IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 8 April 2025, accepted 21 May 2025, date of publication 2 June 2025, date of current version 9 June 2025.
Digital Object Identifier 10.1109/ACCESS.2025.3575093

APPLIED RESEARCH

Detection of Southern Hemisphere Constellations Using Convolutional Neural Networks

VLADIMIR RIFFO, SEBASTIÁN FLORES, EDUARDO CHUY-KAN, AND VÍCTOR ARIZA
Departamento de Ingeniería Informática y Ciencias de la Computación, Universidad de Atacama, Copiapó 1531772, Chile
Corresponding author: Sebastián Flores (sebastian.flores@uda.cl)

ABSTRACT Constellations allow the identification of most stars and celestial objects visible in the night sky at a glance, without the use of a telescope. However, because of the large number of constellations, this task can be overwhelming for astronomy beginners. To perform this identification, many rely on mobile applications that depend on Internet connectivity and the Global Positioning System. Unfortunately, these applications only provide estimates based on geolocation and do not guarantee an accurate visual representation. For this reason, this paper proposes the identification of constellations using Convolutional Neural Networks. The purpose of this research is to detect constellations with greater accuracy from photographs of any size, regardless of the availability of an Internet connection. For this, a convolutional neural network model called You Only Look Once was used. This neural network is widely known for its accuracy in object detection and is ideal for pattern recognition, such as constellations. In this work, different versions of this neural network were used to detect the 21 most representative constellations of the southern hemisphere. The results obtained reveal that all models exhibit outstanding performance, with high precision and recall values, resulting in F1-*scores* of 0.991 and higher.

INDEX TERMS Constellations, convolutional neural networks, detection, YOLO.

I. INTRODUCTION

Historically, constellations have played a crucial role in various practical applications, from ancient navigation and popular culture to modern astronomy. The identification of constellations has traditionally been carried out manually, a task that, while enriching the understanding of the cosmos, is laborious and prone to errors. In response to this need, in recent years, automatic methods based on computer vision techniques have emerged with satisfactory results [1], [2], [3].

This type of detection involves identifying the stars that form a constellation in an image of the sky, a challenging task given the diversity of shapes and sizes of these groupings, as well as the presence of various types of noise in the images [4].

In this proposal, we develop an automatic method for constellation detection based on Convolutional Neural Networks (CNN). It is based on a deep CNN whose approach learns to recognize the distinctive features of the stars that make up a constellation.

Sky images are subject to a series of challenges. Different types of noise, such as background noise, shot noise, and camera noise, can affect the quality of images, complicating the task of automatic identification, whose accuracy depends on the sharpness and clarity of these images. In this way, the signal-to-noise ratio (SNR) is an important metric for astronomical observations, as it represents the amount of information in the data compared to the noise [5].

Background noise is caused by the Earth's atmosphere scattering the light from the stars, while other factors such as camera noise are caused by the device's mechanism that can produce image faults. In particular, background noise can make it difficult to identify the individual stars that form a constellation, while shot noise can cause false positives.

For this reason, related works have emerged that seek to solve the problem automatically and without relying on the human factor. However, automatic methods, despite their efficiency and speed, are not without difficulties.

The associate editor coordinating the review of this manuscript and approving it for publication was Krishna Kant Singh.

Although they overcome human limitations in speed and efficiency, they can be more susceptible to errors, especially in low-quality images [6].

The method proposed in this work utilizes a neural network known as You Only Look Once (YOLO), whose approach has not been previously used in the field of constellation identification, positioning it as a modern alternative compared to previous approaches. In addition, the simplicity of this method results in reduced training time, making it more efficient in terms of computational resources.

II. RELATED WORK

In the work by Doan et al. [7], the use of a convolutional neural network called ''FiF-Net'' for constellation-based classification is proposed, where the method for training the network consists of using radioelectric features of the brightness of the stars, which requires a series of parameters and training that demand a greater number of cores to process the enormous amount of data.

Deep visual features, specifically the points within a constellation, can be fully learned by simultaneously implementing grouped and asymmetric convolutional layers in a single structure. This approach avoids the problem of having to train the network with repetitive data that could lead to input data redundancy and overfitting.

The particularity of this work is that it uses a convolutional neural network known as ''Resnet'' as its base, and by changing a series of parameters, good results were achieved, reaching an accuracy rate of 86%.

Traditional algorithms for recognizing star patterns are based on extracting the characteristics of stars as a vector [8], [9], [10] to compare them with a database. However, many of these algorithms can be sensitive to noise and lack of information.

Jiang et al.'s work [11] presents an algorithm that uses a hierarchical convolutional neural network to recognize images of spider webs. This algorithm is more robust than traditional methods, as it can identify constellations using a spider web pattern even with noise or little information. To do this, a star is taken as a reference and then extends to the nearest points forming the spider web.

Once the shape has been defined, the next step is to compare it with the database that stores various star shapes. This comparison process aims to identify similarities, thus allowing the analyzed shape to be classified into a specific type.

The database that stores this model can also present possible variations and errors that could arise during data collection and storage.

It is essential to recognize that, due to various conditions and factors, the stored information may contain inherent variations. These variations can result from factors such as the quality of the observations, atmospheric conditions, or even errors in the data recording process.

Other authors, such as Galkin and Makarenko [12], also employ constellation detection models using ResNet, and obtained results similar to the previous study, with a main F1-*score* reaching 0.927. Both works show a clear inclination towards the use of ''ResNet'' in the training process of convolutional neural networks. However, neither study details the specific implementation of the ResNet architecture nor the particularities of the training strategy.

Additionally, it would be relevant to investigate whether there are variations in the datasets used, as this could influence the comparison and generalization of the results between both works. This analysis would provide a more comprehensive and accurate view of the similarities and differences between the implementations of ResNet in the constellation detection models proposed by Galkin and other authors [7], [12].

In Lindsey's work [13], a method for identifying constellations using a camera known as a ''star tracker'' onboard spacecraft is also described. Instead of using conventional methods, such as extensive databases, an approach based on convolutional neural networks is proposed. This method compares features of stars near the unknown star, being more robust to positional noise and requiring a smaller database, especially for narrow fields of view.

To achieve this, a convolutional neural network was implemented where the input data is the brightness of a star, which is then compared with the nearest star and the subsequent one. The angle formed between these three stars is what is recognized as a constellation and is compared with a database to find similarities.

Some features shared by these works with our proposal are shown in Table 1.

TABLE 1. Similarities between related works and the current proposal.

Feature	Description
Use of convolutional neural networks	Both the related works and the current proposal employ convolutional neural networks to address the problem of constellation identification in the field of astronomy.
Focus on constellation identification	All the works, including the proposal, aim to identify constellations using astronomical data and image processing techniques based on neural networks.
Use of large datasets	Each work required a significant dataset to train the convolutional neural networks, which implies considerable time and resources for the preparation and processing of these data.
High success rates	It is observed that all works achieved success rates above 85%, indicating the effectiveness of convolutional neural networks in identifying constellations in the astronomical context.

III. METHOD

The proposed method[1] addresses key aspects, such as the acquisition and preparation of specific datasets for

[1]The code developed and used in this work is available on GitHub: (https://github.com/VictorAriz/Datasets-Costelaciones-Emisferio-Sur-datasets.git)

FIGURE 1. Constellation crux or southern cross (left), constellation orion (right).

training and evaluating YOLO models in versions 8-X, 10-X and 11-X. For training, the number of images to be used as a basis for achieving detection must first be taken into account. For this work, 21 constellations from the southern hemisphere were selected, which were chosen for their prominence in the sky and their distribution across different spatial zones.

A. DATA SELECTION

The diversity of the selected constellations covers a wide range of stellar patterns and densities, allowing for the evaluation of the detection models' capabilities in diverse scenarios. Additionally, the visibility of these constellations at different times of the year was considered to ensure an equitable temporal distribution of the training data.

The 21 most representative constellations of the southern hemisphere were considered: *Aquila, Ara, Apus, Aries, Canis Maior, Columba, Crux, Delphinus, Equuleus, Grus, Lacerta, Lyra, Musca, Octans, Orion, Phoenix, Piscis Austrinus, Reticulum, Scorpius, Triangulum Australe and Tucana*.

Here is the translation while keeping the LaTeX tags:

The inclusion of constellations such as *Crux, Scorpio*, and *Orion*, which are widely recognized and have distinctive star patterns, poses interesting challenges for the model due to the presence of multiple bright stars and complex shapes, as shown in Figure 1. On the other hand, the incorporation of less prominent constellations, such as *Musca* or *Reticulum*, allows evaluating the model's ability to detect more subtle and dispersed star patterns.

B. DATA ACQUISITION AND PREPARATION

The images used come from open-access sources and the free software *Stellarium*, which provides images based on the selected geographic location and the year set by the user.

A total of 1063 images of the 21 constellations were collected, sourced from both internet images and the Stellarium software. Figure 2 presents an overview where each constellation is identified with its corresponding name and shown in comparison with its graphical representation.

C. DATA AUGMENTATION

The authors of YOLO [14] recommend having a sufficiently large and diverse training dataset that represents all possible variations the model might encounter during real-world application. For this reason, data augmentation was applied, which is the process of generating new training samples by applying various transformations to the existing images in the dataset.

Transformations such as rotation, translation, distortion (*shear*), and scaling were applied. Additionally, random variations in noise, brightness, and contrast were introduced. Figure 3 shows different types of rotations and the application of filters.

Once the data augmentation process was completed, a total of 16,533 images were obtained, mixing the mentioned variations and filters.[2] Manual labeling of each image was then carried out.

Once the manual labeling is completed, the training dataset is enriched with information about the location and class of objects in each image. This labeled dataset will then be used to train the model, allowing it to learn to associate specific visual patterns with the corresponding classes.

TABLE 2. Distribution of images in the dataset by class and set.

Clase	Train	Val	Test
Crux (0)	700	200	101
Aquila (1)	495	141	72
Ara (2)	543	155	78
Apus (3)	827	236	119
Aries (4)	593	169	86
Canis Maior (5)	422	120	62
Columba (6)	413	118	59
Delphinus (7)	540	154	78
Equuleus (8)	427	122	61
Grus (9)	702	200	102
Lacerta (10)	577	165	83
Lyra (11)	428	122	62
Musca (12)	434	124	62
Octans (13)	414	118	60
Orion (14)	514	147	74
Phoenix (15)	685	195	99
Piscis Austrinus (16)	472	135	68
Reticulum (17)	394	112	58
Scorpius (18)	661	189	95
Triangulum Australe (19)	674	192	98
Tucana (20)	649	185	94

IV. TRAINING THE MODELS

The dataset was divided into training and validation sets. The distribution of images among classes and partitions can be seen in Table 2.

[2]The image database constructed and used for this work is available on GitHub: (https://github.com/VictorAriz/Datasets-Costelaciones-Emisferio-Sur-datasets.git)

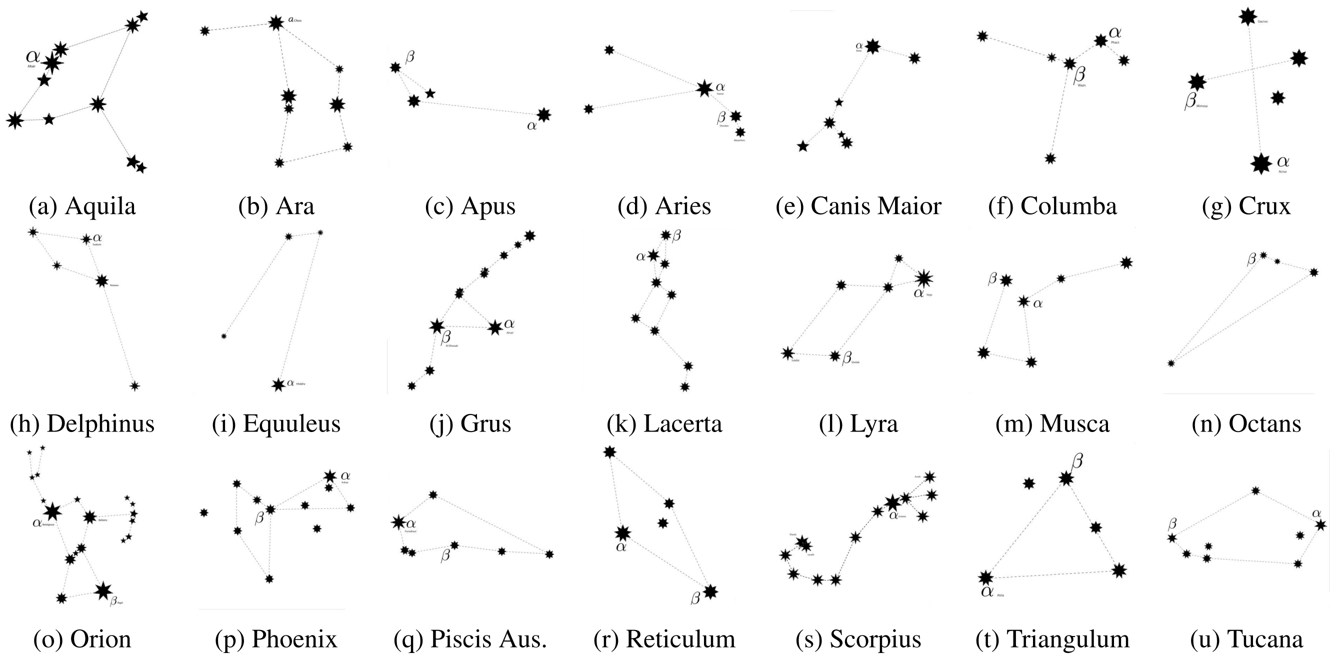


FIGURE 2. Constellations of the southern hemisphere used.

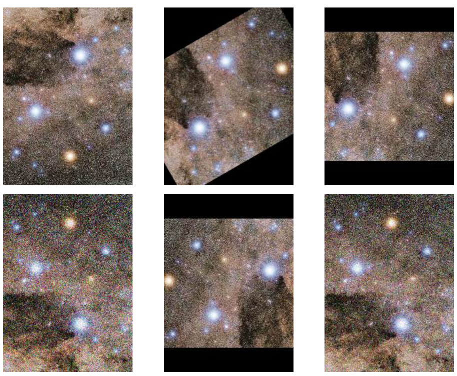


FIGURE 3. Data augmentation process.

The training of the YOLOv8-X, YOLOv10-X, and YOLOv11-X models was carried out following a structured process. Below, the configuration and procedure used are described.

A. TRAINING CONFIGURATION

To ensure a fair comparison between the YOLOv8-X, YOLOv10-X, and YOLOv11-X models, consistent hyperparameters were used during the training process. The learning rate was set at 0.01, while the batch size was 16 samples per iteration. The total number of epochs was set to 100, and the input images were resized to 640 pixels. The selected optimizer was Stochastic Gradient Descent (SGD), due to its robustness and effectiveness in optimization tasks.

The training environment was configured to maximize computational performance and compatibility with the tools used. The hardware employed included an NVIDIA RTX 4080 GPU with 16 GB of dedicated memory, an Intel Core i9-14900KF processor, and 32 GB of RAM. To ensure reproducibility of the experiment, Python 3.9 was used along with PyTorch and CUDA 11.7, ensuring compatibility with the selected GPU. The Ultralytics library was used for managing and configuring the YOLO models, while the environment was virtualized using Conda to isolate dependencies and facilitate the installation of specific packages.

B. TRAINING PROCESS

The training process was divided into two main phases: environment setup and training execution. For the environment setup, a dedicated Conda environment was created, and the necessary dependencies were installed, including PyTorch and the Ultralytics library. Additionally, the correct detection of the NVIDIA RTX 4080 GPU and the configuration of CUDA were verified to ensure optimal hardware utilization.

During the training execution, the following steps were implemented:

i) The specific pre-trained weights for each model were loaded, corresponding to `YOLOv8-X`, `YOLOv10-X`, and `YOLOv11-X`.

ii) The configuration file `dataset.yaml` was used to define the classes and partitions of the dataset, ensuring a proper structure for training.

iii) The results, including metrics and generated weights, were automatically recorded, facilitating progress monitoring and model evaluation.

iv) During training, loss and accuracy were constantly monitored to ensure optimal model fitting and detect potential issues such as overfitting.

C. CONSTELLATION DETECTION

The test images were used by all YOLO versions to establish a meaningful comparison baseline, thereby ensuring a fair and accurate evaluation of each version's performance under the same conditions. By using the same set of test images, differences in detection capacity and accuracy between the versions can be identified.

Concluding the detection process, this generated outputs that showed the constellations identified by each model, along with the coordinates of the detections. These outputs were evaluated and compared with each of the YOLO versions.

V. RESULTS AND ANALYSIS

We performed the evaluation of various performance metrics, both at a global level and by class, thereby having detailed comparisons between the models.

A. GLOBAL PERFORMANCE OF THE MODELS

The evaluation of the computational performance of the YOLOv8-X, YOLOv10-X, and YOLOv11-X models was conducted in terms of total training time, average inference time per image, number of parameters, and computational capacity measured in GFLOPs. These values are presented in Table 3.

TABLE 3. Comparison of the global performance of the models.

Model	Training Time (h)	Inference Time (ms)	Parameters	GFLOPs
YOLOv8-X	19.381	7.3	68,143,791	257.5
YOLOv10-X	75.81	8.0	31,624,526	170.0
YOLOv11-X	30.178	5.9	56,851,279	194.5

The results show that the YOLOv8-X model stands out for being the fastest in terms of training, while YOLOv11-X offers the shortest average inference time, making it highly efficient for real-time applications. In terms of parameters and GFLOPs, YOLOv8-X is the most computationally demanding, while YOLOv10-X represents a more lightweight solution.

B. PERFORMANCE METRICS

The performance of our method is measured using the quality evaluation PASCAL criteria [15], where a detection is considered valid if the normalized area of overlap a_o, between the bounding box of a detection BB_{dt} and the bounding box of the ground truth BB_{gt} is greater than a threshold θ. The normalized area is defined as follows (see Figure 4):

$$a_o = \frac{area(BB_{dt} \cap BB_{gt})}{area(BB_{dt} \cup BB_{gt})}. \quad (1)$$

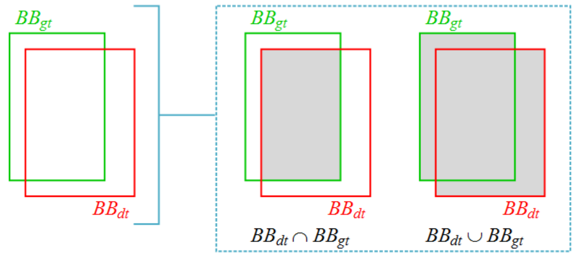


FIGURE 4. Evaluation criteria for comparing bounding boxes. Interpreting the area of overlap criteria. The normalized area a_o is given by the ratio of the intersection to the union areas.

TABLE 4. Comparison of metrics among the YOLOv8-X, YOLOv10-X, and YOLOv11-X models.

Metric	YOLOv8-X	YOLOv10-X	YOLOv11-X	Winner
Precision (Pr)	0.991	**0.993**	0.987	YOLOv10-X
Recall (Re)	0.992	0.990	**0.995**	YOLOv11-X
F1-score	**0.992**	**0.992**	0.991	Tie
Training Time (h)	**19.381**	75.81	30.178	YOLOv8-X
Inference Time (ms)	7.3	8.0	**5.9**	YOLOv11-X
Number of Parameters	68,143,791	**31,624,526**	56,851,279	YOLOv10-X
GFLOPs	257.5	**170.0**	194.5	YOLOv10-X

where, $BB_{dt} \cap BB_{gt}$ is the intersection of the detection window and the ground truth, and $BB_{dt} \cup BB_{gt}$ their union. In case $a_o > \theta$, the detection is considered true positive, or otherwise false positive (usually, the overlapping threshold θ is set to 0.5). In our work, we measured the *Precision*, *Recall* and F1-*score* defined by:

- *Precision*: Indicates the probability that a positive prediction is actually correct.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

- *Recall*: Measures the likelihood that the network correctly identifies all positive images.

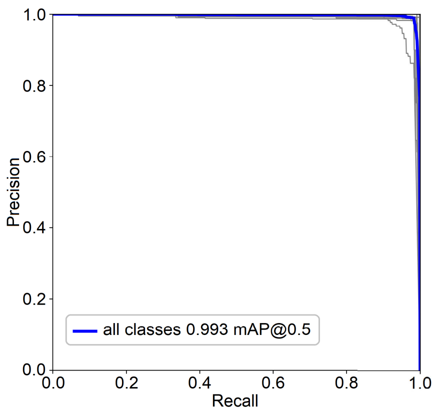
$$Recall = \frac{TP}{TP + FN} \quad (3)$$

- $F1 - score$: It combines the precision and recall by computing the harmonic mean between their two values.

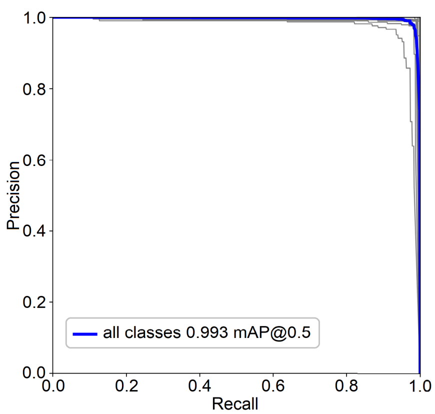
$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (4)$$

where, *TP* is the number of true positives, *FP* is the number of false positives, *TN* is the number of true negative and *FN* is the number of false negative. A perfect classifier achieves $Precision = 1$ and $Recall = 1$, all objects are classified with no false alarm. In this case, $F1\text{-}score = 1$.

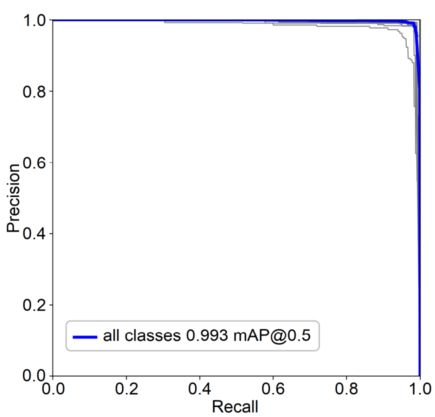
As seen in Table 4, all models exhibit outstanding performance, with very high *Precision* (Pr) and *Recall* (Re) values, resulting in F1-*scores* of 0.991 or higher. The YOLOv10-X model achieves the highest precision value (Pr = 0.993), while YOLOv8-X reaches an F1-*score* of 0.992,



(a) YOLOv8-X

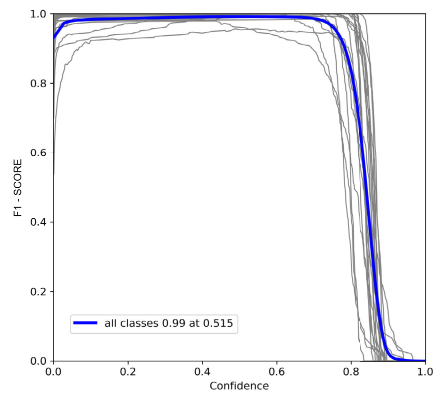


(b) YOLOv10-X

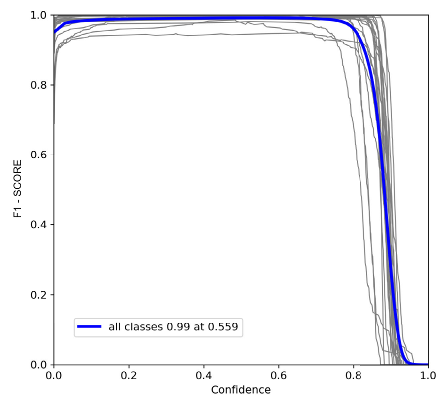


(c) YOLOv11-X

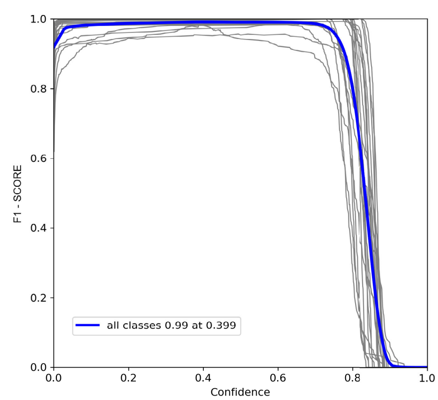
FIGURE 5. PR curves of the YOLOv8-X, YOLOv10-X, and YOLOv11-X models.



(a) YOLOv8-X



(b) YOLOv10-X

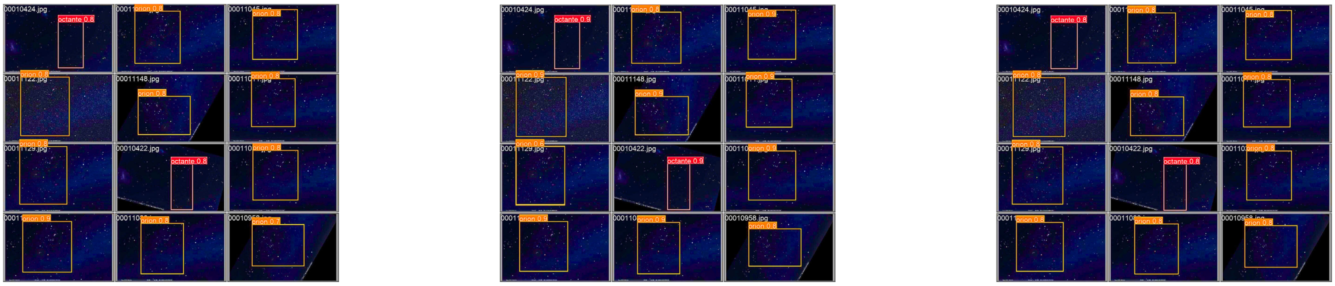


(c) YOLOv11-X

FIGURE 6. F1 curves of the YOLOv8-X, YOLOv10-X, and YOLOv11-X models.

standing out for its balance between precision and recall. On the other hand, YOLOv11-X is distinguished by its high recall value (Re = 0.995), indicating a superior ability to detect objects in images. YOLOv8-X, although with slightly lower precision than YOLOv10-X, matches its F1-*score* of 0.992.

The results highlight the robustness of the three models in the task of constellation detection, although small differences in metrics suggest that certain models might be more suitable depending on the specific use case. The performance of the evaluated models is graphically presented through the PR and F1-score curves in Figures 5(a), 5(b), 5(c), 6(a), 6(b), 6(c).



(a) YOLOv8-X (b) YOLOv10-X (c) YOLOv11-X

FIGURE 7. Visualization of predictions from validation batch 2 for the YOLOv8-X, YOLOv10-X, and YOLOv11-X models.

TABLE 5. Performance of YOLOv8-X, YOLOv10-X, and YOLOv11-X models. Includes Precision, Recall, and F1-*score*.

Ranking	Class	YOLOv8-X			YOLOv10-X			YOLOv11-X		
		Pr	Re	F1-score	Pr	Re	F1-score	Pr	Re	F1-score
Best	Aquila	0.997	1.000	0.999	0.997	1.000	0.999	0.996	1.000	0.998
	Ara	0.997	1.000	0.999	1.000	1.000	1.000	0.997	1.000	0.999
	Grus	0.998	1.000	0.999	1.000	0.983	0.992	0.997	1.000	0.999
Worst	Apus	0.981	0.983	0.982	0.979	0.982	0.980	0.980	0.983	0.982
	Triangulum	0.979	0.984	0.981	0.988	0.990	0.989	0.982	0.990	0.986
	Tucana	0.962	0.950	0.956	0.957	0.935	0.946	0.947	0.959	0.953

C. PERFORMANCE BY CLASS

The analysis of class performance allows for evaluating the effectiveness of the models for specific constellations. Precision, Recall, and F1-*score* metrics were calculated for each class. The results of our method are shown in Table 5, where it can be observed that the constellations *Aquila, Ara, and Grus* stand out with high Precision, Recall, and F1-score values in all three models, reaching values close to 0.999 or higher in all metrics. This indicates a high capacity for detection and precise localization of these classes.

On the contrary, the classes that showed relatively lower performance, especially in F1-*score*, correspond to *Apus* and *Triangulum Australe* with values around 0.98, and *Tucana* with 0.95. This behavior may be related to specific characteristics of these constellations or limitations of the dataset used.

VI. CONCLUSION

In this research, we demonstrate that the method implemented for the automatic detection of constellations in the southern hemisphere using the CNN models YOLOv8-X, YOLOv10-X, and YOLOv11-X achieved remarkable performance in precision, recall, and F1-*score* metrics.

The results show that YOLOv10-X stands out as the best overall model due to its higher precision ($Pr = 0.993$), fewer parameters, and lower GFLOP, making it computationally efficient. On the other hand, YOLOv11-X is the best model for real-time inference, thanks to its low inference time (5.9 ms) and higher recall ($Re = 0.995$). Finally, YOLOv8-X stands out as the fastest training model, with significantly lower training time, making it ideal for scenarios where training time is critical.

REFERENCES

[1] M. A. Abdel-Rahim, M. A. El-Sherif, and M. A. Al-Khodary, "An automatic method for constellation detection in low-quality images based on convolutional neural networks," *Int. J. Comput. Appl.*, vol. 11, no. 3, pp. 147–155, 2020.

[2] S. Nadamoto, N. Mori, and M. Okada, "Constellation identification method using point set data," *Artif. Life Robot.*, vol. 28, no. 2, pp. 361–366, May 2023.

[3] A. Jamal, U. Abid, and M. A. Ismail, "Intelligent constellation detection using pattern recognition," *Phys. Scripta*, vol. 99, no. 7, Jul. 2024, Art. no. 076011.

[4] A. Popowicz, A. R. Kurek, T. Blachowicz, V. Orlov, and B. Smolka, "On the efficiency of techniques for the reduction of impulsive noise in astronomical images," *Monthly Notices Roy. Astronomical Soc.*, vol. 463, no. 2, pp. 2172–2189, Dec. 2016, doi: 10.1093/mnras/stw1983.

[5] A. Vojtekova, M. Lieu, I. Valtchanov, B. Altieri, L. Old, Q. Chen, and F. Hroch, "Learning to denoise astronomical images with U-nets," *Monthly Notices Roy. Astronomical Soc.*, vol. 503, no. 3, pp. 3204–3215, Oct. 2021, doi: 10.1093/mnras/staa3567.

[6] R. M. Hussien, K. Q. Al-Jubouri, M. A. Gburi, A. G. Hussein Qahtan, and A. H. Duaa Jaafar, "Computer vision and image processing the challenges and opportunities for new technologies approach: A paper review," *J. Phys., Conf. Ser.*, vol. 1973, no. 1, Aug. 2021, Art. no. 012002.

[7] V.-S. Doan, T. Huynh-The, C.-H. Hua, Q.-V. Pham, and D.-S. Kim, "Learning constellation map with deep CNN for accurate modulation recognition," in *Proc. IEEE GLOBECOM*, Taipei, Taiwan, Sep. 2020.

[8] L. Luo, L. Xu, and H. Zhang, "An autonomous star identification algorithm based on one-dimensional vector pattern for star sensors," *Sensors*, vol. 15, no. 7, pp. 16412–16429, Jul. 2015. [Online]. Available: https://www.mdpi.com/1424-8220/15/7/16412

[9] H. Liu, X. Wei, J. Li, and G. Wang, "A star identification algorithm based on recommended radial pattern," *IEEE Sensors J.*, vol. 22, no. 8, pp. 8030–8040, Apr. 2022.

[10] J. Liao, X. Wei, A. Niu, Y. Zhang, I. S. Kweon, and C. Qi, "Redundant-coded masked grid pattern for full-sky star identification," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 60, no. 4, pp. 4343–4355, Aug. 2024.

[11] J. Jiang, L. Liu, and G. Zhang, "Star identification based on spider-Web image and hierarchical CNN," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 4, pp. 3055–3062, Aug. 2020.

[12] V. A. Galkin and A. V. Makarenko, "Neural network approach to recognition of visible constellations by sky photo image," *J. Phys., Conf. Ser.*, vol. 1864, no. 1, May 2021, Art. no. 012014.

[13] C. Lindsey, T. Lindblad, and A. Eide, "Method for star identification using neural networks," Ph.D. dissertation, Dept. Phys., Royal Inst. Technol., Stockholm, Sweden, Apr. 1997, doi: 10.1117/12.271545.

[14] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jan. 2018, pp. 7794–7802.

[15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

VLADIMIR RIFFO received the B.Sc.Eng. degree in electronic engineering from the Universidad de Antofagasta, Antofagasta, Chile, in 1998, and the M.Eng. degree and Ph.D. degree in engineering sciences from the Pontificia Universidad Católica de Chile (PUC), Santiago, Chile, in 2011 and 2016, respectively. He is currently a Full Professor with the Department of Computer Engineering and Computer Science, Universidad de Atacama (UDA), Copiapó, Chile, where he has been Director in two periods, from 2006 to 2008 and from 2017 to 2018. In addition, he is the Research Director of UDA. He is an Academic Collaborator with the XCV Research Laboratory, PUC. His current research interests include pattern recognition, object detection, automatic visual inspection, computer vision using multiple views and X-ray testing, and the ways to combine those approaches. He received a scholarship for his Ph.D. studies from the Comisión Nacional de Investigación Científica y Tecnológica (CONICYT) and was a recipient of the Ron Halmshaw Award, in 2017 and 2012, the John Green Award from the British Institute for Non-Destructive Testing, in 2013, which was established for the best papers published in Insight Journal on *Industrial Radiography*.

SEBASTIÁN FLORES received the bachelor's degree in computing and informatics from the University of Atacama, Copiapó, Chile, in 2013, and the master's degree in computer science from the Universidad de Atacama (UDA), in 2015. He was a Civil Engineer in computing and informatics from the University of Atacama, in 2013. From 2012 to 2013, he was a Consultant for a national IT services company and from 2013 to 2022, he was a Research Analyst with the Vice-Rector's Office for Research and Postgraduate Studies, UDA. Since 2023, he has been an Instructor Professor with the Department of Computer Engineering and Computer Science, UDA. In his emerging research career, he has authored two journal articles and led three regional educational innovation projects. His research interests include cybersecurity culture, applications of artificial intelligence for object detection, computer management, and the general field of applied computing.

EDUARDO CHUY-KAN was born in Copiapó, Chile. He received the Civil Engineering degree in computer science from the University of Atacama, in 2024. He is currently a Software Developer and the Head of information technology. His professional interests include data science, web development, and process optimization. He actively applies technological solutions to enhance organizational efficiency and decision-making.

VÍCTOR ARIZA was born in Cereté, Córdoba, Colombia. He received the Professional degree in systems engineering from the Universidad del Sinú, Colombia, in 2023. He is currently pursuing the master's degree in computer engineering and computer science with the University of Atacama, Copiapó, Chile. In his second year, he is actively developing his thesis as part of the final stage of his graduate studies. He has participated in projects focused on developing artificial intelligence models in Montería, Colombia, and has been a Software Developer for a private company, gaining hands-on experience in applying intelligent technologies. His research interests include artificial intelligence, machine learning, computer vision, autonomic computing, precision livestock farming, and smart environments.