel manner ensemble ROS classifiers. The empirical test results on the ImageNet dataset show that our ROS-GPU algorithm with the GPU (Gigabyte GeForce RTX 3060 12GB GDDR6, 3584 CUDA cores) is faster and more accurate than the state-of-the-art parallel SVM algorithms and the parallel covertree-kNN run on a PC (Intel(R) Core i7-4790 CPU, 3.6 GHz, 4 cores, 32GB RAM).

Keywords: ImageNet challenge · Random oblique stumps · Multi-class linear discriminant analysis · Graphics processing unit.

1 Introduction

The ImageNet dataset[6], consisting of over 14 million images across 21,841 classes, presents numerous challenges to computer vision and machine learning communities. There are some characteristics, making the ImageNet dataset challenging: large scale, high variability, fine-grained categories, object scale, limited training data per class, background clutter and interclass variability.

The process of image classification is to automatically assign an image to one of the predefined classes, including two crucial steps: feature extraction and the training of the classifier.

Traditional methodologies, as reflected in works such as [2,8,17,22,25,30], rely on well-known handcrafted features like the scale-invariant feature transform (SIFT [18,19]), the bag-of-words model (BoW), and the training of support vector machines (SVM [28]). In contrast, deep neural networks, LeNet [16] attempts to learn visual features and classifiers simultaneously within a unified algorithm. And then, recent developments of deep networks, such as AlexNet [15], VGGNet

[24], ResNet50 [12], Inception v3 [26], Xception [5], DenseNet [14], EfficientNet [27], and MobileNet [13], consistently achieve prediction correctness more over 70% for the ImageNet dataset. The vision transformer (ViT [9]) proposes a departure from deep neural networks by applying the transformer architecture, which was originally developed for natural language processing tasks, to images. ViT gives the competitive classification correctness compared to recent deep networks.

In this paper, we present the novel random oblique stumps algorithm (ROS) using the graphics processing unit (GPU) to replace the Multi-Layer Perceptron (MLP) classification head in the ViT network architecture for classifying the ImageNet challenge. The ROS-GPU algorithm bases on our proposed parallel multi-class linear discriminant analysis (MC-LDA) on GPU, in which the MC-LDA algorithm learns on GPU binary models in the One-Versus-All (OVA) multi-class strategy, from training samples with under-sampling the negative class (the rest) and the positive class. The empirical test results on the ImageNet dataset show that our ROS-GPU algorithm with the GPU (Gigabyte GeForce RTX 3060 12GB GDDR6, 3584 CUDA cores) is faster and more accurate than the state-of-the-art parallel SVM algorithms and the parallel covertree-kNN run on a PC (Intel(R) Core i7-4790 CPU, 3.6 GHz, 4 cores, 32GB RAM). Our ROS-GPU algorithm takes 10.49 minutes to classify the ImageNet dataset having 1,281,167 images in 768 visual features into 1,000 classes with an accuracy of 89.44%.

The remainder of this paper is organized as follows. Section 2 briefly presents the ROS-GPU algorithm for classifying the ImageNet dataset. Section 3 shows the experimental results before conclusions and future works presented in section 4.

2 Random oblique stumps algorithm on the graphics processing unit for the ImageNet challenge

To perform the classification task of the ImageNet challenge, the pre-trained ViT [9] without the top layer (i.e. Multi-Layer Perceptron - MLP head) is used to extract 768 visual features from images of ImageNet dataset, after that, we develop the novel random oblique stumps algorithm (ROS) using the graphics processing unit (GPU) to perform the classification for the ImageNet challenge. It means that the ROS classifier substitutes for the MLP head in the ViT architecture.

2.1 Random oblique stumps algorithm

Our proposed learning algorithm of random oblique stumps algorithm on the graphics processing unit (ROS-GPU) is designed to train multiple random oblique stumps (ROS), as illustrated in Algorithm 1 and Figure 1.

A ROS classifier constitutes a one-level decision tree, comprising a root node directly connected to terminal nodes. In contrast to decision tree algorithms that

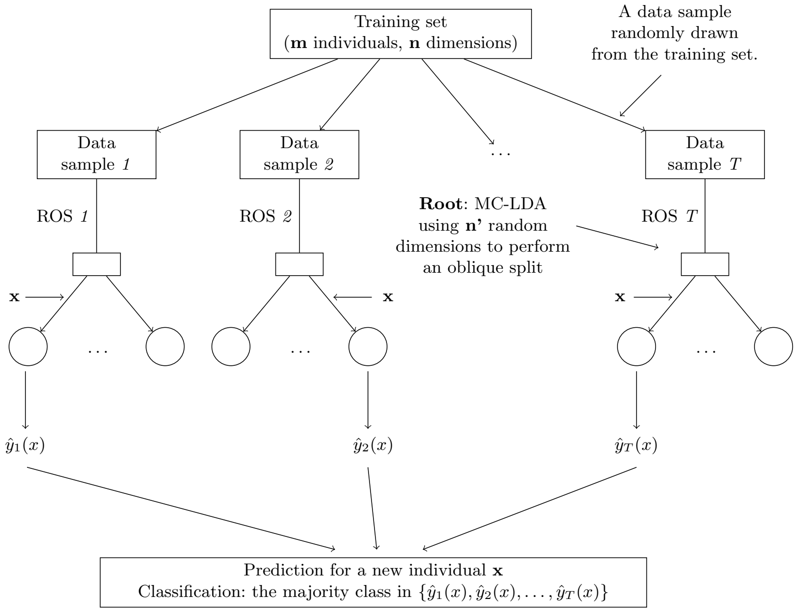
Training set
(**m** individuals, **n** dimensions)
A data sample randomly drawn from the training set.
Data sample *1*
Data sample *2*
...
Data sample *T*
ROS *1*
ROS *2*
ROS *T*
Root: MC-LDA using **n'** random dimensions to perform an oblique split
x
$\hat{y}_1(x)$
$\hat{y}_2(x)$
$\hat{y}_T(x)$
Prediction for a new individual **x**
Classification: the majority class in $\{\hat{y}_1(x), \hat{y}_2(x), \ldots, \hat{y}_T(x)\}$

Fig. 1: Learning algorithm for random oblique stumps (ROS)

Algorithm 1: Training algorithm for random oblique stumps (ROS)

```
input :
        Training dataset $D = \{D_1, D_2, \ldots, D_T\}$
        Number of random dimensions $n'$
output:
        Ensemble of $T$ random oblique stumps (ROS)
1 begin
2     for $i \leftarrow 1$ to $T$ do
3         Loading data block $D_i$
4         $ROS_i$ = MC-LDA($D_i, n'$)
5     end
6     Return $T$ ROS models = $\{ROS_1, ROS_2, \ldots, ROS_T\}$
7 end
```

use a single dimension for finding the best node splitting [3,4], our ROS algorithm employs the multi-class linear discriminant analysis (MC-LDA). It adopts the One-Versus-All (OVA) multi-class strategy, along with an under-sampling technique. This approach facilitates the creation of oblique decision stumps in n' randomly sampled dimensions from the original n dimensions, enhancing the robustness of the stumps.

The prediction for a new datapoint x (as illustrated in Algorithm 2) is determined through a majority vote among the classification results $\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T$ generated by T ROS classifiers.

Algorithm 2: Prediction for a new datapoint x with ROS classifiers

```
input :
        A new datapoint $x$
        $T$ ROS classifiers
output:
        Predicted class $\hat{y}$
1 begin
2     for $t \leftarrow 1$ to $T$ do
3         $\hat{y}_t = predict(x, ROS_t)$
4     end
5     $\hat{y} =$ majority-vote$\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T\}$
6     Return predicted class $\hat{y}$
7 end
```

It is evident that we propose to use the multi-class linear discriminant analysis (MC-LDA) to perform random oblique stumps while training the ROS classifier in Algorithm 1.

2.2 Multi-class linear discriminant analysis

We start with linear binary classification task, as depicted in Figure 2. The dataset comprises m datapoints x_i $(i = 1, \ldots, m)$ in the n-dimensional input space R^n, with corresponding classes $y_i = \pm 1$. Linear discriminant analysis (LDA [11]) tries to find the best direction w to maximize the separation between the means and the variances of the projection of the two classes onto w. We denote $\bar{x}_{+1}$, $\bar{x}_{-1}$ the means of the two classes, and μ_{+1}, μ_{-1} the means of their projections on w.

The distance between these means of the projection is:

$$(\mu_{+1} - \mu_{-1})^2 = w^T(\bar{x}_{+1} - \bar{x}_{-1})(\bar{x}_{+1} - \bar{x}_{-1})^T w = w^T S_B w \quad (1)$$

$$\text{with } S_B = (\bar{x}_{+1} - \bar{x}_{-1})(\bar{x}_{+1} - \bar{x}_{-1})^T$$

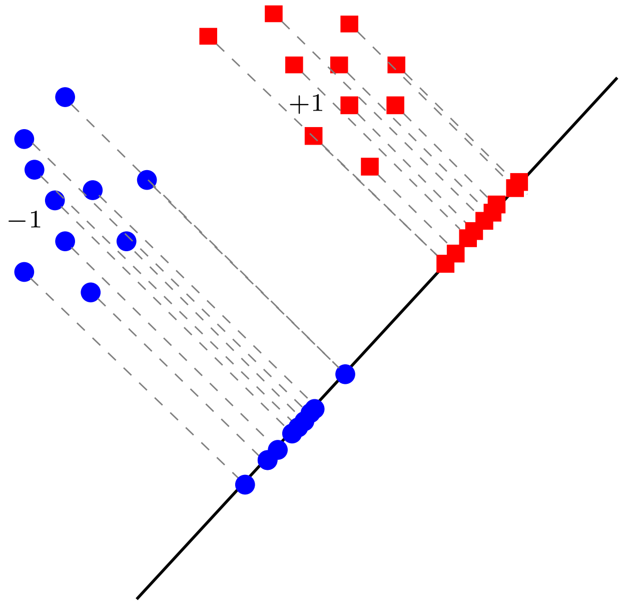

Fig. 2: LDA projection for separating the datapoints into two classes

The scatter matrices of the two classes are defined as:

$$\tilde{s}_{+1} = \sum_{x \in class+1} (x - \bar{x}_{+1})(x - \bar{x}_{+1})^T \tag{2}$$

$$\tilde{s}_{-1} = \sum_{x \in class-1} (x - \bar{x}_{-1})(x - \bar{x}_{-1})^T \tag{3}$$

And then the variance of the projections of the two classes is:

$$s_{+1}^2 + s_{-1}^2 = w^T(\tilde{s}_{+1} + \tilde{s}_{-1})w = w^T S_w w \tag{4}$$

$$\text{with } S_w = \tilde{s}_{+1} + \tilde{s}_{-1}$$

LDA tries to find the direction w to pursue the goal of the maximum separation between the means and the variances of the projection of the two classes onto w. This is accomplished through the maximization of the ratio of the between-class variance and the within-class variance, denoted by $J(w)$ in (5):

$$\max_w J(w) = \frac{(\mu_{+1} - \mu_{-1})^2}{s_{+1}^2 + s_{-1}^2} = \frac{w^T S_B w}{w^T S_w w} \tag{5}$$

The solution to maximize the objective function $J(w)$ is to derive it and set it to zero. And then, training LDA to find the direction w is nothing more than solving the linear system of equations (6):

$$w \propto S_w^{-1}(\bar{x}_{+1} - \bar{x}_{-1}) \tag{6}$$

To determine the cutting point on this projection is computed by the Shannon's entropy [23] or the Gini index [4].

The LDA learning algorithm has the capability to deal with multi-class problems (c classes, $c \geq 3$). We propose to use the One-Versus-All training approach (OVA [28]) which involves learning a series of binary LDA models. For training the multi-class LDA (MC-LDA) model, the algorithm learns c binary LDA ones (depicted in Figure 3). Each binary model is trained to separate the i^{th} class from the remaining classes. And then, the prediction for a new datapoint is determined by a vote based on the largest distance.

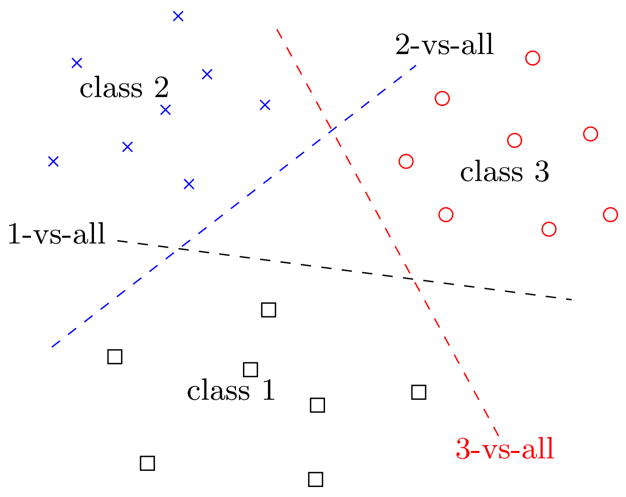

Fig. 3: Multi-class SVM (One-Versus-All)

Nevertheless, with the context of very large multi-class datasets such as ImageNet [6] comprising 1,281,167 images and $c = 1,000$ classes, it leads to two problems for the MC-LDA algorithm:

1. imbalanced classification issues arise during the training of binary LDA models, where the size of the i^{th} class (positive class) is significantly smaller than that of the remaining classes (negative class), making poor performance,
2. the algorithm's implementation is usually based on the classical Central Processing Unit (CPU) architecture, of course, the training time is long.

To overcome these issues, we propose an improved MC-LDA algorithm for training parallel manner ensemble binary LDA classifiers on standard computers using graphic processing units (GPU).

In the training dataset comprising c classes, our improved MC-LDA trains c binary classifiers. Typically, for each i^{th} class, the MC-LDA focuses on training the corresponding binary model (dedicated to distinguishing class c_i from the rest). This training process involves under-sampling the negative class (representing the rest of the classes, i.e. 6.67 times the positive class) and the positive class c_i. Separating the positive class from a subset of the negative class is easier than separating the classes on the full set. Therefore, our proposal not only reduces the complexity of the training algorithm by 100 times but also improves the ability to separate the positive class from the negative class.

Furthermore, we develop a parallel version of MC-LDA algorithm based on GPU. Leveraging the CUBLAS library, the MC-LDA algorithm executes matrix computations on the GPU, harnessing the power of massively parallel computing architecture. Consequently, this implementation achieves accelerated training speeds.

Hence, our proposed learning algorithm of random oblique stumps uses MC-LDA to train an ensemble of random oblique stumps on the GPU, referred to as ROS-GPU.

3 Experimental results

We are interested in the assessment of the random oblique stumps algorithm on the graphics processing unit (ROS-GPU) for classifying the ImageNet challenge dataset. Consequently, there is a need to evaluate its performance concerning both training time and classification accuracy.

3.1 Software programs

We implemented the ROS-GPU algorithm in C/C++ using library cuBLAS [20]. We would like to compare with the best state-of-the-art linear SVM algorithm, LIBLINEAR [10] implemented in C/C++ (the parallel version on multi-core computers with OpenMP [21]) and our last algorithm kSVM [7], parallelized on multi-core CPUs, implemented in C/C++ using the library ATLAS [29], the parallel multi-class linear SVM on multi-core CPUs, using the Stochastic Gradient Descent, called SVM-SGD. We have developed in C/C++: the parallel algorithm of k nearest neighbors using the multi-dimensional indexing covertree [1], denoted by covertree-kNN on multi-core computers with OpenMP [21];

We use a machine (PC) Linux Fedora 34, Intel(R) Core i7-4790 CPU, 3.6 GHz, 4 cores and 32 GB main memory, with the GPU (Gigabyte GeForce RTX 3060 12GB GDDR6, 3584 CUDA cores). Our ROS-GPU algorithm trains the ensemble of ROS classifiers on the GPU. LIBLINEAR, kSVM learn classification models on the PC with 4 cores.

3.2 ImageNet dataset

The experimental outcomes are assessed using the ImageNet challenging ILSVRC-2012 dataset [6], comprising 1,281,167 images and 1,000 classes. This dataset is widely recognized as the most prominent benchmark for image classification [5,8,12,22,24,26,30] and serves as a comprehensive evaluation.

The pre-trained Vision Transformer (ViT [9], the Base variant with 16×16 input patch size, without the Multi-Layer Perceptron (MLP Head), is used to extract 768 features from images of the ImageNet dataset.

The ImageNet dataset is randomly partitioned into a trainset (1,024,933 images) and a testset (256,234 images).

Our evaluation is based on the overall accuracy, specifically the Top-1 accuracy or simply accuracy, which measures the percentage of correctly classified images in comparison to the total number of images within the testset. Given the dataset's 1,000 classes, a random guess performance would yield an accuracy of 0.1%.

3.3 Tuning parameters

For training LIBLINEAR SVM models, the positive constant C is tuned to keep the trade-off between the margin size and the errors. We use the cross-validation (hold-out) protocol to find-out the best value $C = 100,000$. Furthermore, LIBLINEAR using the L2-regularized logistic regression loss function give classification results very closed to the MLP classifier in ViT.

The kSVM algorithm trains in the parallel way 1000 local non-linear SVM models using a Gaussian kernel function with $\gamma = 0.001$ and the positive constant $C = 100,000$.

The ROS-GPU algorithm learns 20 ROS classifiers from the training sample size being 51,247 using $n' = 480$ dimensions randomly chosen from 768 original ones.

The SVM-SGD algorithm learns the classification model with the learning rate $\eta = 0.001$ and 200 epochs.

For the covertree-kNN algorithm, the parameter k is set to 1, and the Euclidean distance metric is used in the experiment.

Due to the PC (Intel(R) Core i7-4790 CPU, 4 cores) used in the experimental setup, the number of threads is setting to 8 for all training tasks.

3.4 Classification results

We obtain classification results of ROS-GPU, LIBLINEAR and kSVM in Table 1, Fig. 4 and Fig. 5. The fastest training algorithm is highlighted in bold-faced. The same presentation format is accorded to performance in terms of classification accuracy.

Our ROS-GPU algorithm, implemented on GPU, successfully completed the ImageNet dataset classification task in 10.49 minutes, achieving an accuracy of 89.44%. In comparison, the kSVM algorithm, executed on a PC, attained

Table 1: Classification results

No	Algorithm	Time (min)	Accuracy (%)
1	LIBLINEAR	2,820.25	85.72
2	covertree-*k*NN	914.01	87.77
3	SVM-SGD	79.01	88.22
4	*k*SVM	43.32	88.20
5	ROS-GPU	**10.49**	**89.44**

an accuracy of 88.20% with a training duration of 43.32 minutes. The SVM-SGD takes 79.01 minutes to train the model with a classification correctness of 88.22%. The covertree-*k*NN gives a classification correctness of 87.77% with the prediction time of 914.01 minutes on a PC. The LIBLINEAR algorithm, also run on a PC, necessitated 2,820.25 minutes for training the classification model, resulting in an accuracy of 85.72%. Notably, this accuracy of LIBLINEAR aligns closely with the original ViT [9], typically 85.08% obtained by R50+ViT-B_16.

Regarding training time, the ROS-GPU implementation on GPU outperforms LIBLINEAR on a PC by a factor of 268.81. The ROS-GPU is 87.12 times faster than covertree-*k*NN. The ROS-GPU demonstrates a slight 7.53 and 4.13 times speed advantage over the SVM-SGD and the *k*SVM, respectively.

In terms of overall accuracy, the ROS-GPU exhibits an improvement of 3.72%, 1.67%, 1.24% and 1.22% compared to LIBLINEAR, covertree-*k*NN, *k*SVM and SVM-SGD, respectively.

The classification results show that our ROS-GPU algorithm is efficient for handling such large-scale multi-class datasets.

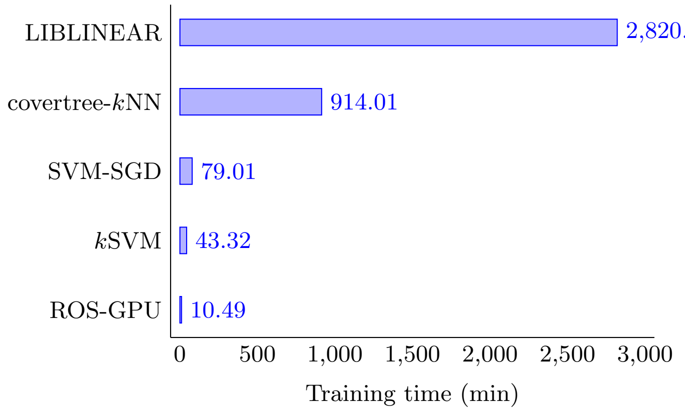

Fig. 4: Training time (min)

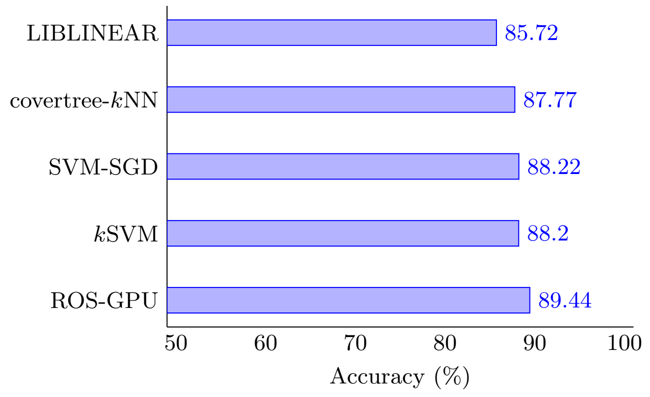

Fig. 5: Overall classification accuracy

4 Conclusion and future works

We have introduced the novel ROS algorithm on the GPU, which achieves high performances for classifying the ImageNet challenge dataset having 1,281,167 images with 1,000 classes. The ROS-GPU algorithm trains an ensemble of ROS classifiers in parallel on the GPU, where each ROS classifier is performed by the MC-LDA with the OVA multi-class strategy and an under-sampling technique. Numerical results on the ImageNet dataset demonstrate that our ROS-GPU algorithm is faster and more accurate than state-of-the-art parallel SVM algorithms, including LIBLINEAR, SVM-SGD, kSVM and the parallel covertree-kNN.

In the near future, we plan to develop an incremental learning algorithm specifically designed for edge devices. Additionally, we will explore weighted combinations of ROS classifiers to enhance classification accuracy.

Acknowledgments This research has received support from the European Union's Horizon research and innovation programme under the MSCA-SE (Marie Skłodowska-Curie Actions Staff Exchange) grant agreement 101086252; Call: HORIZON-MSCA-2021-SE-01; Project title: STARWARS (STormwAteR and WastewAteR networkS heterogeneous data AI-driven management). We would like to thank very much the College of Information Technology, Can Tho University.

References

1. Beygelzimer, A., Kakade, S., Langford, J.: Cover trees for nearest neighbor. In: Proceedings of the 23rd international conference on Machine learning. pp. 97–104. ACM (2006)
2. Bosch, A., Zisserman, A., Munoz, X.: Scene classification via pLSA. In: Proceedings of the European Conference on Computer Vision. pp. 517–530 (2006)

3. Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)
4. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.: Classification and Regression Trees. Wadsworth International, (1984)
5. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. CoRR **abs/1610.02357** (2016)
6. Deng, J., Berg, A.C., Li, K., Li, F.: What does classifying more than 10, 000 image categories tell us? In: Computer Vision - ECCV 2010 - 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part V. pp. 71–84 (2010)
7. Do, T., Poulet, F.: Parallel learning of local SVM algorithms for classifying large datasets. Trans. Large Scale Data Knowl. Centered Syst. **31**, 67–93 (2017)
8. Doan, T., Do, T., Poulet, F.: Large scale classifiers for visual classification tasks. Multimedia Tools Appl. **74**(4), 1199–1224 (2015)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021)
10. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research **9**(4), 1871–1874 (2008)
11. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: data mining, inference and prediction. Springer, 2 edn. (2009)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015)
13. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR **abs/1704.04861** (2017)
14. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR. pp. 2261–2269. IEEE Computer Society (2017)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Bartlett, P.L., Pereira, F.C.N., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States. pp. 1106–1114 (2012)
16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE. vol. 86, pp. 2278–2324 (1998)
17. Li, F., Perona, P.: A bayesian hierarchical model for learning natural scene categories. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA. pp. 524–531 (2005)
18. Lowe, D.: Object recognition from local scale invariant features. In: Proceedings of the 7th International Conference on Computer Vision. pp. 1150–1157 (1999)
19. Lowe, D.: Distinctive image features from scale invariant keypoints. International Journal of Computer Vision pp. 91–110 (2004)
20. NVIDIA, Vingelmann, P., Fitzek, F.H.: Cuda, cublas release: 12.2 (2023), https://developer.nvidia.com/cuda-toolkit
21. OpenMP Architecture Review Board: OpenMP application program interface version 3.0 (2008), http://www.openmp.org/mp-documents/spec30.pdf

22. Perronnin, F., Sánchez, J., Liu, Y.: Large-scale image categorization with explicit data embedding. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 2297–2304 (2010)
23. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo, CA (1993)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
25. Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: 9th IEEE International Conference on Computer Vision (ICCV 2003), 14-17 October 2003, Nice, France. pp. 1470–1477 (2003)
26. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. CoRR **abs/1512.00567** (2015)
27. Tan, M., Le, Q.V.: Efficientnetv2: Smaller models and faster training (2021)
28. Vapnik, V.: The Nature of Statistical Learning Theory. Springer-Verlag (1995)
29. Whaley, R., Dongarra, J.: Automatically tuned linear algebra software. In: Ninth SIAM Conference on Parallel Processing for Scientific Computing (1999), cD-ROM Proceedings
30. Wu, J.: Power mean svm for large scale visual classification. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 2344–2351 (2012)