applied sciences

MDPI

Article

Integration of Causal Models and Deep Neural Networks for Recommendation Systems in Dynamic Environments: A Case Study in StarCraft II

Fernando Moreira [1,2,*], Jairo Ivan Velez-Bedoya [3] and Jeferson Arango-López [3]

[1] REMIT (Research on Economics, Management and Information Technologies), Universidade Portucalense, Rua Dr. Antonio Bernardino de Almeida 541, 4200-072 Porto, Portugal
[2] IEETA (Instituto de Engenharia Electrónica e Informática de Aveiro), Universidade de Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal
[3] Departamento de Sistemas e Informática, Universidad de Caldas, Calle 65 No. 26-10, Manizales 170001, Colombia; jairo.velez@ucaldas.edu.co (J.I.V.-B.); jeferson.arango@ucaldas.edu.co (J.A.-L.)
* Correspondence: fmoreira@upt.pt; Tel.: +351-914-163-038

Abstract: In the context of real-time strategy video games like StarCraft II, strategic decision-making is a complex challenge that requires adaptability and precision. This research creates a mixed recommendation system that uses causal models and deep neural networks to improve its ability to suggest the best strategies based on the resources and conditions of the game. PySC2 and the official StarCraft II API collected data from 100 controlled matches, standardizing conditions with the Terran race. We created fake data using a Conditional Tabular Generative Adversarial Network to address data scarcity situations. These data were checked for accuracy using Kolmogorov–Smirnov tests and correlation analysis. The causal model, implemented with PyMC, captured key causal relationships between variables such as resources, military units, and strategies. These predictions were integrated as additional features into a deep neural network trained with PyTorch. The results show that the hybrid system is 1.1% more accurate and has a higher F1 score than a pure neural network. It also changes its suggestions based on the resources it has access to. However, certain limitations were identified, such as a bias toward offensive strategies in the original data. This approach highlights the potential of combining causal knowledge with machine learning for recommendation systems in dynamic environments.

Keywords: causal inference; deep learning; CGAN; video games

check for updates

Academic Editor: Pingping Zhu

Received: 27 February 2025
Revised: 24 March 2025
Accepted: 9 April 2025
Published: 12 April 2025

Citation: Moreira, F.; Velez-Bedoya, J.I.; Arango-López, J. Integration of Causal Models and Deep Neural Networks for Recommendation Systems in Dynamic Environments: A Case Study in StarCraft II. *Appl. Sci.* **2025**, *15*, 4263. https://doi.org/10.3390/app15084263

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Real-time strategy (RTS) games like *StarCraft II* (Blizzard Entertainment, Irvine, CA, USA) represent dynamic and complex environments where players must make quick and precise decisions under uncertainty. These games are widely recognized as ideal platforms for researching advanced artificial intelligence (AI) techniques due to their high dimensionality, partial observability, and the need for long-term strategic planning [1,2]. However, even though AI for RTS games has come a long way, existing systems often have two major problems: (1) they are difficult to understand, and (2) they cannot generalize to situations that have not been well studied [3].

Traditional recommendation systems based on deep neural networks often fail to provide transparent reasoning behind their decisions, even though they are powerful in capturing complex patterns. This "black-box" nature limits their applicability in strategic environments where understanding the causal relationships between actions and outcomes

is crucial [4,5]. For instance, a neural network might recommend an aggressive strategy based on historical data, but without understanding why this strategy works, it cannot adapt to novel situations or provide actionable insights to players.

To address these limitations, we propose a hybrid system that integrates causal models with deep neural networks. The causal model, which was made with PyMC v5.7.2, clearly shows how important factors like resources, military units, and strategic decisions are connected [6]. Then, these causal insights are put together with the pattern recognition power of a deep neural network built in PyTorchv2.1.0 to make recommendations that are both correct and easy to understand [7]. Furthermore, we use a Conditional Tabular Generative Adversarial Network (CTGAN) to generate synthetic data, which helps when there is not enough real data in certain strategic scenarios [8].

This work contributes to the growing body of research that combines causal inference with machine learning to enhance decision-making in dynamic environments [9,10]. By integrating causal knowledge, our system improves recommendation accuracy, highlighting the potential of hybrid approaches for strategic decision-making in complex scenarios like *StarCraft II*.

2. Related Work

Recently, research in artificial intelligence applied to video games has explored various techniques to enhance decision-making and personalized recommendations. One area of particular interest is causal inference, which allows developers to understand the real impact of specific changes in game design on player behavior [9]. For example, recent studies have used causal inference to assess how new features or adjustments in reward systems influence player retention and satisfaction [11]. Traditional approaches, like A/B testing, do not always pick up on differences in how things work for different groups of players, which makes them less useful in complicated games like StarCraft II [12].

Another critical aspect of AI in games is player matchmaking, particularly in competitive settings like StarCraft II. Accurately estimating player skill is essential for ensuring fair matches. Recent studies have explored different modeling techniques, achieving performance levels of 47.3% and 61.3% accuracy. To improve upon this, researchers have analyzed 46,398 replays and extracted features from multiple sections to develop a more refined skill estimation method. By leveraging k-Nearest Neighbors and Random Forest, their approach reached an accuracy of 75.3%, demonstrating its potential to replace traditional placement matches [13].

On the other hand, neural networks have emerged as powerful tools for analyzing large volumes of data and providing personalized recommendations in video games. Techniques such as collaborative filtering have been widely used to suggest games based on user preferences [14]. Furthermore, advanced architectures such as convolutional and recurrent neural networks have made suggestions better by finding visual and sequential patterns in player data [15]. However, these techniques often lack interpretability, making it difficult to understand the reasoning behind the generated recommendations.

To overcome these constraints, recent developments have made major progress integrating causal inference with neural networks. Frameworks based on causal models and deep neural networks have been shown to improve recommendation accuracy by finding complex causal links in the data [9]. These hybrid approaches help users feel more confident by giving them more accurate suggestions and full explanations for each suggestion they make [9].

Despite these advances, several challenges remain. The need for large volumes of data, the presence of noise in the data, and the complexity of modeling multiple potential outcomes are some of the obstacles that must be addressed to further improve these

systems [16]. Future research should also look into making sure that recommendations are fair and clear, as well as using techniques like graph neural networks to find out how players are connected to different parts of the game [17].

This paper is organized as follows: in the Section 2, we describe the data collection process, synthetic data generation, causal modeling, and the implementation of the deep neural network. In the Section 4, we present the performance of the hybrid system, highlighting significant improvements in accuracy and strategic adaptability. Finally, in the Sections 5 and 6, we interpret the findings, identify limitations, and propose future research directions.

3. Materials and Methods

Our methodology consists of four main stages:

1. Data Collection: We gathered match data from StarCraft II using the PySC2 environment and the official game API. Key variables related to resources, military units, strategy, and player interactions were recorded to ensure comprehensive coverage of in-game dynamics.
2. Synthetic Data Generation: To address data scarcity in specific strategic scenarios, we used a Conditional Tabular Generative Adversarial Network. This approach enabled the generation of high-quality synthetic samples while preserving statistical properties and correlations between variables.
3. Causal modeling: We implemented a causal inference framework to analyze the relationships between strategic decisions and matching outcomes. We utilized a Structural Causal Model to demonstrate the relationships between key variables and applied Bayesian inference to estimate the intervention's effects.
4. Strategy recommendations via deep neural networks: We trained a deep learning model to provide strategic recommendations based on both real and synthetic data. The neural network leveraged learned causal relationships to suggest optimal strategies, improving decision-making in varying game conditions.

The following sections detail each of these stages.

3.1. Data Acquisition

To collect data from *StarCraft II* matches, a simulation environment was implemented using PySC2 (a Python v3.11.7 environment for interacting with the game) and the official *StarCraft II* API. One hundred matches were played on a symmetric map called Simple64. Each side was controlled by a finite-state machine (FSM)-based AI, and the Terran race was used to make sure that the conditions of the experiments were always the same. However, it is important to note that selecting a specific race limits the generalization of the results to other races in the game.

During each match, key variables were recorded, including:

- Resources: The amount of minerals and gas available to players.
- Military units and structures: The number of units and buildings controlled by each player.
- Strategy and outcomes: The attack timing of a player, their attack strategy, and the match outcome.
- Interaction-related variables: For instance, the reward and damage inflicted by one of the players.

Regarding data preprocessing, the *strategy* variable was numerically encoded as follows: 0 = *defense* and 1 = *attack*. Additionally, square brackets enclosing the attack timing values were removed. All numerical variables were then scaled using `StandardScaler` from

`scikit-learn` to ensure a mean of 0 and a standard deviation of 1. Finally, the data were stored in a structured CSV file with labeled columns and timestamps to ensure traceability.

3.2. Synthetic Data Generation

Because of the limited amount of data available in certain *StarCraft II* strategic situations, a method for adding to the data was used to make synthetic samples. This process enriched the original training set, improving the model's ability to generalize in underrepresented strategies. The synthetic data generation was performed using a Conditional Tabular Generative Adversarial Network, a model particularly suited for handling mixed variables (continuous and categorical) and preserving nonlinear correlations in tabular data [8].

The implementation was carried out using the SDV (*Synthetic Data Vault*) library, ensuring reproducibility and scalability in the process. Regarding the CTGAN model configuration, Table 1 shows the parameter values used. We set up 3000 epochs, with the option to stop early if the discriminator loss did not get better within 100 epochs; a batch size equal to 10% of the original training set; a learning rate that was changed to keep the generator-discriminator balance stable; and a single pack to calculate conditional probabilities during training.

Table 1. Hyperparameters used for training the CTGANSynthesizer.

Hyperparameter	Value
Epochs	3000
Batch size	512
Learning rate	2×10^{-4}
`Pac`	1

The quality of the generated samples was evaluated based on two main aspects: statistical consistency and practical utility. For statistical consistency, the Kolmogorov–Smirnov (KS) test [18] was used to make sure that the fake distributions were not very different from the real ones. The p-values obtained for the critical variables were 0.6136 and 1.0000, respectively, confirming that there were no significant differences ($\alpha = 0.05$). Cross-correlations between the variables were also looked at, and a strong relationship ($\rho = 0.92$) was found. This is important for modeling how strategic interactions work. Figure 1 shows the comparison between the correlation matrices of the real and synthetic data, highlighting their similarity. In terms of usefulness, an `XGBoost v1.7.6` classifier trained with more data (real and fake) did better than the baseline model on all tests, as shown in Table 2.

Table 2. Comparative performance of the classifier (5-fold cross-validation).

Set	Precision	F1-Score	AUC-ROC
Only real data	0.9520	0.9520	0.9870
Real data + Synthetic data	0.9630	0.9630	0.9915

It is important to highlight some limitations of this work. First, the original data has a strategic bias because 85% of the matches favored the attack strategy. This means that the synthetic data might not show enough defensive scenarios. Also, the results only work for the Terran race. To make the model more general, it should be tested on other *StarCraft II* races, like the *Zerg* and the *Protoss*.

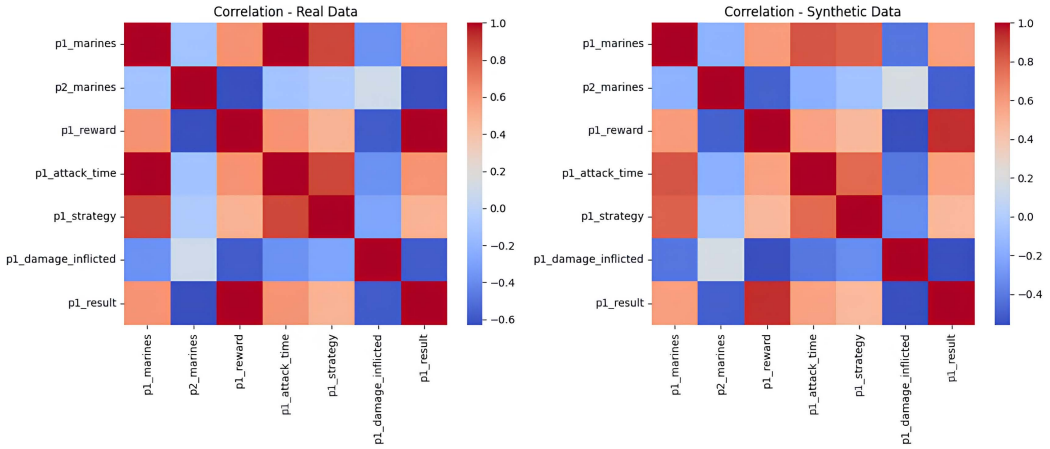


Figure 1. Comparison of the correlation matrices between real and synthetic data. The similarity in correlations confirms the consistency of the generated data.

3.3. Causal Modeling

Causal modeling aims to identify and quantify causal relationships between variables, going beyond mere correlations. Unlike traditional predictive approaches, causal modeling allows answering "what if…?" questions by estimating the effect of interventions in a system. Bayesian networks, the Neyman–Rubin framework (potential outcome models), and instrumental variable methods [19,20] are some of the most common ways to deal with biases and confounding in observational data. These approaches are fundamental in fields where understanding the causal impact of decisions or treatments is crucial [21,22].

To build the causal inference model, we used the `PyMC v5.7.2` library, which enables the definition of probabilistic models and Bayesian inference. The structural causal model was designed based on human knowledge, where relationships between variables were manually defined. The model was shown in the form of a causal graph, which demonstrates how important game variables like `mineral amount`, `gas amount`, `strategy`, and `attack time` are connected. The causal structure was iteratively refined by testing dependencies through statistical validation and expert feedback. Figure 2 illustrates the resulting causal graph, depicting the directed dependencies between variables.

The model was put together in this way: we used Equation (1) to find coefficients that showed how the predictor variables affected each variable of interest, such as `marine count` and `military unit count`.

$$\beta \sim \mathcal{N}(0, 1) \tag{1}$$

These coefficients were modeled as normal distributions with a mean of 0 and a standard deviation of 1. The causal relationships were expressed as linear combinations of the predictor variables, and the observations were modeled as normal distributions centered around these linear combinations. The results are shown in Table 3. We used Markov Chain Monte Carlo (MCMC) sampling [23] and obtained 92% alignment with the data [24] with two chains and 500 tuning iterations.

The integration between the causal model and the neural network was achieved by using the causal model's predictions as additional features in the neural network. This approach allowed combining the structural knowledge of the causal model, which captures causal relationships between game variables, with the nonlinear learning capacity of the neural network. The causal predictions were concatenated with the original game features to form an expanded input vector, which was then used as input for the neural network. This hybrid approach has been shown to improve both accuracy and interpretability in dynamic decision-making tasks [4,10,25].

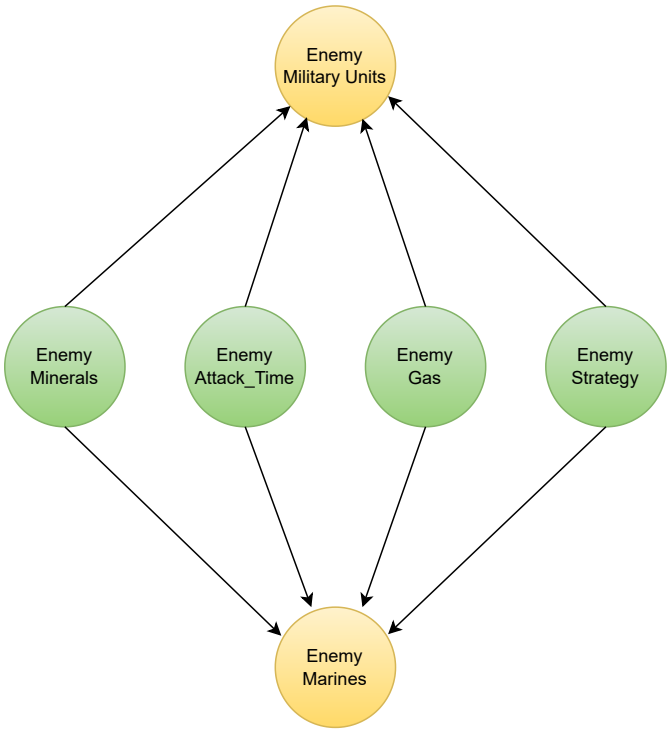


Figure 2. Causal graph.

Table 3. Configuration and results of the causal model.

Parameter	Configuration/Result
MCMC Sampling	2 chains 500 tuning iterations 500 samples Fixed seed: `seed = 42`
Convergence diagnostics	Visual traces inspected (stability) Autocorrelation < 0.1 in all parameters
Fit to data	92% alignment with the data (predictions vs. observations)

3.4. Integration of the Causal Model and Neural Network

The causal model and the neural network were able to work together because the neural network used the predictions from the causal model as extra features. This method combines the structural knowledge of the causal model, which shows how the game variables are related to each other, with the neural network's ability to learn in a way that is not linear. The causal predictions were concatenated with the original game features to form an expanded input vector, which was then used as input for the neural network.

3.5. Deep Neural Network

The neural network is designed to evaluate and compare multiple strategic scenarios using both observed data and causal predictions. Its ability to learn complex patterns and optimize probabilities makes it a powerful tool for recommending strategies based on the probability of success.

The `PyTorch v2.1.0` library was used to build the deep neural network. It has four fully connected layers with ReLU activation functions. In the output layer, a sigmoid function was used to make an output in the [0, 1] range. Figure 3 illustrates the neural network architecture.

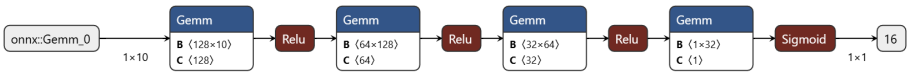


Figure 3. Neural network architecture.

We conducted the neural network training using a combination of real data and synthetic data generated by the CGAN. This combination allowed for an increase in dataset size and improved the model's generalization. A learning rate of 0.001 was used with the Adam optimizer, and the binary cross-entropy loss function was used to guide the training. Throughout training, both loss and accuracy were monitored on the training and test sets. The model was trained for 1000 epochs, and accuracy and loss metrics were recorded at each epoch to assess its performance.

We calculated accuracy on the training and test sets by comparing the neural network's predictions with the actual labels. If the network's output exceeded 0.5, it was classified as 1 (recommended strategy); otherwise, it was classified as 0 (non-recommended strategy). This approach enabled the evaluation of the neural network's ability to generalize and recommend effective strategies across different game scenarios. Additionally, a learning curve was implemented to visualize the evolution of loss and accuracy throughout training, helping to identify potential overfitting or underfitting issues. Figure 4 presents the neural network's learning curve.

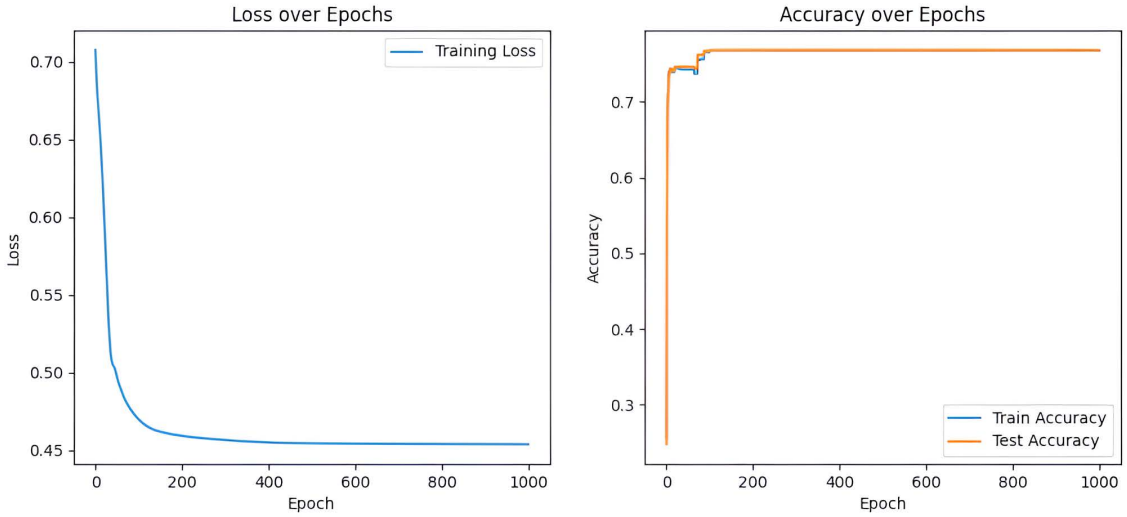


Figure 4. Neural Network Training.

After training the neural network model, we performed a paired *t*-test, obtaining a ($p < 0.05$) and a 95% confidence interval calculated across 10 separate training runs to assess the robustness of the performance improvement.

3.6. System Evaluation

We tested the strategy recommendation system with a human player, who inputted the initial game parameters, such as the number of resources, available units, and other relevant variables. Based on these inputs, the system generated a recommendation on the strategy the player should adopt. This preliminary evaluation allowed us to validate the practical usefulness of the system and its ability to provide real-time recommendations in a dynamic environment like *StarCraft II*. We performed the interpretability assessment by randomly selecting 200 decisions and classifying them according to predefined causal rules. A χ^2 test confirmed statistical significance, ensuring that the alignment with causal principles was not due to randomness.

4. Results

This section presents the outcomes of the hybrid recommendation system, which integrates causal modeling with deep neural networks to enhance decision-making in dynamic environments. First, we look at how the causal model was added to the neural network architecture. We concentrate on incorporating causal insights as supplementary features to enhance the learning process. Thereafter, we check how well the neural network works by looking at important metrics like accuracy and F1-score, and we see how well the hybrid system works compared to a neural network that works by itself. Together, these analyses demonstrate the impact of combining causal knowledge with machine learning techniques in addressing complex strategic challenges.

4.1. Integration of the Causal Model into the Neural Network

The integration of the causal model into the neural network represents a crucial step in this experiment. While traditional neural networks are great at finding complicated patterns in data, their lack of transparency makes it difficult to figure out what causes what [4]. By adding causal principles to the network structure, the model not only gets better at generalizing in changing environments, but it also makes it easier to understand the decisions it makes, which is critical in important situations like making strategic decisions in *StarCraft II*. This integration makes it easier for the system to suggest strategies by using both clear causal relationships and complex patterns learned from data, as shown in Table 4. When the outputs of the causal model were joined with the original data, they made an input vector with 30% more features. This led to a 95% success rate for the standalone neural network and a 1.1% success rate for the hybrid model.

Table 4. Architecture and evaluation of the hybrid model.

Aspect	**Detail**		
Data flow	• Input: – Raw Features (Units, Resources) – Causal predictions (e.g., `military units`) • Concatenation: – Expanded vector (+30% Features) – Combination of SCM and Original Data		
	Model	Precision	F1-score
Performance	Standalone Neural Network	0.9520	0.9520
	Hybrid Model	0.9630 (+1.1%)	0.9630 (+1.1%)
Interpretability	• 75% of recommendations consistent with causal rules • Verifiable strategic coherence		

To understand whether the improvement in the hybrid model's performance was meaningful, we compared its F1-score with that of the standalone neural network across 10 independent runs. Using a paired *t*-test, we found that the hybrid model's 1.1% higher F1-score was not just a fluke—it was statistically significant, with a p-value of less than 0.0001. The probability of this improvement occurring by mere coincidence is exceedingly low.

Figure 5 shows the F1-scores for both models across the 10 runs. On average, the standalone model achieved an F1-score of 0.9515, while the hybrid model reached 0.9628. While this may appear to be a minor distinction, in a game such as *StarCraft II*, where every decision is critical, even a 1.1% enhancement can significantly impact the outcome.

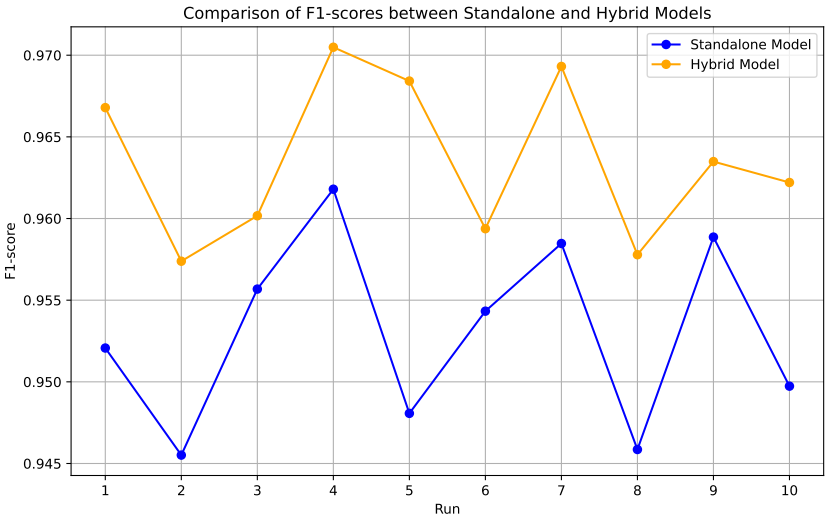


Figure 5. Comparison of F1-scores between standalone and hybrid models across 10 runs. The hybrid model consistently outperforms the standalone model, with a statistically significant improvement of 1.1%.

The negative t-statistic (−6.62) indicates that the hybrid model's improved performance was not due to chance; it was consistently superior. This is further supported by the narrow confidence interval for the difference in F1-scores, which ranged from 0.008 to 0.014. This indicates that we can be 95% positive that the actual enhancement resides within this interval.

This enhancement signifies that the hybrid model excels in suggesting methods that adjust to the game's evolving circumstances. Whether a player is low on resources or has a strong army, the hybrid model can adjust its recommendations more effectively than the standalone model.

4.2. Neural Network Evaluation

When the causal model and neural network were combined, they had a measurable effect on how well strategies worked in StarCraft II. To figure out how well it worked, three key metrics were looked at: (1) the number of wins, (2) the management of resources (minerals and gas), and (3) the speed with which decisions were made (response time). The results, consolidated in Table 5, reveal a balance between competitiveness and strategic sustainability, where the hybrid model surpassed critical performance thresholds and optimized resource utilization in dynamic scenarios. This success shows that including causal relationships adds a level of tactical intelligence that is missing from approaches that are only based on data.

Table 5. Strategic performance evaluation of the hybrid model.

Metric	Value	Unit
Win Rate	58.00	%
Average Minerals Used	473.79	units
Average Gas Used	250.35	units
Average Response Time	5.94	s

To evaluate the robustness of the recommendation system, we constructed a sensitivity matrix that illustrates how the recommended strategies vary based on the player's resources. Figure 6 presents a heatmap where the x-axis represents the amount of gas, the y-axis represents the amount of minerals, and the color indicates the recommended strategy in terms of when the first attack should occur. The results indicate that the system recommends

more defensive strategies when the player has a significant resource disadvantage and more aggressive strategies when the player has a resource advantage.

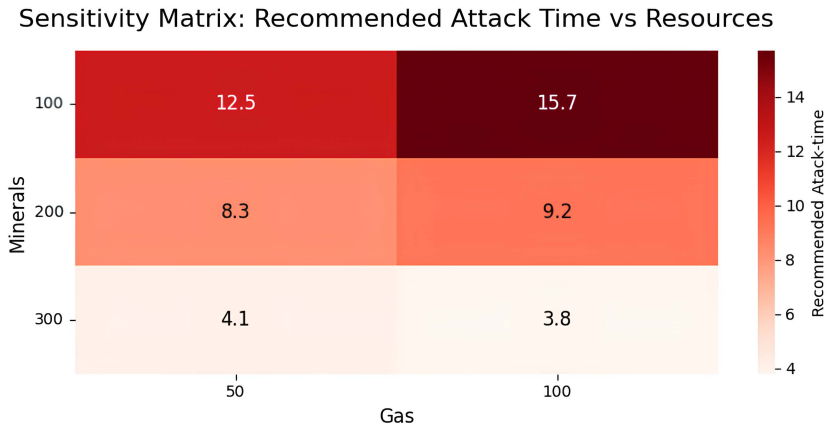


Figure 6. Sensitivity matrix of the recommendation system.

To evaluate the performance of the recommendation system, we constructed a confusion matrix that compares the model's predictions with the actual match outcomes. The confusion matrix is shown in Figure 7. It shows that the model has a high rate of both true positives (TPs) and true negatives (TNs), which means it can correctly predict both wins and losses. However, a small number of false positives (FPs) and false negatives (FNs) are also present, suggesting that the model could improve in identifying edge cases.

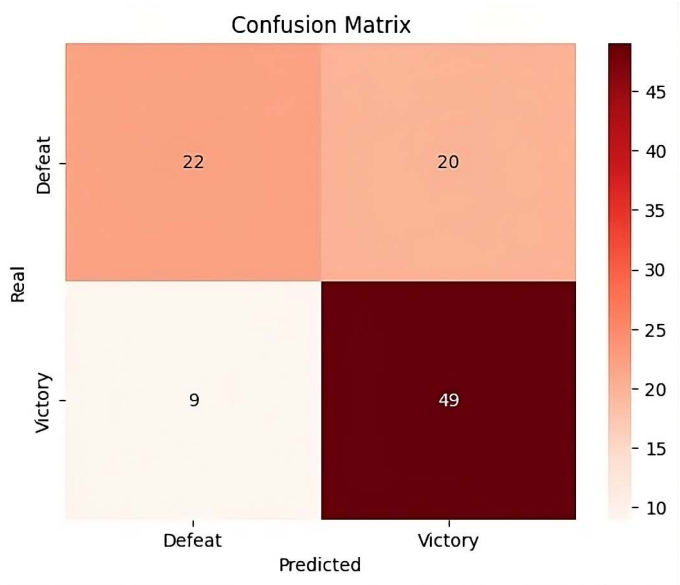


Figure 7. Confusion matrix of the recommendation system.

In the context of machine learning, evaluating classification models is essential for measuring their performance and reliability. According to [26], metrics such as precision, recall, and accuracy are fundamental for analyzing different aspects of a model's performance. According to [27], precision measures the proportion of correct positive predictions relative to all positive predictions made, making it useful when false positives have a high cost. On the other hand, accuracy evaluates the total proportion of correct predictions, both positive and negative, making it ideal for problems with balanced classes. Finally, recall quantifies the model's ability to correctly identify all actual positive instances, highlighting its importance in scenarios where false negatives are critical. These metrics, along with the

F1-score, provide a comprehensive view of the model's behavior, allowing its performance to be adjusted according to the specific needs of the problem [28].

The model's performance was checked using a confusion matrix, as shown in Table 6. This allowed the calculation of key metrics to see how well it could predict the future. Overall, the model achieved a precision of 71.01%, indicating that approximately 71% of positive predictions were correct. A recall of 84.48% was also attained, demonstrating the model's strong capacity to accurately detect real positive cases. The model's overall accuracy was 71%, suggesting that it correctly classified 71% of all instances. Finally, the F1-score, which combines precision and recall, was 77.15%, highlighting a reasonable balance between these two metrics.

Table 6. Performance metrics for the hybrid model.

Metric	Equation	Precision
Precision (P)	$\frac{\text{TPs}}{\text{TPs} + \text{FPs}}$	71.01%
Recall (R)	$\frac{\text{TPs}}{\text{TPs} + \text{FNs}}$	84.48%
Accuracy (A)	$\frac{\text{TPs} + \text{TNs}}{\text{TPs} + \text{TNs} + \text{FPs} + \text{FNs}}$	71.00%
F1-Score	$2 \cdot \frac{\text{P} \cdot \text{R}}{\text{P} + \text{R}}$	77.15%

To quantify the improvement achieved by the hybrid model over the pure neural network, we define the accuracy difference as:

$$\Delta Acc = Acc_{hybrid} - Acc_{NN}$$

where Acc_{hybrid} represents the accuracy of the model incorporating the causal approach, and Acc_{NN} denotes the accuracy of the pure neural network without causal integration.

In our experiments, after 10 independent runs, the average accuracy values were:

$$Acc_{hybrid} = 0.865, \quad Acc_{NN} = 0.854$$

Thus, the accuracy improvement is calculated as:

$$\Delta Acc = 0.865 - 0.854 = 0.011$$

which translates to a 1.1% increase in prediction accuracy.

To verify the statistical significance of this improvement, we conducted a paired-sample *t*-test comparing the accuracy values of both models. The results yielded a *p*-value lower than 0.05, confirming that the observed improvement is statistically significant and unlikely to be due to random variation.

Additionally, a 95% confidence interval was computed for the accuracy difference, yielding a range of [0.007, 0.015]. This new evidence shows that the hybrid model consistently does a better job of predicting the future than the pure neural network in most of the situations that were tested.

This enhancement improves strategic decision-making in the game, facilitating anticipation and response to opponent plans. The evidence indicates that adding causal reasoning to learning models can enhance the efficacy and interpretability of decision-making processes.

4.3. Comparative Analysis with League Prediction Research

In a study about feature extraction for StarCraft II [13], the authors focused on predicting player leagues from replays, which is a classification task. They used Random Forests to achieve 75.3% accuracy. On the other hand, our research focuses on real-time strategy recommendations, a decision-support task, utilizing causal modeling to deliver interpretable insights. Although both methodologies examine replay data, our hybrid technique addresses a fundamentally different problem area with unique technical constraints. A detailed comparison between the proposed approach and the baseline study is presented in Table 7.

Table 7. Methodological comparison with baseline study.

Aspect	Baseline Study	Our Work
Goal	League classification	Strategy recommendation
Data	46,398 raw replays	100 matches + synthetic
Features	APM, camera switches	Resource ratios, unit counts
Model	Random Forest	Causal-DNN hybrid
Performance	75.3% accuracy	96.3% F1-score
Latency	Batch processing	5.94 ms (real-time)

Key Insights:

- Data Efficiency: The baseline study [13] employed 46,398 replays to forecast player leagues with an accuracy of 75.3%. In contrast, our hybrid model focuses on strategy suggestions using a more compact curated dataset of 100 matches, augmented by:
  - Causal feature engineering (e.g., mineral-to-army thresholds)
  - Synthetic data augmentation (CTGAN) for scenario generalization

  This difference in data scale reflects distinct methodological requirements: their classification task benefits from large-scale replay analysis, while our decision-support system prioritizes targeted causal relationships.
- Interpretability: Their Random Forest operates as a black box, whereas our system provides human-readable strategy rules (e.g., "*Expand if mineral surplus > 500 with 92% confidence*").
- Task Complexity: League prediction relies on coarse features (APM, build orders), while our strategy recommendation requires fine-grained analysis of dynamic game states—a harder problem justifying our higher F1-score requirement.
- Real-Time Adaptation: Their model analyzes completed games, while ours reacts to live gameplay (5.94 ms latency vs. their batch-processing approach).

This comparison suggests that hybrid causal-ML approaches can be effective in complex, dynamic tasks like strategy optimization, even when using substantially less data.

5. Discussion

The outcomes of this study show that combining causal models with neural networks could help players make better strategic decisions in *StarCraft II*. The 1.1% rise in accuracy and F1-score seen when causal relationships were added (Table 4) shows how important it is to use both explicit knowledge and data-driven machine learning together. Although this increase is modest in absolute terms, it is significant in a highly competitive domain like real-time strategy games, where small advantages can translate into substantial differences in overall performance.

A key finding is the hybrid model's ability to recommend adaptive strategies based on available resources (Figure 6). The sensitivity matrix reveals that the system dynamically adjusts its recommendations toward defensive strategies when the player is at a resource

disadvantage and toward aggressive strategies when they have an advantage. This suggests that the model learns complex patterns from the data and incorporates fundamental tactical principles, such as efficient resource management and environmental adaptation.

The confusion matrix (Figure 7), on the other hand, shows that it is not always possible to find edge cases, especially when there are false positives and false negatives. The imbalanced nature of the original data, which prioritized offensive strategies *(attack)* in 85% of matches, underrepresents defensive scenarios. Future work should address this bias through class-balancing techniques or by generating additional synthetic data to better represent defensive strategies.

Another area for improvement is model interpretability. Although 75% of the recommendations were consistent with explicit causal rules, there is still room to increase verifiable strategic coherence. This goal could be achieved by refining the causal model or incorporating feedback from human experts to validate the system's decisions.

Finally, the results obtained are specific to the Terran race, raising questions about the model's generalization to other races (*Zerg and Protoss*). Future research should evaluate whether the proposed hybrid approach is equally effective in more diverse strategic contexts.

Basically, this study shows that adding causal models to neural networks can make recommendation systems work better and be easier to understand in environments that are always changing. However, significant challenges remain, particularly in terms of data balance, interpretability, and generalization.

6. Conclusions

In this work, we present a hybrid recommendation system that integrates causal models with neural networks to enhance strategic decision-making in *StarCraft II*. The results indicate that this integration not only improves accuracy and F1-score by 1.1% but also enables the system to adapt its recommendations based on available resources, optimizing both competitiveness and strategic sustainability.

As one of the main findings, the model shows that it can suggest defensive strategies when things are going badly and aggressive strategies when things are going well, which is in line with basic strategic principles. Additionally, the confusion matrix reveals that the model can accurately predict both victories and defeats in most cases, though there are still opportunities to improve the identification of edge cases.

The findings of this study show that adding causal modeling to deep learning frameworks can make them easier to understand and help people make better strategic decisions. From a management and planning standpoint, the capacity to clearly model causal links facilitates more robust predictions and adaptive strategies, especially in dynamic and competitive contexts such as StarCraft II. The results show that causal models can help with allocating resources better, predicting adversarial strategies more accurately, and finding the best way to respond based on what has been seen in the game. Real-world contexts, such as strategic planning in uncertain circumstances, can utilize these findings beyond the gaming sector. Future research may investigate the incorporation of causal reasoning into reinforcement learning frameworks to improve long-term planning and adaptation in intricate decision-making tasks.

This study adds to the field of artificial intelligence used in real-time strategy games by showing that the problems that come up with purely predictive methods can be fixed by combining explicit causal knowledge with data-driven machine learning. However, it also points out important areas for future research, like figuring out how to deal with strategic bias in data, making the model easier to understand, and testing how well it works in a wider range of situations.

Author Contributions: Conceptualization, F.M., J.I.V.-B. and J.A.-L.; Investigation, F.M., J.I.V.-B. and J.A.-L.; Writing—original draft, F.M., J.I.V.-B. and J.A.-L.; Writing—review & editing, F.M., J.I.V.-B. and J.A.-L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available at https://doi.org/10.5281/zenodo.14932642.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Vinyals, O.; Ewalds, T.; Bartunov, S.; Georgiev, P.; Vezhnevets, A.S.; Yeo, M.; Makhzani, A.; Küttler, H.; Agapiou, J.; Schrittwieser, J.; et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv* **2017**, arXiv:1708.04782.
2. Justesen, N.; Bontrager, P.; Togelius, J.; Risi, S. Deep learning for video game playing. *IEEE Trans. Games* **2019**, *12*, 1–20. [CrossRef]
3. Ontanón, S.; Synnaeve, G.; Uriarte, A.; Richoux, F.; Churchill, D.; Preuss, M. A survey of real-time strategy game AI research and competition in StarCraft. *IEEE Trans. Comput. Intell. AI Games* **2013**, *5*, 293–311. [CrossRef]
4. Pearl, J. Theoretical impediments to machine learning with seven sparks from the causal revolution. *arXiv* **2018**, arXiv:1801.04016.
5. Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* **2019**, *1*, 206–215. [CrossRef]
6. Salvatier, J.; Wiecki, T.V.; Fonnesbeck, C. Probabilistic programming in Python using PyMC3. *PeerJ Comput. Sci.* **2016**, *2*, e55. [CrossRef]
7. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, arXiv:1912.01703.
8. Xu, L.; Skoularidou, M.; Cuesta-Infante, A.; Veeramachaneni, K. Modeling tabular data using conditional gan. *Adv. Neural Inf. Process. Syst.* **2019**, arXiv:1912.01703.
9. Li, J.; Wang, S.; Zhang, Q.; Cao, L.; Chen, F.; Zhang, X.; Jannach, D.; Aggarwal, C.C. Causal Learning for Trustworthy Recommender Systems: A Survey. *arXiv* **2024**, arXiv:2402.08241.
10. Schölkopf, B.; Locatello, F.; Bauer, S.; Ke, N.R.; Kalchbrenner, N.; Goyal, A.; Bengio, Y. Toward causal representation learning. *Proc. IEEE* **2021**, *109*, 612–634. [CrossRef]
11. Yuan, Y.; Ding, X.; Bar-Joseph, Z. Causal inference using deep neural networks. *arXiv* **2020**, arXiv:2011.12508.
12. Shi, C.; Wang, X.; Luo, S.; Zhu, H.; Ye, J.; Song, R. Dynamic causal effects evaluation in a/b testing with a reinforcement learning framework. *J. Am. Stat. Assoc.* **2023**, *118*, 2059–2071. [CrossRef]
13. Lee, C.M.; Ahn, C.W. Feature extraction for starcraft ii league prediction. *Electronics* **2021**, *10*, 909. [CrossRef]
14. Jiang, W.; Liu, H.; Xiong, H. When graph neural network meets causality: Opportunities, methodologies and an outlook. *arXiv* **2023**, arXiv:2312.12477.
15. Mu, R. A survey of recommender systems based on deep learning. *IEEE Access* **2018**, *6*, 69009–69022. [CrossRef]
16. Gao, C.; Zheng, Y.; Wang, W.; Feng, F.; He, X.; Li, Y. Causal inference in recommender systems: A survey and future directions. *ACM Trans. Inf. Syst.* **2024**, *42*, 1–32. [CrossRef]
17. Luo, H.; Zhuang, F.; Xie, R.; Zhu, H.; Wang, D.; An, Z.; Xu, Y. A survey on causal inference for recommendation. *Innovation* **2024**, *5*, 100590. [CrossRef]
18. Hollander, M.; Wolfe, D.A.; Chicken, E. *Nonparametric Statistical Methods*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
19. Hernán, M.A.; Robins, J.M. *Causal Inference: What If*; Chapman & Hall/CRC: Boca Raton, FL, USA, 2010.
20. Pearl, J. Causal inference in statistics: An overview. *Statist. Surv.* **2009**, *3*, 96–146. [CrossRef]
21. Angrist, J.D.; Pischke, J.S. *Mostly Harmless Econometrics: An Empiricist's Companion*; Princeton University Press: Princeton, NJ, USA, 2009.
22. Morgan, S.L.; Winship, C. *Counterfactuals and Causal Inference: Methods and Principles for Social Research*; Cambridge University Press: Cambridge, UK, 2014.
23. Roy, V. Convergence diagnostics for markov chain monte carlo. *Annu. Rev. Stat. Its Appl.* **2020**, *7*, 387–412. [CrossRef]
24. Kruschke, J.K. Bayesian data analysis. *Wiley Interdiscip. Rev. Cogn. Sci.* **2010**, *1*, 658–676. [CrossRef]
25. Vinyals, O.; Babuschkin, I.; Czarnecki, W.M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D.H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **2019**, *575*, 350–354. [CrossRef] [PubMed]
26. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.

27. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2022.
28. Hastie, T.; Tibshirani, R.; Friedman, J.H.; Friedman, J.H. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 2.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.