Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 19 No. 5 (2025) | OPEN ACCESS

https://doi.org/10.3991/ijim.v19i05.54265

PAPER

Prediction of Emotions in Human Beings, Using Emotional Intelligence and Data Mining

Arnulfo Alanis(✉), Bogart Yail Marquez, Guillermo Daniel Prieto, Ángeles Quezada, José Sergio Magdaleno-Palencia

Tecnológico Nacional de México, Campus Tijuana, México

alanis@tectijuana.edu.mx

ABSTRACT

Human beings go through several periods in life where it is crucial to learn how to adapt to their own needs—biological and physical. Added to these are social conditions, which may involve participation in society, education, care, and impact. This last aspect is a central topic that supports many other situations; hence, emotional intelligence (EI) is addressed. EI underpins the way individuals handle their emotions, helping them develop various skills, such as understanding and managing those emotions for effective interaction with others. Emotions will be examined by taking into account a dataset, and the classification and prediction will be based on the paradigm of artificial intelligence. Machine learning will be used, specifically convolutional neural networks (CNNs). This paper analyzes emotion prediction using two types of machine learning networks to identify emotions: CNNs and recurrent neural networks (RNNs), aiming to determine which of these two yields better predictive performance.

KEYWORDS

emotional intelligence, data mining

1 INTRODUCTION

The importance of considering the capacity to predict human emotions in different areas is a relevant field of study, due to its wide range of applications, such as health, education, and marketing (whether local or online). Emotional intelligence (EI) [1] is defined as having the capacity for several factors, for example, recognizing and understanding emotions. In addition, it identifies five fundamental components of EI: emotional self-awareness, emotional self-regulation, motivation, empathy, and social skills. Along with the vast amount of data that is generated, data mining techniques provide effective tools for analyzing large volumes of emotional data. Taking this into account is essential for examining relationships among human beings. When looking at the possibility of predicting emotions, it is mentioned that EI is a factor guiding the interpretation of emotions and enabling them to adapt to the paradigm of artificial intelligence to achieve a point of prediction—that is,

Alanis, A., Marquez, B.Y., Prieto, G.D., Quezada, Á., Magdaleno-Palencia, J.S. (2025). Prediction of Emotions in Human Beings, Using Emotional Intelligence and Data Mining. *International Journal of Interactive Mobile Technologies (iJIM)*, 19(5), pp. 159–169. https://doi.org/10.3991/ijim.v19i05.54265

Article submitted 2024-11-06. Revision uploaded 2025-01-10. Final acceptance 2025-01-16.

how an adequate level of emotional accuracy can be reached in specific contexts. Implementing data mining involves the application of advanced algorithms to visualize and analyze large amounts of information (data), thereby identifying meaningful patterns. For predicting emotions in particular, such patterns may be found in various datasets, for example: facial expressions, spoken language, and written text. In addition to the above, applying one or more data mining techniques is essential. Zhang et al. [2] mention that one of the deep neural networks (DNN) models is effective for identifying emotions from unstructured signals, such as facial expressions and tone of voice. According to [3], deep learning is a component of machine learning focused on finding algorithms that structure high-level abstractions in datasets, often involving numerous layers. Convolutional neural networks (CNNs) are considered among the top performers in pattern recognition. In recent years, CNNs have seen extensive application and development, and their significant impact has yielded very high results, as well as spurred new lines of research [4] [5] [6] [7] [8] [9] [10]. This trend has also enabled the creation of massive datasets [11]. Another type of network to consider for object or pattern recognition is recurrent neural networks (RNNs). These networks possess the ability to learn from problems considered to be complex. This type of network is made up of a set of hidden, distributed states that can store extensive information about the past in an efficient way [12]. These networks are dynamic, which means they may have feedback connections. A simple RNN features a neuron that receives inputs and generates an output, which then feeds back into the neuron itself, as shown in Figure 1.

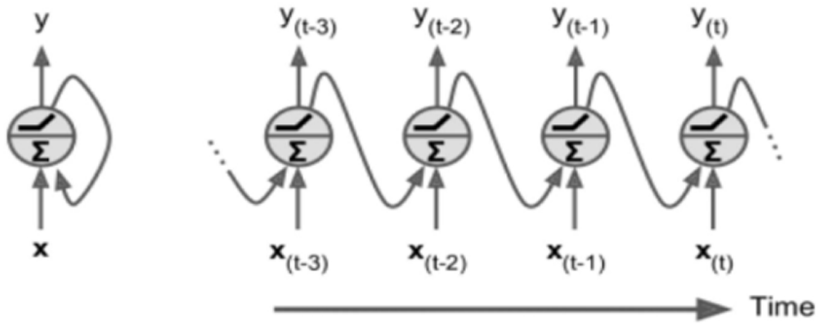

Fig. 1. Recurrent neuron (left), development over time (right) [12]

2 NETWORK STRUCTURE

The analysis of emotions is currently a topic of utmost relevance, forming a key area of study such as EI [1]. EI supports the way in which individuals manage their emotions and helps them develop various skills, including how to utilize and handle them for interaction among human beings.

Emotions will be analyzed by taking into account a dataset [13]. Classification and prediction will be carried out by comparing two machine learning methods: CNNs and recurrent neural networks.

Considering the convolution operation

$$Si(i,j) = (X * k)(i,j) = \sum_{m}\sum_{m} x(i+m, j+n) \cdot k(m,n) + b \quad (1)$$

Where:

- X is the input image (or the feature map from the previous layer).
- k is the filter (kernel) applied to the input.

- b is the bias.
- $S(i, j)$ is the resulting value in the feature map at position (i, j).

This operation is carried out by sliding the filter k across the input image X, multiplying their corresponding values, and summing them for each position.

3 ALGORITHMS

The code organization is influenced by common practices in modular programming, such as using classes to encapsulate model logic and structures aligned with a "component-based" web development approach (although applied to Python code for data processing and modeling). This is reflected in the clear separation of tasks: data preparation, model definition, training, evaluation, result export, and real-time recognition. Additionally, specialized libraries for deep learning (PyTorch) [14] and visualization (matplotlib) [15] are employed, along with tools for reporting (FPDF) [16] and data handling (JSON, argparse) [17], [18].

4 DEFINITION OF THE MODELS ARCHITECTURES

Two classes are defined: EmotionCNN and EmotionRNN, both inheriting from nn.Module in PyTorch.

EmotionCNN: Defines a CNN with three convolutional layers (from 1 to 32 filters, then from 32 to 64, and so on), followed by pooling layers (MaxPool2d) and a fully connected layer. ReLU activation functions and dropout are used for regularization. The size of the resulting vector after the convolutions is calculated dynamically in _calculate_flatten_size(). The CNN is ideal for image-type data, since it learns spatial features and local patterns in facial images.

EmotionRNN: Defines a simplified bidimensional LSTM. Each image is considered as a sequence of 48 steps (each step is a vector of 48 pixels, assuming one line of the image). The RNN/LSTM is suitable for sequential data, although here its use is more exploratory, aiming to compare its performance with the CNN. The network consists of two LSTM layers and one final dense layer. This approach is less common for images but serves as a basis for comparison.

4.1 Training and evaluation

There will be three functions for training, evaluating, and managing the model:

Train model (...): Receives the model (either CNN or RNN), the training DataLoader, a test DataLoader, and parameters such as the number of epochs and the learning rate. It uses CrossEntropyLoss and the Adam optimizer. In each epoch, it iterates over all the batches of the training set, calculates the loss, and performs backpropagation to update the weights. At the end, the average loss is reported.

Evaluate model (...): Performed once the model has been trained; this function evaluates it on the test set. It calculates predictions, class probabilities, and obtains metrics such as multiclass AUC and Accuracy. It also generates the ROC curve, saving it as an image in the imagenes_roc_entrenamientos folder.

Load or train model (...): Checks whether a previously trained model with the specified parameters (epochs, LR, dataset_tag) exists. If so, it loads it; if not, it trains a new model and saves it. This saves time by reusing previous results. This process is defined by the implementation of: train, evaluate CNN and RNN, as shown in Algorithms 1 and 2.

Algorithm for training and evaluating the CNN model

Algorithm 1: Train and Evaluate the CNN

```
# Train and Evaluate CNN
cnn_model = EmotionCNN()
cnn_model, dataset_tag = load or train model (cnn_model, train_loader, test_loader,
"emotion models_cnn", epochs=args.cnn_epochs, learning_rate=args.cnn_lr,
use_min=args.use_min)
cnn_auc, cnn_acc, cnn_roc_img = evaluate model (cnn_model, test_loader,
model_name=f"cnn{dataset_tag}")
```

Algorithm for training and evaluating the RNN model

Algorithm 2: Train and Evaluate the RNN

```
# Train and evaluate RNN
rnn_model = EmotionRNN()
rnn_model, _ = load or train model (rnn_model, train_loader, test_loader,
"emotion models_rnn", epochs=args.rnn_epochs, learning_rate=args.rnn_lr,
use_min=args.use_min)
rnn_auc, rnn_acc, rnn_roc_img = evaluate model (rnn_model, test_loader,
model_name=f"rnn{dataset_tag}")
```

5 DATASET

The dataset used is prepared in the data preparation(...) function, which establishes the image transformation pipeline: images are converted to grayscale, resized to 48×48, normalized, and converted to tensors. Next, the training and test sets are loaded using datasets.ImageFolder. If the use_min mode is activated, class balancing is performed by taking the minimum number of images across all classes, thus preventing bias toward a majority class. This creates a balanced subset (ETTM, "Equal to the Minimum"), reducing the risk of overfitting a dominant class. If that option is not enabled, all available images are used (FULL mode). Finally, DataLoader objects are created for both the training and test sets, facilitating batching and efficient iteration over the data during training.

The first step involves choosing the computing device (CUDA/CPU): the process begins by detecting if a GPU (CUDA) is available. In this way, the best device is automatically selected for training and evaluation. The second step is creating directories: before starting the training process, two directories are ensured to exist: modelos (where trained weights are stored) and imagenes_roc_entrenamientos (for the ROC plots).

The dataset used is FER2013 (Facial Expression Recognition 2013 Dataset) [13]. It requires data preparation via preparar_datos(...), which defines the image transformation pipeline: converting to grayscale, resizing to 48×48, normalization, and conversion to tensors. The training and test sets are then loaded through datasets. ImageFolder. When use_min mode is active, class balancing is performed by selecting the minimum number of images among all classes, preventing bias toward any majority class. This results in a balanced subset (ETTM, "Equal to the Minimum"), reducing the risk of overfitting a dominant class. If this option is not active, all available images are used (FULL mode). Finally, the DataLoader objects for the training and test sets are created, enabling efficient batching and iteration over the data during training.

6 MODEL COMPARISON AND REPORTING

To compare the CNN and RNN models and determine under which conditions one model may outperform the other, multiple conditions and processes are analyzed. Specifically:

- Compare model(...): Compares the AUC obtained by the CNN and RNN, selects the best model, and displays a summary.
- Save internal(...): Each training attempt (set of parameters and results) is stored as a JSON object, added to the report_data.json file. This consolidates a history of runs.
- Generarte report_pdf(...): From the JSON containing all previous reports, a PDF is generated with the summary of each attempt: metrics, parameters, the best model, and ROC images are included. This facilitates the review and sharing of results, resulting in a coherent document that aggregates multiple runs.

7 RESULTS

For each model's structure and architecture, different values were considered to carry out training and measure the accuracy, such as the number of epochs, the number of data items in the dataset, as well as the type of data set mode (ETTM or FULL). The behavior of each model evaluated with the ROC curve, to identify the best model, is shown in Figures 2–7.

Below are some of the results from various scenarios:

Dataset type: ETTM

Scenario 1:

CNN -> Epochs: 20 – LR: 0.001
RNN -> Epochs: 20 – LR: 0.001
Images per Class (Minimum): 436
AUC CNN: 0.78 | AUC RNN: 0.65
Accuracy CNN: 0.42 | Accuracy RNN: 0.28

Result

Best model: CNN

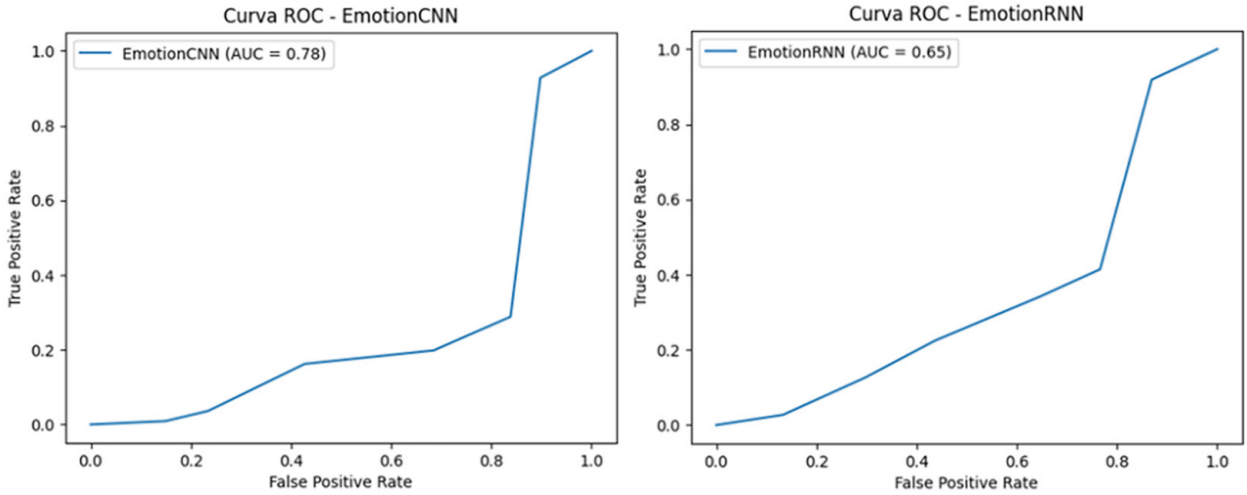

Fig. 2. ROC curve: CNN and RNN with 20 epochs and with the data set type: ETTM

Scenario 2:

CNN -> Epochs: 40 – LR: 0.001
RNN -> Epochs: 40 – LR: 0.001
Images per Class (Minimum): 436
AUC CNN: 0.76 | AUC RNN: 0.65
Accuracy CNN: 0.40 | Accuracy RNN: 0.27

Result

Best model: CNN

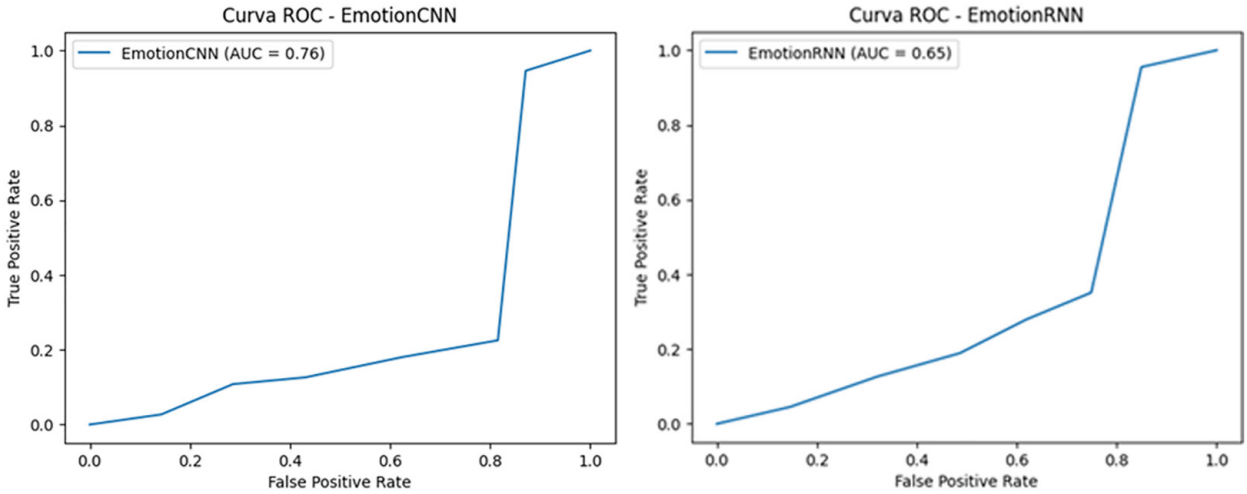

Fig. 3. ROC curve: CNN and RNN with 40 epochs and with the data set type: ETTM

Scenario 3:

CNN -> Epochs: 60 – LR: 0.001
RNN -> Epochs: 60 – LR: 0.001
Images per Class (Minimum): 436
AUC CNN: 0.75 | AUC RNN: 0.66
Accuracy CNN: 0.38 | Accuracy RNN: 0.28

Result

Best model: CNN

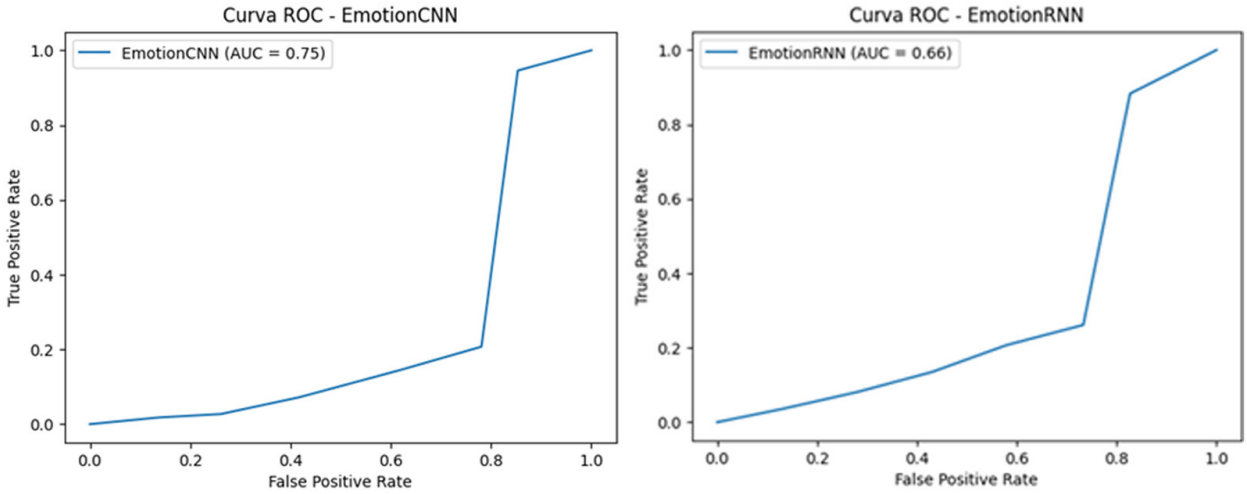

Fig. 4. ROC curve: CNN and RNN with 60 epochs and with the data set type: ETTM

Dataset type: FULL

Scenario 4:

CNN -> Epochs: 20 – LR: 0.001
RNN -> Epochs: 20 – LR: 0.001
Images per Class (Minimum): 436
AUC CNN: 0.87 | AUC RNN: 0.79
Accuracy CNN: 0.57 | Accuracy RNN: 0.47

Result

Best model: CNN

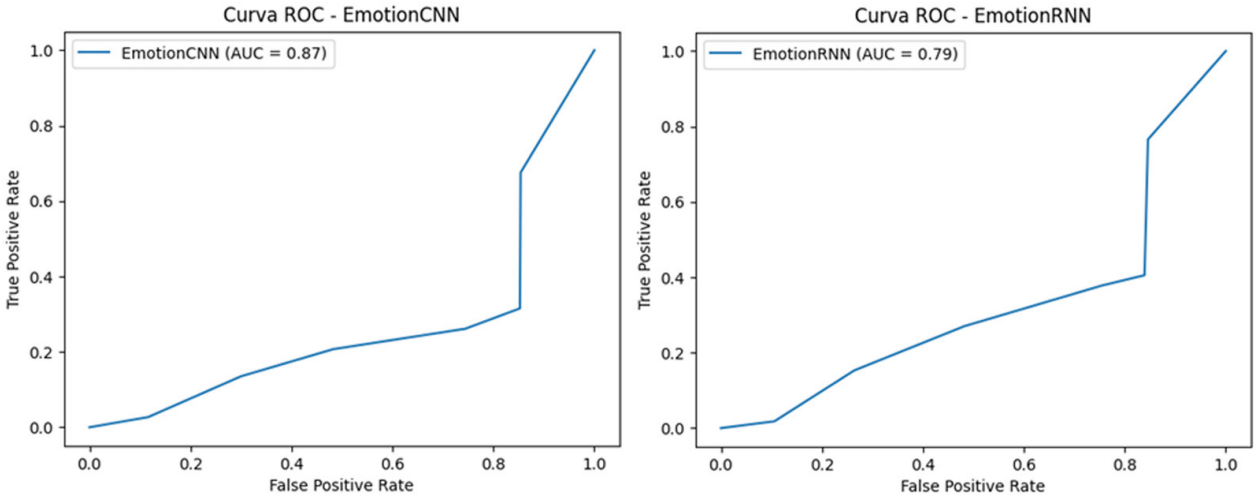

Fig. 5. ROC curve: CNN and RNN with 20 epochs and with the data set type: FULL

Scenario 5:

CNN -> Epochs: 40 – LR: 0.001
RNN -> Epochs: 40 – LR: 0.001
Images per Class (Minimum): 436
AUC CNN: 0.85 | AUC RNN: 0.79
Accuracy CNN: 0.56 | Accuracy RNN: 0.47

Result

Best model: CNN

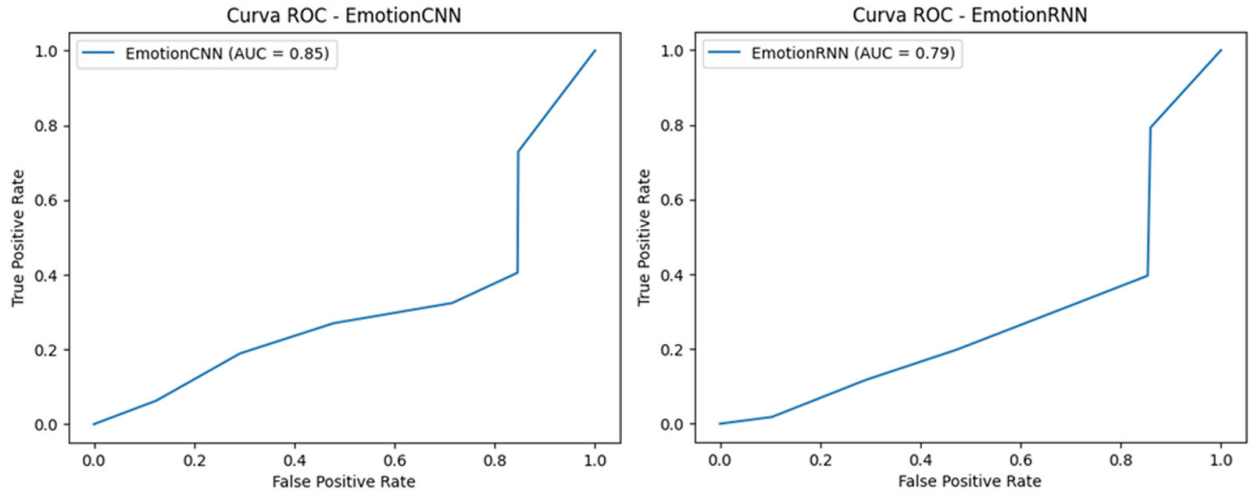

Fig. 6. ROC curve: CNN and RNN with 40 epochs and with the data set type: FULL

Scenario 6:

CNN -> Epochs: 60 – LR: 0.001
RNN -> Epochs: 60 – LR: 0.001
Images per Class (Minimum): 436
AUC CNN: 0.85 | AUC RNN: 0.78
Accuracy CNN: 0.56 | Accuracy RNN: 0.47

Result

Best model: CNN

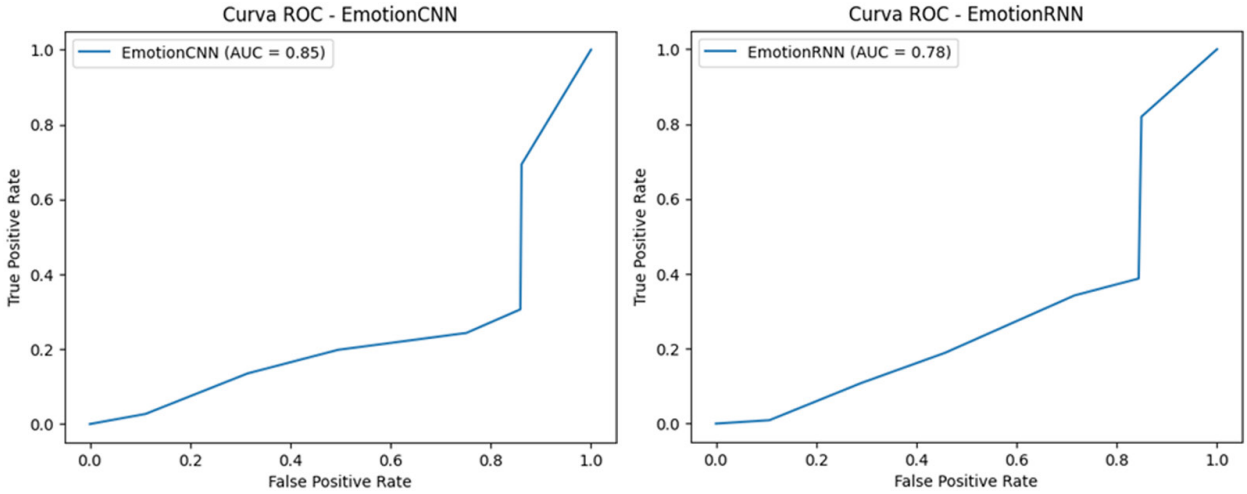

Fig. 7. ROC curve: CNN and RNN with 60 epochs and with the data set type: FULL

7.1 Confusion matrix

To organize the results of both models, an analysis through the confusion matrix was performed, as shown in Figures 8 and 9. The confusion matrix allows visualizing the number of correct and incorrect predictions for each class.

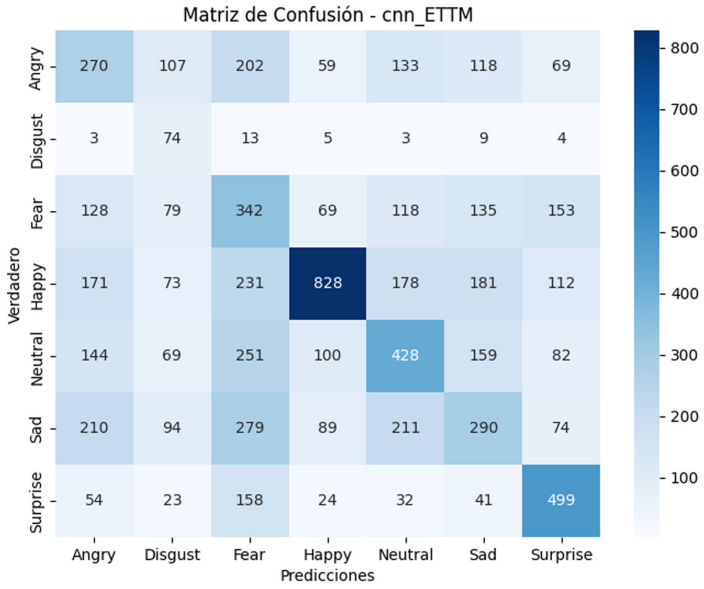

Fig. 8. CNN_ETTM confusion matrix

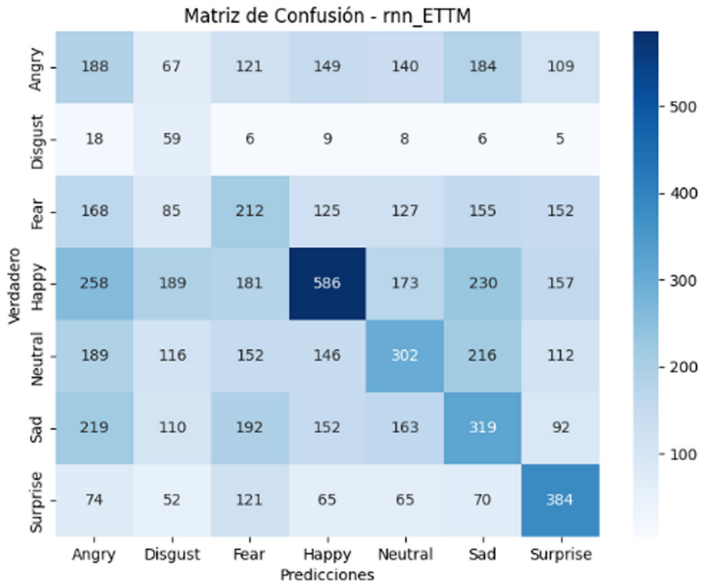

Fig. 9. RNN_ETTM confusion matrix

7.2 Model comparison

This section describes how the results of both models were organized. Several training sessions were conducted, considering different numbers of epochs and images per class for both models, using both ETTM and FULL datasets, as shown in Tables 1 and 2.

Table 1. Dataset ETTM with CNN and RNN

Dataset	CNN Epochs	CNN LR	RNN Epochs	RNN LR	Imágenes/ Clase	CNN AUC	CNN ACC	RNN AUC	RNN ACC	Best Model
_ETTM	20	0.001	20	0.001	436	0.7771	0.4245	0.6531	0.2843	CNN
_ETTM	120	0.001	120	0.001	436	0.75	0.3936	0.6569	0.2868	CNN
_ETTM	200	0.001	200	0.001	436	0.7328	0.3636	0.6479	0.2665	CNN
_ETTM	220	0.001	220	0.001	436	0.7423	0.3707	0.6523	0.2824	CNN
_ETTM	320	0.001	320	0.001	436	0.7391	0.3656	0.6519	0.2756	CNN
_ETTM	420	0.001	420	0.001	436	0.7485	0.3862	0.6513	0.2802	CNN
_ETTM	520	0.001	520	0.001	436	0.7416	0.3781	0.6458	0.264	CNN
_ETTM	620	0.001	620	0.001	436	0.7505	0.3965	0.6522	0.2722	CNN
_ETTM	720	0.001	720	0.001	436	0.7502	0.3867	0.6514	0.2742	CNN
_ETTM	780	0.001	780	0.001	436	0.7405	0.3721	0.6635	0.2916	CNN

Table 2. Dataset FULL with CNN and RNN

Dataset	CNN Epochs	CNN LR	RNN Epochs	RNN LR	Imágenes/ Clase	CNN AUC	CNN ACC	RNN AUC	RNN ACC	Best Model
_FULL	20	0.001	20	0.001	sin límite	0.8652	0.5747	0.7929	0.4657	CNN
_FULL	80	0.001	80	0.001	sin límite	0.8554	0.5681	0.7893	0.4777	CNN
_FULL	100	0.001	100	0.001	sin límite	0.8505	0.5605	0.7829	0.4661	CNN

(Continued)

Table 2. Dataset FULL with CNN and RNN *(Continued)*

Dataset	CNN Epochs	CNN LR	RNN Epochs	RNN LR	Imágenes/ Clase	CNN AUC	CNN ACC	RNN AUC	RNN ACC	Best Model
_FULL	180	0.001	180	0.001	sin límite	0.849	0.5593	0.7792	0.4733	CNN
_FULL	200	0.001	200	0.001	sin límite	0.8512	0.5655	0.7874	0.4766	CNN
_FULL	260	0.001	260	0.001	sin límite	0.8479	0.5578	0.7861	0.4795	CNN
_FULL	340	0.001	340	0.001	sin límite	0.8446	0.5485	0.7878	0.4811	CNN
_FULL	400	0.001	400	0.001	sin límite	0.8445	0.5564	0.7935	0.4886	CNN
_FULL	420	0.001	420	0.001	sin límite	0.838	0.5482	0.7857	0.4703	CNN
_FULL	500	0.001	500	0.001	sin límite	0.839	0.5579	0.787	0.4727	CNN

8 CONCLUSIONS ON THE RESULTS

Based on the previous reports and comparative tables (selecting 20 representative results), several conclusions can be drawn.

- **Best Model:** The **CNN** model consistently outperformed the **RNN** in terms of AUC and accuracy. This is not surprising, given that CNNs are more suitable for spatial data (images), whereas RNNs are better suited for sequential data.
- **Effect of balancing (ETTM vs. FULL)**
  - The dataset in **FULL** mode tends to provide higher metrics (an AUC around 0.85 and ACC > 0.56 in some cases) compared to **ETTM** mode. This suggests that having more images per class (even if unbalanced) could help the CNN capture variations more effectively.
  - On the other hand, **ETTM** mode avoids excessive bias toward the most represented classes, yet the maximum performance does not surpass what's achieved with FULL.
- **Increasing the number of epochs:** Raising the number of epochs does not guarantee a linear improvement. In many cases, performance stabilizes or fluctuates, indicating there may be an optimal training point (for example, between 20 and 100 epochs) beyond which no major improvements are observed.
- **AUC–accuracy relationship:** A high AUC does not always translate into higher accuracy. This indicates that while the model is good at **ranking** the classes, it might not be selecting the optimal classification threshold. Fine-tuning may be necessary to maximize accuracy.

Predicting human emotions through emotional intelligence and data mining is a promising field that offers significant applications in various domains. Despite the progress achieved so far, technical and ethical challenges remain to be addressed. Future research should focus on improving model accuracy, exploring new sources of emotional data, and developing ethical guidelines for implementation.

9 REFERENCES

[1] D. Goleman, *Emotional Intelligence: Why It Can Matter More Than IQ*. New York, NY: Bantam Books, 1995.

[2] Z. Zhang and X. Zhang, "Emotion recognition using deep learning models: A survey," *Journal of AI Research*, vol. 18, no. 1, pp. 45–60, 2023.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.
[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278–2324, 1998. https://doi.org/10.1109/5.726791
[5] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: The difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, S. C. Kremer and J. F. Kolen, Eds., IEEE Press, 2001.
[6] G. E. Hinton, "To recognize shapes, first learn to generate images," *Prog. Brain Res.*, vol. 165, pp. 535–547, 2007. https://doi.org/10.1016/S0079-6123(06)65034-6
[7] Y. Bengio, *Learning Deep Architectures for AI*. Now Publishers Inc., 2009. https://doi.org/10.1561/2200000006
[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012.
[9] I. Goodfellow *et al.*, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020. https://doi.org/10.1145/3422622
[10] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV),* Santiago, Chile, 2015, pp. 1026–1034. https://doi.org/10.1109/ICCV.2015.123
[11] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, pp. 211–252, 2015. https://doi.org/10.1007/s11263-015-0816-y
[12] A. Géron, *Hands-On Machine Learning with Scikit-Learn & TensorFlow*. Sebastopol, CA: O'Reilly Media, 2017. Available in: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/
[13] O. Collins Oguine, K. J. Oguine, H. I. Bisallah, and D. Ofuani, "Hybrid facial expression recognition (FER2013) model for real-time emotion classification and prediction," *BOHR Int. J. Internet Things, Artif. Intell. Machine Learn. (BIJIAM)*, vol. 1, no. 1, pp. 56–64, 2022. https://doi.org/10.54646/bijiam.011
[14] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019. https://doi.org/10.48550/arXiv.1912.01703
[15] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. https://doi.org/10.1109/MCSE.2007.55
[16] F. Derbaix, "FPDF: Free PDF Creation Library," 2024. Available in: https://www.fpdf.org/
[17] P. S. Foundation, "JSON—A lightweight data-interchange format," 2024. Available in: https://www.json.org
[18] P. S. Foundation, "Argparse—Command-line option and argument parser," 2024. Available in: https://docs.python.org/3/library/argparse.html

10 AUTHORS

Arnulfo Alanis is with the Tecnológico Nacional de México, Campus Tijuana, México (E-mail: alanis@tectijuana.edu.mx).

Bogart Yail Marquez is with the Tecnológico Nacional de México, Campus Tijuana, México.

Guillermo Daniel Prieto is with the Tecnológico Nacional de México, Campus Tijuana, México.

Ángeles Quezada is with the Tecnológico Nacional de México, Campus Tijuana, México.

José Sergio Magdaleno-Palencia is with the Tecnológico Nacional de México, Campus Tijuana, México.