A Comparative Study of Loss Functions and Optimizers for Parking Space Classification

Paulo Mateus Luza Alves*
Department of Informatics
Universidade Federal do Paraná
Curitiba, PR, Brazil
paulomateus@ufpr.br

Henrique Margotte*
Department of Informatics
Universidade Federal do Paraná
Curitiba, PR, Brazil
hmargotte@inf.ufpr.br

Carmem Satie Hara
Department of Informatics
Universidade Federal do Paraná
Curitiba, PR, Brazil
carmem@inf.ufpr.br

Aurora Trinidad Ramirez Pozo
Department of Informatics
Universidade Federal do Paraná
Curitiba, PR, Brazil
aurora@inf.ufpr.br

Paulo Ricardo Lisboa de Almeida
Department of Informatics
Universidade Federal do Paraná
Curitiba, PR, Brazil
paulo@inf.ufpr.br

Vinícius Fülber Garcia
Department of Informatics
Universidade Federal do Paraná
Curitiba, PR, Brazil
vinicius@inf.ufpr.br

Abstract

Convolutional Neural Networks (CNNs) are being widely applied to the task of parking space classification, due to their high performance demonstrated in well-established state-of-the-art challenges. However, as deep learning models, CNNs encompass a large number of parameters, making the training and refinement process extensive, with loss functions and optimizers being two crucial training components that can help define the speed and quality of model training. Despite the wide use of these techniques in the literature, there is no consensus about the optimal combination for parking space classification, and a wide variety of different combinations have emerged. To address this concern, we propose a comparative study of loss function and optimizer combinations using a selected set of these functions and algorithms from the literature, with the aim of defining a possible optimal pair for this task. Using the PKlot dataset in our experiments, we show that the pair of Binary Cross Entropy (BCE) loss function and Stochastic Gradient Descent (SGD) optimizer has the best balance between accuracy and time required to train among the pairs created in this study, achieving an average accuracy of 98.2% and requiring approximately 4.6 epochs for fine-tuning.

Keywords

Machine Learning, Deep Learning, Computer vision, Parking Space Classification

1 Introduction

Deep learning approaches gained attention in the last decade in the area of computer vision, being broadly used in a variety of tasks, such as image segmentation, object detection and image classification. Regarding the last activity, the parking space classification (Fig. 1) appears relevant, being able to alleviate urban problems such as traffic jams, gas emissions, among others [1]. Thus, the use of Convolutional Neural Networks (CNNs) are widely applied in these tasks, due to their high performance demonstrated in state-of-the-art challenges such as ImageNet [2].

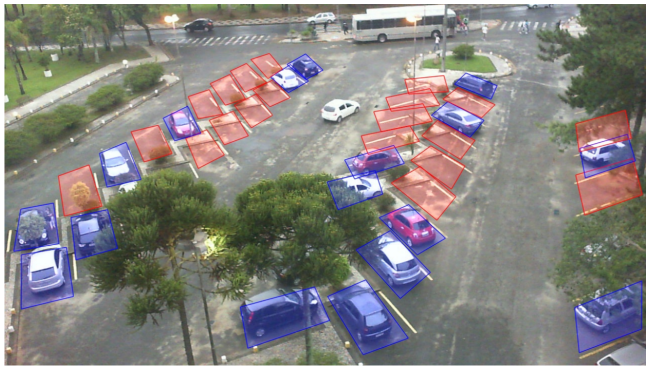


Figure 1: Image from the PKLot dataset, with parking lots classified as occupied (blue) and empty (red) [3].

As deep learning models, CNNs encompass a large number of parameters, making the training process extensive, highlighting the need for well-established training approaches to reduce time and computational power needed. Thus, loss functions and optimizers are two crucial components in the training process, helping to define how fast and well the models are trained. However, among all the possible options of loss functions and optimizers, there is no consensus in the literature on which combination to use, aiming to obtain better trained models.

A wide variety of loss functions and optimizers are used by the literature, as demonstrated in [3–8]. The predominance of Cross Entropy (CE) Loss and its variants as loss functions, and of Adam [9] as optimizer is notable, although the use of Adam variants and more traditional algorithms such as Stochastic Gradient Descent (SGD) for optimizer, are also present in the literature.

We propose a comparative study, combining pairs of loss functions and optimizers, among a selected set of algorithms from the literature, searching for a possible optimal combination in the parking space classification. Thus, the proposed work aims to answer the following research question:

*Both authors contributed equally to this research.

- RQ1: Which combination of loss function and optimizer, among the selected algorithms, yields the best results in the parking space classification?

Our results show that the Binary Cross Entropy (BCE) Loss with the SGD optimizer produces the best results, balancing accuracy and time required for fine-tuning, achieving 98.2% accuracy and requiring approximately 4.6 epochs. Also, using other loss functions, such as CE loss and BCE loss with logits (BCELogits), with the same optimizer, are good alternatives for fine-tuning the parking space classification task.

The code will be available on a GitHub[1] repository for reproducibility and to contribute for future related experiments. The PKLot [10] dataset can be found in its original paper.

The remainder of this paper is organized as follows: In Section 2, we show the related work. In Section 3, we detail the materials used in this work. In Section 4, we present the experimental protocol applied in our experiments, while in Section 5, we discuss the results obtained with it. Finally, in Section 6, we present our conclusions.

2 Related Work

This section presents some state-of-the-art works related to parking space classification using CNNs, focusing on presenting the loss functions and optimizers used. The results of such works are not discussed, as it's not possible to compare them fairly due to structural differences, such as different models and training/test protocols applied.

For example, Alves et al. [4] uses Cross Entropy Loss as the loss function, and Adam algorithm as the optimizer to train teacher and student CNN models in a knowledge distillation procedure for parking space classification. The MobileNetV3 [11] was used as teacher model, and a generic *Custom-3 layer* CNN as student, reaching accuracies of 96%.

In Thakur et al. [5], the ResNet50 and VGG16 models are trained with the Categorical Cross Entropy Loss as the loss function and SGD as the optimizer. The authors sought to create a low-cost solution for outdoor parking lot occupancy systems, reporting accuracies of 98.9% and 93.4%

Similarly, Zhang et al. [6] propose a lightweight CNN model to classify low-resolution images, aiming to address potential security and privacy concerns. The applied loss function was the CE loss, and the chosen optimizer was the AdamW [12]. The authors reported an accuracy of 91,68%.

Also, Hochuli et al. [7] propose a three-layer custom CNN scaled to resource-constrained devices for parking space classification. The CE was selected as the loss function, and the SGD algorithm as the optimizer. The reported accuracy was 80,9%, in addition a MobileNetV3 trained with the same configuration reached 89,9%.

In the same context, Goumiri et al. [8] propose a ultralight CNN model, composed of a single convolutional layer, trained with the BCE as the loss function and the Adam algorithm as optimizer. The results reported by the authors range from 97% to 99%.

Kujavski et al. [3] propose two lightweight models based on the architecture proposed by Hochuli et al. [7], using the Adam optimizer and CE loss. The authors aimed to achieve similar results of state of the art models in the parking space classification, but with fewer parameters (34x and 88x less parameters when compared to MobileNetV3 Large). The achieved accuracies were around 93% for both models in the 3 datasets used for testing.

Table 1 presents a summary of the works presented in this section, highlighting the variety of loss functions and optimizers used in the state-of-the-art. Nonetheless, to the best of our knowledge, there is no comprehensive study on the impact of different combinations of loss functions and optimizers on parking space classification, which is an open question in the literature. Thus, an empirical study that demonstrates the influence of different combinations of these tools has its relevance justified.

Authors	Loss function	Optimizer
Alves et al. [4]	Cross Entropy Loss	Adam
Thakur et al. [5]	Categorical Cross Entropy Loss	SGD
Zhang et al. [6]	Cross Entropy Loss	AdamW
Hochuli et al. [7]	Cross Entropy Loss	SGD
Goumiri et al. [8]	Binary Cross Entropy Loss	Adam
Kujavski et al. [3]	Cross Entropy Loss	Adam

Table 1: Parking space classification works summary.

3 Materials

In this Section, we intend to detail the materials used in this work, such as dataset, models, loss functions and optimizers, fostering the subsequent Sections of this paper.

3.1 Dataset

The PKLot dataset [10] was employed in our experiments. Table ?? details the main properties of this dataset. In total, the dataset comprises 695,899 images of parking spaces, collected in two different parking lots, named UFPR and PUCPR. Two camera angles were applied in the UFPR parking lot (UFPR04 and UFPR05) and one in the PUCPR parking lot. A five min time-lapse interval was applied for a period of more than 30 days, collecting images of different weather conditions (cloudy, sunny and rainy). Fig. 2 exemplifies the images of the dataset.

PKLot – 12,417 images					
			# Of parking spaces		
Subset	# Of days[a]	# Of images	Occupied	Empty	Total
UFPR04	49	3,791	46,125	59,720	105,845
UFPR05	52	4,152	97,426	68,359	165,785
PUCPR	43	4,474	194,229	230,040	424,269
Total	144	12,417	337,780	358,119	695,899

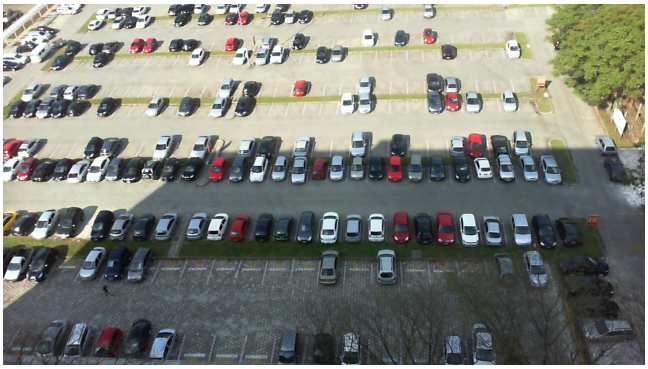
Table 2: Dataset used in the experiments.

[a]The quantity refers to all days collected per camera, not unique calendar days.

[1]Link not available to ensure the blind-review anonymity.

(a) UFPR04 image example.

(b) UFPR05 image example.



(c) PUCPR image example.

Figure 2: Examples of images within the PKLot dataset. Images (a), (b), and (c) were captured on sunny days.

3.2 CNN Model

We employed the large version of the MobileNetV3 [11] networks, also called MobileNetV3 Large. This architecture is designed for resource constrained devices, such as mobile devices, smart cameras and Internet of Things devices. It's comprises a series of improvements to the convolutional layers, such as residual connections, non-linearity transformations and squeeze-and-excitation blocks, with the goal of reducing computation while maintaining reasonable accuracy on state-of-the-art benchmarks.

3.3 Loss Functions

Loss functions are used to quantify the distance between the probabilities predicted by the model and the actual probabilities (labels). The output of these functions is used to update the parameters of the Neural Network, aiming to bring the predicted probabilities closer to the real ones. Since we are dealing with a binary classification, we selected loss functions widely applied in the literature for this type of problem, such as:

- CE Loss
- BCE Loss
- BCElogits Loss

All losses apply the concept of cross entropy, which aims to quantify the difference between two probabilities, usually the true distribution (labels) and the distribution predicted by the model. Thus, the lower the cross entropy, the better the correspondence between the predicted probabilities and the true ones, that is, this type of algorithm penalizes predictions that are far from the true class.

The CE function is the direct application of the concept to multiclass problems (including binary classification), BCE is the optimized version for binary problems and BCELogits applies an extra sigmoid layer to BCE, aiming to improve the numerical stability of the results.

3.4 Optimizers

Optimizers are algorithms that use the outputs of loss functions to compute the gradients, i.e. derivatives of the loss function in relation to the Neural Network parameters, to update model parameters, in a process known as gradient descent, aiming to minimize the loss value and, consequently, bring the predicted probabilities closer to the true ones. As possible options for optimizers, we selected well-established algorithms from the literature, such as:

- SGD
- RMSProp
- Adam
- AdamW

All optimizers use the concept of learning rate, which controls how fast the parameters are updated. SGD is the standard gradient descent algorithm, applying a global learning rate to all parameters. RMSProp applies the idea of moving average of squared gradients, adapting the learning rates for each parameter. Adam is the merge of SGD and RMSProp, also adding the concept of momentum, which applies the idea of inertia to gradient descent (older iterations either push the value of the current iteration forward or slow it down), and AdamW is an update to Adam, fixing the weight decay problem that existed in the previous version.

4 Experimental Protocol

To train our models we applied the leave-one-out technique to the PKLot subsets (one subset is left for testing and the others for training), generating three different test scenarios in which no images from the training set appear in the test set, as shown in Table 3. Each generated training subset was divided with a ratio of 70% to 30% (70% for training and 30% for validation) in chronological

order by days, avoiding images from the same day to appear in both subsets. Both sets were leveled by the least recurrent class.

Train & Validation	Test
UFPR04 + UFPR05	PUCPR
PUCPR + UFPR05	UFPR04
UFPR04 + PUCPR	UFPR05

Table 3: Train/Test scenarios created with leave-one-out.

All models were pre-trained on the ImageNet dateset. During fine-tuning, all layers of the networks were trained using images in the RGB format sized at 128×128, normalized with the ImageNet values, with a mini-batch of 32. We trained our models for 15 epochs. In all scenarios, the model with the lowest validation loss was chosen. As classification threshold, we select the value that minimized the Equal Error Rate in the validation dataset.

To create the loss function and optimizer pairs, the following strategy was adopted:

- For each loss function l_i in $i \in [1..n]$, where i is the index of each possible n loss function selected.
- For each optimizer o_j in $j \in [1..m]$, where j is the index of each possible m optimizer selected.
- We created a pair p_{ij} with the ith loss function and jth optimizer.

With this, we were able to create 12 different pairs. Default parameter values were used in all losses and optimizers to avoid possible tradeoffs or advantages between the selected algorithms. All training and testing was performed in a NVIDIA RTX 6000 Ada Generation GPU and the results shown are an average of five runs.

5 Results

To obtain a comprehensive comparison of the results obtained with each combination pair defined in the Section 4 the experiments held in this work were analyzed from two different points of view. In Section 5.1 we stress the discussion on the quantitative accuracy results achieved with each pair. In Section 5.2, we detail the number of epochs required to achieve the accuracies. The values shown are the weighted average of the 3 scenarios for the accuracy and the simple average for the epochs.

5.1 Combinations Versus Accuracy

Table 4 shows the average accuracy for each loss and optimizer pair. Higher accuracies are colored with green background, while lower ones are red, following a color gradient. The combination of the BCE loss function with the RMSprop optimizer presented the overall worst accuracy of 52.4% while all other pairs scored values higher than 95%. On the better results, the use of the SGD optimizer with the BCE, CE and BCELogits loss functions scored the three highest accuracies of 98.2% on the first two and 98.1% on the third, while the CE loss with AdamW optimizer scored the 4th position, with 97.8%, followed by the BCE and Adam's combination, with 97.7%. The use of RMSprop with the BCELogits and CE loss functions also performed lower than the other optimizers, scoring 96% and 96.2% accuracies, respectively. All other combinations scored values around 97%.

	Optimizer			
Loss	SGD (%)	RMSprop (%)	Adam (%)	AdamW (%)
CE	98.2 ± 0.6	96.2 ± 2.3	97.2 ± 1.4	97.8 ± 1.6
BCE	98.2 ± 0.4	52.4 ± 2.5	97.7 ± 2.3	96.9 ± 2.1
BCELogits	98.1 ± 0.5	96.0 ± 1.5	97.5 ± 1.3	96.9 ± 1.5

Table 4: Accuracy for each loss and optimizer pair (average ± std).

The lower results scored using the RMSprop indicate that this optimizer may not be suitable for the presented problem, what is enforced by the lack of its use on the related works presented on Section 2. On the other hand, the use of the SGD optimizer was highlighted with the overall better accuracies, proposing that its use could help improve the results of similar systems, specially when combined with the CE and BCE functions. Since Adam consists of a technique that combine the concepts of SGD and RMSprop, it is possible that the RMSprop's concepts are the responsible for lowering the results when compared to the pure SGD.

The use of the BCE loss function, expected to present improvement on binary problems when compared to the traditional CE, only did when in combination with the SGD and Adam optimizers, scoring the better results in each, while the BCELogits performed worst on most cases. Similar to that, the AdamW optimizer did not present much improvement when comparing to its predecessor, Adam, only in combination with the CE function, with both achieving lower results than the more traditional SGD.

5.2 Combinations Versus Epochs

Table 5 shows the average epochs performed by the model until the best validation was scored with each combination pairs training. The cells are filled with a color gradient from green, lower number of epochs, to red, higher number of epochs.

	Optimizer			
Loss	SGD	RMSprop	Adam	AdamW
CE	$5.\bar{6} \pm 5.8$	6.0 ± 4.8	6.0 ± 4.0	8.0 ± 3.7
BCE	$4.\bar{6} \pm 5.5$	$1.\bar{6} \pm 2.5$	8.0 ± 3.2	$7.\bar{6} \pm 4.1$
BCELogits	5.0 ± 6.0	$5.\bar{6} \pm 4.2$	5.0 ± 3.6	7.0 ± 4.0

Table 5: Epochs until convergence for each loss and optimizer pair (average ± std).

In contrast with the accuracy, the combination of the BCE loss function and the RMSprop optimizer was the pair that needed fewest epochs to converge, reaching the best validation on an average of $1.\bar{6}$ epochs on the tests executed. In second, the BCE function with SGD optimizer required an average of $4.\bar{6}$ epochs, followed by the use of BCELogits function with the SDG and Adam optimizers, that needed an average of 5 epochs each, with the second achieving the lower standard deviation of them.

The use of the CE function with AdamW optimizer and the BCE with Adam were the ones that took more epochs to train, with an average of 8 epochs. The two other pairs with the AdamW optimizer followed, scoring high values, at an average of 7.$\bar{6}$ epochs with BCE and 7.0 with BCELogits.

As the epochs give an intuition of how much time would be needed to train each model configuration, the average time to train and execute between the 5 runs and 3 scenarios for each combination, on the machine specified on Section 3, was also calculated and presented on Table 6. Regarding the loss functions, the combinations with the CE executed faster than the others on the same optimizer, with the slowest being the BCE function, while the fastest optimizer was the SGD, followed by the RMSprop, with AdamW being the slowest of them.

	Optimizer			
Loss	SGD	RMSprop	Adam	AdamW
CE	46min59s	49min06s	50min40s	50min59s
BCE	52min14s	53min29s	54min17s	53min44s
BCELogits	47min26s	50min47s	53min17s	53min30s

Table 6: Average execution time for each loss and optimizer pair.

Overall, the fastest pair was the CE with SGD, executing in 46 minutes and 59 seconds, followed by the BCELogits function with the same optimizer, taking 47 minutes and 26 seconds. The use of BCE and Adam took the longer, executing in 54 minutes and 17 seconds, with the combinations of Adam and AdamW optimizers with BCE and BCELogits all taking more than 53 minutes each. This mark also includes the BCE and RMSprop combination, that, regarding needing fewer epochs to train, took 53 minutes and 9 seconds on average to execute.

6 Conclusion

In this work, we proposed a comparative study of combinations of loss functions and optimizers for the parking space classification, selecting a set of these functions and algorithms from the literature that covered the most widely used ones. Through a wide set of experiments, analyzed from the points of view of accuracy and time, we reached the following conclusion for our research question:

RQ1: Which combination of loss function and optimizer, among the selected algorithms, yields the best results in the parking space classification? Considering both accuracy and number of training epochs, the combination of the BCE loss function with the SGD optimizer obtained the best results, at 98.2% accuracy and 4.$\bar{6}$ epochs. The use of CE and BCELogits functions with the same optimizer also was presented as a good alternative for the problem, achieving similar accuracies and performing faster than the previous one. Besides taking the fewer epochs to train, the BCE function with RMSprop optimizer combination achieved the worst accuracy overall, scoring more than 40% lower than all other pairs, not being considered as a suitable option on the presented scope.

The results presented demonstrate that the use of different loss functions and optimizers on machine learning algorithms can impact on the accuracy and performance of a classification task. This analysis can provide insights on how to improve these models to achieve better results. However, this study focused solely on the parking space classification using CNNs, considering the mentioned scope and model. Further research is encouraged to testify the effectiveness of these and other loss functions and optimizers combinations in similar tasks and scopes, as the results may vary between each specific problem, aiming to achieve more accurate and faster machine learning solutions.

Acknowledgments

This study was partially supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – Projeto 405511/2022-1 and Coordination for the Improvement of Higher Education Personnel (CAPES) - Program of Academic Excellence (PROEX).

References

[1] Vijay Paidi, Hasan Fleyeh, Johan Håkansson, and Roger Nyberg. Smart parking sensors, technologies and applications for open parking lots: A review. *IET Intelligent Transport Systems*, 12, 04 2018. doi: 10.1049/iet-its.2017.0406.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[3] Luan Marko Kujavski, Paulo Mateus Luza Alves, and Paulo Lisboa de Almeida. Proposta de modelos leves para classificação de vagas de estacionamentos em cidades inteligentes. In *Anais do Computer on the Beach*, volume 16, pages 219–222, 2025. doi: 10.14210/cotb.v16.p219-226.

[4] Paulo Luza Alves, André Hochuli, Luiz Eduardo de Oliveira, and Paulo Lisboa de Almeida. Optimizing parking space classification: Distilling ensembles into lightweight classifiers. In *2024 International Conference on Machine Learning and Applications (ICMLA)*, pages 1016–1020, 2024. doi: 10.1109/ICMLA61862.2024.00152.

[5] Narina Thakur, Eshanika Bhattacharjee, Rachna Jain, Biswaranjan Acharya, and Yu-Chen Hu. Deep learning-based parking occupancy detection framework using resnet and vgg-16. *Multimedia Tools and Applications*, 83(1):1941–1964, 2024.

[6] Shuo Zhang, Xin Chen, and Zixuan Wang. Bcfpl: binary classification convnet based fast parking space recognition with low resolution image. In *International Conference on Image, Signal Processing, and Pattern Recognition (ISPP 2024)*, volume 13180, pages 1442–1449. SPIE, 2024.

[7] Andre G. Hochuli, Alceu S. Britto, Paulo R. L. de Almeida, Williams B. S. Alves, and Fábio M. C. Cagni. Evaluation of different annotation strategies for deployment of parking spaces classification systems. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2022. doi: 10.1109/IJCNN55064.2022.9892783.

[8] Soumia Goumiri, Dalila Benboudjema, and Wojciech Pieczynski. One convolutional layer model for parking occupancy detection. In *2021 IEEE International Smart Cities Conference (ISC2)*, pages 1–5, 2021. doi: 10.1109/ISC253183.2021.9562946.

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

[10] Paulo R.L. de Almeida, Luiz S. Oliveira, Alceu S. Britto, Eunelson J. Silva, and Alessandro L. Koerich. Pklot – a robust dataset for parking lot classification. *Expert Systems with Applications*, 42(11):4937–4949, 2015. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2015.02.009. URL https://www.sciencedirect.com/science/article/pii/S0957417415001086.

[11] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. 2019. URL https://arxiv.org/abs/1905.02244.

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.