Predicting Player Churn with LLMs: A Comprehensive Evaluation of World Knowledge and Reasoning

Tobias Schneider[1,2], Lorenz Sparrenberg[2], Rafet Sifa[1,2]
[1]Fraunhofer IAIS, Sankt Augustin, Germany
[2]University of Bonn, Bonn, Germany
tobias.schneider@iais.fraunhofer.de

***Abstract*—While large language models (LLMs) have demonstrated impressive results on public benchmarks, their effectiveness in structured, real-world problems like behavioral analytics remains underexplored. This work assesses the out-of-the-box performance of LLMs for industry-specific downstream tasks, with player churn prediction as a representative task. Evaluating LLMs on public benchmarks risks data leakage and task-specific overfitting, so instead we perform experiments on a novel self-compiled dataset for churn prediction, a task not part of any standard benchmark. We compare the performance of OpenAI's GPT-4.1 with traditional machine learning models, such as XGBoost and MLPs, and analyze the impact of the LLM's extensive internal world knowledge and reasoning capabilities. With few-shot prompting, GPT-4.1 achieves a weighted F1 score of 0.787, matching the performance of XGBoost on the same set of samples. We show that the LLM can compensate for missing information with its internal world knowledge and reasoning capabilities, performing best if it can leverage both. Our results highlight the potential of LLMs for cross-game churn prediction and other structured, industry-specific tasks.**

***Index Terms*—Large Language Models, Player Churn Prediction, Video Game Analytics, Foundation Models, In-Context Learning, Few-Shot Prompting**

I. Introduction

The rise of large language models (LLMs) and the concept of foundation models have shifted model development from training new models for each downstream task to leveraging foundation models' broad generalization capabilities [1]. Trained on enormous text corpora, LLMs can be directly applied to various downstream tasks without further model updates. While new LLMs such as GPT-4.1 [2] show strong performance on public benchmarks, overfitting on datasets or the specific tasks of the benchmark is rarely considered. A common problem when evaluating LLMs on public datasets is potential data leakage, especially with proprietary closed-source models. Therefore, it is crucial to evaluate LLMs on relevant structured industry-specific tasks that solve real-world problems. We evaluate the performance of LLMs on a novel dataset, which is not part of any LLM pretraining, to eliminate any chance of data leakage[1].

Specifically, focusing on OpenAI's newest GPT-4.1 model [2], we assess state-of-the-art LLM capabilities for player churn prediction in video games, where churning refers to a player ceasing to play the investigated game [3].

While LLMs are not commonly trained or evaluated on this task, their training data contains information about human behavior and might also cover literature or datasets, which could teach the LLM about churn prediction. Complementing the prior work [4], we analyze the churn prediction performance of LLMs without any fine-tuning on a novel dataset to enable a fair evaluation of their out-of-the-box performance. We aim to provide insights into how well LLMs can be applied to structured, industry-specific problems and assess their internal world knowledge and reasoning capabilities. In this work, we distinguish internal reasoning capabilities, the ability of LLMs to apply logic, draw conclusions, and adapt to novel tasks based on unseen data, from verbalized reasoning, where the LLM is explicitly prompted to articulate its decision process before making a final prediction. Internal reasoning refers to the model's general problem-solving skills, for example, when performance is improved by providing few-shot examples or statistics about the dataset, such as class imbalance, feature value distributions, or the importance of features.

We isolate the impact of different information in the system prompt on the classification performance of an LLM to evaluate the contributions of internal knowledge and reasoning capabilities. We perform an empirical study comparing the churn prediction performance of various LLM configurations against traditional machine learning models and a naive baseline. Our work contributes to the currently underexplored field of using LLMs for video game analytics. We address the gap introduced by limited access to public video game behavior datasets. Many game developers do not share their data, and the publicly available datasets may have been included in LLM pretraining. We provide a novel self-compiled player churn prediction dataset based on the game DDRaceNetwork [5] to demonstrate the foundation model capabilities of LLMs for

This research has been funded by the Federal Ministry of Education and Research of Germany and the state of North-Rhine Westphalia as part of the Lamarr-Institute for Machine Learning and Artificial Intelligence.

[1]The dataset and source code are available at https://github.com/tschneider-iais/llm-churn-prediction

video game churn prediction. We find that GPT-4.1 matches the performance of traditional machine learning models using few-shot prompting, highlighting the potential of LLMs for cross-game churn prediction and other structured, industry-specific problems. With limited information, the LLM compensates using its extensive internal knowledge and reasoning capabilities, performing best when utilizing both.

II. Related Work

LLMs can be applied to tasks such as machine translation [6], question answering [7], or text classification [8]–[10] using natural language prompts. Especially relevant are their in-context learning capabilities, which allow them to learn desired behavior based on provided examples in a few-shot setting [11]. This allows LLMs to be calibrated to provide specific outputs and adapt to tasks for which they were not specifically trained. For classification tasks, Hegselmann et al. achieved competitive performance using LLMs compared to state-of-the-art baseline algorithms, even outperforming them in scenarios with low data availability [8]. They claim that because of all the knowledge encoded in the parameters of LLMs, they require little or no labeled training data to obtain such a performance [8]. Even in zero-shot experiments, LLMs achieved non-trivial accuracy, indicating they were leveraging their internal knowledge [8].

Hendrycks et al. constructed the Massive Multitask Language Understanding (MMLU) benchmark, consisting of 57 tasks to evaluate the multitask accuracy of LLMs [12]. They claim that achieving high accuracy on MMLU implies that a model has extensive world knowledge and problem-solving abilities. Their results for OpenAI's GPT-3 [11] model underscore that LLMs exhibit emergent knowledge but show lopsided performance across tasks, resulting in an MMLU accuracy of 43.9%. OpenAI's newest model, GPT-4.1 [2], achieves an MMLU accuracy of 90.2%. This substantial improvement highlights human-level performance on various professional and academic tasks. LLMs show a wide range of abilities not directly connected with the task for which they are trained, underlining that LLMs possess internal models of the world [13]. Their success demonstrates deep comprehension, cognitive skills, and extensive world knowledge [13].

Given LLMs' internal reasoning capabilities and extensive world knowledge, an important question is whether they can also predict complex human behavior, such as churning. Churn prediction addresses the problem of detecting whether customers are likely to leave or cancel a subscription to a service, which among other lifespan indicators such as purchase behavior prediction [14]–[16] has become an important marketing aspect for related businesses [3], [17]–[20]. With many service businesses competing over the same customers, retaining existing customers is highly relevant, as it directly impacts business revenues [21]. Therefore, it is crucial to analyze the churning behavior of the target group so that possible future churning can be predicted and possibly prevented [22]. Churn prediction can provide valuable insights in various sectors, such as telecommunications [21], e-commerce [23], video games [3], [18]–[20], [24]–[26], and subscription-based services [17], [27]. From a technical perspective, churn prediction is a standard binary classification task, using past behavioral data as input to predict whether a person will churn or return in the future, represented by binary classes. However, for a specific dataset, the precise definition of churning is highly subjective [24]. Additionally, the underlying data is based on complex human behavior and is typically high in dimensionality, making churn prediction a complex and challenging task [21]. Nevertheless, churn prediction remains a relevant problem for many industries.

This raises the question of whether LLMs can predict human churning behavior. Meryem et al. have performed a related study, analyzing text embedding models' capabilities to uncover complex patterns in churn prediction through semantic representation [4]. However, they evaluate their approach using a publicly accessible dataset, which introduces problems of potential data leakage when using closed-source text embedding models. Furthermore, the approach does not match the performance of traditional machine learning models trained on the same dataset without using text embeddings as preprocessing.

III. Method

A. Dataset

We create a novel churn prediction dataset based on server logs of the game DDRaceNetwork (DDNet) [5], a free-to-play cooperative platformer. We collected player behavior data on two functionally equivalent game servers from 2024-09-30 to 2025-03-11, spanning 163 days. We observe the presence of all types of players, ranging from veterans to complete beginners. These include regulars, infrequent casuals, and one-time players.

DDNet has been developed open-source by its community, allowing everyone to host game servers. Typically, engagement in an entire game is considered in player churn prediction. However, as DDNet features a decentralized system of game servers, access to server logs is restricted. The selected servers provide a unique and exclusive set of playable maps, incentivizing players to return to these servers. While the servers provide a controlled environment for churn prediction, the environment adds additional challenges to regular churn prediction, as the dataset captures player engagement on a specific set of game servers rather than general engagement in the game.

The game servers record various events, such as server joins, leaves, and chat messages, in the form of textual server logs. However, the raw logs sometimes miss relevant information, leading to ambiguous cases that make parsing the logs challenging. DDNet servers support an optional feature, enabling the recording of Teehistorian [28] files. Teehistorian is a custom file format for recording all inputs a game server receives from clients, allowing it to faithfully restore all events. While it is hard to parse because it is essentially a stream of messages describing the raw inputs to the server, almost no ambiguous cases occur. However, since Teehistorian files

miss chat messages, map finishes, and IP addresses, we built a hybrid parser that uses these files as ground truth, completing the missing data with raw server logs. The parser returns game sessions, which record all available information about player behavior between joining and leaving a DDNet game server.

The free and open nature of DDNet allows easy collection of player behavior data when hosting game servers. However, the data requires a significant number of preprocessing and cleaning steps to be applicable for player behavior analysis, such as churn prediction. One challenge is the lack of unique player identification, which is crucial for behavior analysis. Instead of an account system, players can freely choose their names. In practice, most players use one primary name to be recognized by others and collect points on servers for completing maps. These points are tied to the player name used to finish a map, incentivizing players to use one primary name. However, players might change their primary name over time or temporarily use fake names for various reasons. Decent identification can be derived by combining the player names and DDNet's timeout feature. If a client loses connection, it needs a way to prove its identity upon reconnection, for which DDNet uses randomly generated timeout codes. When joining a DDNet server, the client sends its timeout code so that if it loses connection, it can correctly reconnect by providing the same timeout code again. This timeout code does not change over time by default, making it a suitable identification tool. Combining the timeout codes and player names allows them to be merged to derive an identity across devices or game reinstalls, providing an effective basis for player identification.

The available game sessions are filtered, removing sessions of people not truly playing the game or that do not allow proper player identification. Player identification relies on timeout codes, which are not available for all game sessions. This can occur because clients leave immediately after joining or never successfully connect to the server in the first place. Another source of missing timeout codes is malicious clients who use the in-game chat to promote scams, intentionally not providing a timeout code. Therefore, game sessions missing timeout codes, which make up 8.26% of the sessions, can safely be dropped without removing sessions relevant to behavior analysis. Game sessions with ambiguous names, such as whitespace or DDNet's default name ("nameless tee"), are dropped, as they complicate player identification while only making up 0.72% of the available sessions. Also, game sessions of DDNet dummy connections, allowing a player to join a server a second time using a different name, are dropped. They make up 3.97% of available game sessions, and only the player's main sessions are kept.

Changing the map played on the server, shortly losing connection, or switching between the game servers results in multiple game sessions per player. However, these game sessions are effectively part of one play session. Thus, consecutive game sessions with a gap of less than 15 seconds are merged and aggregated into play sessions.

We construct a novel churn prediction dataset from the extracted play sessions. In this work, we follow the definitions by Hadiji et al. [3]. They provide a naive churn definition (**P1**), which considers players who never return following a cut-off date as churning. In contrast, they propose a more realistic definition (**P2**) that classifies players as churning if they do not return within a specified churn period following a cut-off date. The **P2** definition effectively captures whether players are still engaged, making it more applicable for real-world settings [3]. Also, the length of the churn period is a parameter that allows one to choose whether short- or long-term engagement should be considered. We adopt their **M2** method for data generation, which generates training samples using a sliding window approach, considering various cut-off points as training dates. Observations of a player in the observation period before a training date are used to derive behavior features. The behavior following the training date is used to determine if the sample is labeled as churning or returning, following the **P2** definition. Figure 1 illustrates the data generation approach for a sample training date, showing one returning and one churning player.

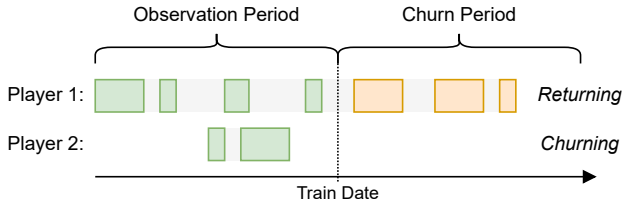

Fig. 1. Visualization of churn sample generation: Features are extracted based on the play sessions in the behavior period before a given training date. The sample's churn label is derived based on the presence of play sessions in the following churn period.

The approach models churn prediction by considering some date in the recorded past data as the cut-off date, enabling real-time features such as the player's absence time up to the chosen cut-off date. This closely reflects how churn prediction is performed in real-world applications. The dataset is generated with an observation period of seven days to cover a complete week cycle. Using brief churn periods is more applicable for real-world settings, as this also classifies players who are beginning to lose interest and only occasionally appear as churners [3]. The advantage of this approach is that it enables actively targeting players starting to show signs of disengagement, as they are easier to reactivate than people who have already left [3]. However, small churn periods lead to more churning samples in dataset generation, increasing class imbalance. For the available play sessions, a one-day churn period would result in a total churn rate of 80.1%. Such imbalanced churn samples inflate baseline scores, which is impractical for performance comparisons. Therefore, we construct the final dataset using a seven-day churn period, resulting in a total churn rate of 71.2%, balancing churn recency and class imbalance. Analyzing the absence durations between play sessions shows that 86.6% of returning players do so within a week, indicating that a seven-day churn period captures most returning players.

Analysis of the dataset showed that player activity across the game servers is lowest at 2:00 AM in their timezone. Therefore, we use 2:00 AM as the timestamp for all training dates, ensuring that player activity mostly occurs before or after the cut-off. We construct the final churn prediction dataset by iterating over all possible training dates with a stride of one day, excluding a buffer for the observation period at the beginning and churn period at the end of the data collection, as windows can otherwise overlap with the global collection bounds, resulting in incomplete behavioral context. Considering each possible training date, a sample is constructed for each player with at least one play session in the observation period. For each sample, various features are extracted based on the play sessions during the observation period. The sample is labeled as returning if play sessions occur in the following churn period. Otherwise, it is assigned to the churning class. Overall, similar to previous studies, we consider features based on playtime, session activity, temporal patterns, and game-specific events [15], [18].

B. Feature selection

Before model training and evaluation, we perform feature engineering, dropping uninformative features based on Pearson correlations. Table I presents the final feature selection, providing feature descriptions and statistics. A correlation analysis on the training set detects groups of collinear features with a correlation above 0.8, reflecting strong linear dependence. For each group, we retain the feature most strongly correlated with the churning label. The features *rescue_counter* and *messages* are highly correlated with the *total_active_mins* feature, exceeding the 0.8 threshold. However, we keep these features to retain their game-specific patterns, which are not captured by playtime-, session-, or temporal-based features. Finally, all features with a correlation to the churning label below 0.1 are excluded, as they contribute little information to the prediction task. The remaining 14 features are ranked based on their correlation with the churning label. We choose this minimal feature engineering method to compare various traditional models and LLMs fairly, as it is model-agnostic and does not assume nonlinear capabilities of the models.

C. Traditional Models

We train various traditional machine learning models as a baseline for evaluating the performance of LLMs on our novel churn prediction dataset. A test set is held out using an 80/20 split, and the remaining samples are split into training (80%) and validation (20%) sets. To ensure representative splits, they are grouped by the player identity and stratified with respect to the churning label. We evaluate the performance of Multilayer Perceptron (MLP), Logistic Regression (LR), Decision Tree (DT), Naive Bayes (NB), and XGBoost (XGB). For MLP, LR, and NB, the features are standardized.

Feature subsets with $k \in [1, 14]$ top-ranked features are considered sequentially to avoid a complete exhaustive search. We perform hyperparameter optimization for the model parameters and number of features k. For each k value, the traditional machine learning models are trained on all training samples. Optimal hyperparameters are determined based on validation performance. Then, based on the weighted F1 score, we declare the best-performing model as the representative traditional model for comparison with LLM performance.

D. LLM Performance

To fairly compare the LLM's churn prediction capabilities, we reuse the same dataset splits as for the traditional models and evaluate the LLM configurations on the test split. Any data revealed to the LLM inside the prompts, such as examples for few-shot prompting, is only derived from the training set. We perform experiments on a subsampled test set with 1000 samples to make an extensive exploration of different LLM prompt configurations feasible. While this places some limits on analyzing absolute performance, it enables a systematic comparison of relative performance, as all LLM configurations and baselines are evaluated on the exact same subset. The test performance of an XGBoost classifier on the same subsampled test set serves as a strong traditional baseline, while a classifier that always predicts the majority class serves as a naive baseline.

We perform multiple experiments to analyze to what extent LLM churn prediction performance relies on the LLM's internal knowledge or reasoning capabilities. Disabling or swapping certain parts of the LLM prompt allows one to measure the impact of the withheld information or replaced instruction. Figures 2 and 3 present the dynamic LLM prompt definitions. They include fixed parts (colored green) describing the LLM identity and churn prediction task. The system prompt features multiple optional parts (colored orange), including either predefined content such as feature descriptions or information about the dataset derived from the training set. The sections describing the model output have alternative versions (colored blue), which explicitly prompt the LLM to use verbalized reasoning before making a final decision. These optional and alternative sections are labeled using a single letter, which the following experiment descriptions will reference.

In the first group of experiments, the LLM receives comprehensive information derived from the training set, meaning that the prompt sections *A*, *C*, and *E* are enabled. We evaluate the impact of few-shot prompting, verbalized reasoning, and providing the model with detailed feature statistics. With few-shot prompting, the LLM is provided seven examples per class by enabling prompt section *F*. We chose this number of examples because preliminary validation showed that adding more examples did not further improve performance but only increased prompting cost. Representative examples are extracted by performing K-Medoids clustering on the training set. When feature statistics are enabled by including the prompt section *D*, the model receives a tabular summary of feature statistics, including the mean, standard deviation, minimum, maximum, and the 25th, 50th, and 75th percentiles.

Based on the best-performing configuration, we further evaluate how much the LLM relies on information provided in the prompts versus using internal knowledge. Specifically,

TABLE I
DESCRIPTIONS AND STATISTICS OF SELECTED FEATURES FROM THE CHURN PREDICTION DATASET. FEATURES ARE ORDERED BY CORRELATION WITH THE CHURN LABEL.

Feature	Description	Mean ± Std	Min	Max
n_sessions	Number of sessions during the observation period	2.46 ± 4.53	1	96
longest_session_gap_days	Maximum gap between sessions (days)	0.53 ± 1.10	0.00	6.83
max_session_duration_mins	Maximum session duration (minutes)	11.05 ± 30.13	0.04	1052.02
days_since_first_seen	Days since first recorded session	24.01 ± 31.51	0.00	154.57
total_active_mins	Total active playtime across sessions (minutes)	21.26 ± 88.96	0.00	2156.89
avg_active_mins	Average active playtime per session (minutes)	4.14 ± 7.78	0.00	205.93
gap_after_last_session_days	Gap after the final session (days)	3.03 ± 2.06	0.00	7.00
rescue_counter	Number of times the player used *rescue* (reset to last platform)	228.16 ± 1290.59	0	40674
var_session_start_hour	Variance of session start times (hours)	4.90 ± 14.83	0.00	274.95
var_active_mins	Variance of active playtime across sessions (minutes)	45.54 ± 263.14	0.00	11889.79
messages	Number of in-game chat messages sent	17.99 ± 112.44	0	4750
long_finish	Number of long maps completed (high time commitment)	0.31 ± 2.53	0	105
kill_counter	Number of times the player used *kill* (reset to spawn)	20.12 ± 87.11	0	7820
medium_finish	Number of medium maps completed (moderate time commitment)	0.31 ± 2.25	0	131

the impact of providing the LLM information about class imbalance and the ordering of features is analyzed, referring to the prompt sections *A* and *C*. Additionally, feature shuffling is used to analyze possible order biases in the LLM.

Finally, we evaluate the LLM's internal reasoning and problem-solving abilities by obfuscating all task-related information to transform the churn prediction problem into a generic binary classification problem. We keep the same overall prompt definition as shown in Figure 2 but modify the texts to obfuscate any information regarding churn prediction. Feature names are replaced with unique single letters, and feature explanations are disabled for all experiments, referring to prompt section *E*. Feature values are standardized to a mean of zero and a standard deviation of one. All task-specific formulations in the prompt are replaced to match a generic binary classification, such as using true and false as the model output. Then, similar to the first group of experiments, the impact of few-shot prompting, verbalized reasoning, and providing feature statistics is evaluated.

IV. RESULTS

A. Dataset

Table II reports on relevant numbers for the churn prediction dataset generation. Of the initial 166,128 game sessions, 20,554 (12.4%) are excluded by the preprocessing. For the remaining 145,574 game sessions, consecutive sessions are merged into 70,507 play sessions, resulting in an average of 2.36 game sessions per play session. Based on the extracted play sessions, 185,399 churn prediction samples are extracted using observation and churn periods of seven days, resulting in a total churn rate of 71.2%.

In total, we recorded 17,727 unique player names on the game servers. The proposed identification method reduces the number of identities to 14,969, indicating that 2,758 (15.5%) of the observed player names do not correspond to a unique identity but are likely fake names of known players. Due to the buffer periods used in the churn sample generation, the play sessions of 506 identities are excluded, leading to a total number of 14,463 identities in the final churn dataset.

TABLE II
DATASET STATISTICS FOR PREPROCESSING AND CHURN SAMPLE GENERATION.

Session Processing	
Initial game sessions	166 128
Game sessions (after preprocessing)	145 574
Merged play sessions	70 507
Player Identification	
Unique player names	17 727
Identified unique players	14 969
Identities in final dataset	14 463
Churn Dataset	
Churn prediction samples	185 399
Total churn rate	71.2%

B. Traditional Models

Table III presents the validation and test F1 scores for the selected traditional models, using the determined optimal model parameters and top k features. We provide the F1 scores for the churning class ($F1_C$) and returning class ($F1_R$), as well as the weighted F1 score ($F1_W$). Across all models, the difference in F1 scores between the validation and test sets is marginal. Models with strong nonlinear modeling capabilities (i.e., XGB, MLP, and DT) outperform LR and NB, with XGBoost achieving the highest weighted F1 score of 0.791 on both the validation and test sets. LR and NB are especially lacking in the minority returning class, for which they only achieve F1 scores of 0.550 and 0.538 on the test set.

C. LLM Performance

Table IV reports the results for the first group of LLM experiments, evaluating combinations using few-shot prompting, enabling verbalized reasoning, and providing the LLM with detailed feature statistics. To contextualize the values, we also present the performance of an XGBoost model and a naive baseline that always predicts the majority class. The table depicts the prompting cost of LLM experiments relative to the

 # Identity
You are a video game player analytic specialist. You are an expert at determining whether a player will churn or return to a game, based on user behavior data.

 # Instructions
You will be provided user behavior data of the game DDRaceNetwork from an observation period of ***{observation period}*** days. Your task is to predict whether the same player will return in the following ***{churn period}*** days.

The Features of the user behavior data are ordered by significance with respect to player churning behavior. A

Respond only with a single Word: "Churning" or "Returning", depending on your decision.

First, write one short sentence describing the available data. Then compare the data to the examples. Then come up with a final prediction and explaining your decision. Then *on a new line* respond only with a single Word: "Churning" or "Returning", depending on your decision. B

 # Dataset
The dataset is imbalanced, approximately ***{churn rate}***% of the samples are churning and ***{inverse churn rate}***% of the samples are returning. Your main priority is to to correctly classify returning samples, especially since there are fewer of them. C

 # Features Statistical Analysis
{feature statistics} D

 # Feature Explanations
{feature descriptions} E

 # Examples
You are now being provided with various examples. They consist of feature values in the same format you will receive them and the correct output label.
{examples}
 # End of Examples F

Fig. 2. Dynamic LLM system prompt definition, consisting of fixed (green), optional (orange), and alternative parts (blue). Curly brackets denote dynamic placeholders.

 # Player Behavior Data
n_sessions=7
longest_session_gap_days=2.94
max_session_duration_mins=24.21
days_since_first_seen=7.33
total_active_mins=31.58
avg_active_mins=4.51
gap_after_last_session_days=2.44
rescue_counter=326
var_session_start_hour=10.25
var_active_mins=10.52
messages=5
long_finish=0
kill_counter=40
medium_finish=1

Analyze all given information and predict if the player will churn or return. Respond with a single word: 'Churning' or 'Returning'.

Analyze all given information and first give a brief reasoning, then on a new line output 'Churning' or 'Returning'. B

Fig. 3. Dynamic LLM instruction prompt definition with example feature values. The instruction prompt consists of fixed (green) and alternative parts (blue).

TABLE III
VALIDATION AND TEST F1 SCORES FOR TRADITIONAL MODELS, WITH OPTIMAL MODEL PARAMETERS USING TOP-k FEATURES.

Model	k	Validation			Test		
		$F1_R$	$F1_C$	$F1_W$	$F1_R$	$F1_C$	$F1_W$
XGB	11	0.589	**0.871**	**0.791**	0.589	**0.872**	**0.791**
MLP	14	**0.591**	0.870	0.790	0.589	0.870	0.789
DT	7	0.585	0.868	0.787	**0.592**	0.870	0.790
LR	9	0.554	0.868	0.778	0.550	0.868	0.776
NB	4	0.544	0.862	0.771	0.538	0.861	0.768

best-performing configuration, determined by the weighted F1 score. Using the default configuration (i.e., neither few-shot, feature statistics, nor verbalized reasoning), the LLM achieves a weighted F1 score of 0.731. The default configuration outperforms the naive baseline, even when only considering the churning class, reaching an F1 score of 0.864. However, it lacks in performance in comparison to the XGB model. Enabling feature statistics results in a significant performance boost, obtaining a weighted F1 score of 0.775, which approaches the traditional model's performance using XGBoost. Enabling only few-shot prompting instead achieves a weighted F1 score of 0.787, matching the performance of XGBoost. Across all experiments, verbalized reasoning either barely changes or decreases model performance while consistently increasing prompting costs. Especially when paired with few-shot prompting, it significantly deteriorates performance. Using feature statistics generally produces good results. Notably, the best F1 score for the returning minority class is reached using few-shot prompting combined with feature statistics. However, the best weighted F1 score is achieved with just few-shot prompting, omitting verbalized reasoning and feature statistics. This configuration also provides the best balance in model performance and prompting cost.

Table V reports the results for the second group of LLM experiments. We evaluate combinations of explicitly declaring the dataset's class imbalance, revealing feature rankings, and shuffling the feature order. The first row represents the optimal configuration as a baseline, determined in the first group

TABLE IV
IMPACT OF FEW-SHOT PROMPTING, VERBALIZED REASONING, AND FEATURE STATISTICS ON LLM PERFORMANCE.

Model	Few-Shot	Reason	Stats	$F1_R$	$F1_C$	$F1_W$	Cost
Baseline	–	–	–	0.000	0.841	0.611	
XGB	–	–	–	**0.548**	**0.877**	**0.787**	
GPT-4.1	–	–	–	0.379	0.864	0.731	74%
GPT-4.1	–	–	✓	0.509	0.875	0.775	65%
GPT-4.1	–	✓	–	0.372	0.867	0.732	120%
GPT-4.1	✓	–	–	0.548	**0.877**	**0.787**	100%
GPT-4.1	✓	✓	–	0.494	0.870	0.767	121%
GPT-4.1	✓	–	✓	**0.553**	0.867	0.781	116%
GPT-4.1	–	✓	✓	0.528	0.876	0.780	97%
GPT-4.1	✓	✓	✓	0.540	0.873	0.782	145%

of LLM experiments. This configuration provides the LLM with information about class imbalance and feature ranking. Omitting the declaration of class imbalance leads to a loss in performance, particularly for the minority returning class, where the F1 score drops from 0.548 to 0.495. Instead, leaving out the information that features are ranked by importance significantly deteriorates model performance, dropping the F1 score of the returning class to 0.426 and the weighted F1 score from 0.787 to 0.749. However, shuffling the feature order to measure a potential LLM order bias does not significantly impact performance. Surprisingly, when excluding all further information (i.e., disabling declaration of class imbalance and feature ranking) and shuffling the feature order, performance is close to the baseline, slightly lacking in performance on the returning class. In all experiments, the F1 score for the churning class is mostly unchanged. All changes in performance especially impact the minority returning class.

TABLE V
IMPACT OF CLASS IMBALANCE AND ORDERING OF FEATURES ON LLM PERFORMANCE.

Declare Imbalance	Declare Ranking	Shuffle Features	$F1_R$	$F1_C$	$F1_W$	Cost
✓	✓	–	0.548	0.877	0.787	100%
–	✓	–	0.495	**0.878**	0.773	92%
✓	–	–	0.426	0.871	0.749	96%
✓	–	✓	0.418	0.873	0.748	102%
–	–	✓	**0.529**	0.878	**0.782**	91%

Table VI reports the results for the third group of LLM experiments, which repeat the experiments from the first group but transform the churn prediction task to a generic classification problem. Obfuscating the problem and not offering any information for calibration (i.e., few-shot examples or feature statistics) results in abysmal performance, with a weighted F1 score of 0.119. The F1 score for the churning majority class is near zero, indicating that the LLM mostly predicts the returning class. Enabling verbalized reasoning does not help with improving predictions. Providing the LLM with feature statistics increases the weighted F1 score from 0.119 to 0.422, exhibiting balanced F1 scores for both classes. However, the model is still clearly underperforming compared to the naive baseline. Combining feature statistics with verbalized reasoning in the obfuscated cases is the only experiment in which verbalized reasoning significantly improved model performance, raising the weighted F1 score from 0.422 to 0.651. Additionally, the verbalized reasoning allows one to inspect the LLM's decision process, which reveals that the LLM seems to assume that the minority class corresponds to strong outlier feature values while the majority class corresponds to feature values near the mean. While this is not generally true, it allows the LLM to beat the naive baseline. However, the extensive verbalized reasoning results in the highest prompting costs observed across all experiments. Similar to the non-obfuscated experiments, the best results are achieved using few-shot prompting, obtaining a weighted F1 score of 0.742 with just few-shot prompting. Contrary to the non-obfuscated experiments, combining few-shot prompting with feature statistics notably increases the F1 score for the minority returning class to 0.532, further increasing the weighted F1 score to 0.751. With this configuration, the LLM approaches the performance achieved in the non-obfuscated experiments but without providing the LLM with any churn prediction context of the classification task.

TABLE VI
IMPACT OF TASK AND FEATURE OBFUSCATION ON LLM PERFORMANCE.

Few-Shot	Reason	Stats	$F1_R$	$F1_C$	$F1_W$	Cost
–	–	–	0.421	0.005	0.119	34%
–	✓	–	0.375	0.050	0.139	84%
–	–	✓	0.427	0.421	0.422	108%
–	✓	✓	0.403	0.744	0.651	165%
✓	–	–	0.481	**0.841**	0.742	70%
✓	–	✓	**0.532**	0.834	**0.751**	98%

D. Hard Samples

Figures 4 and 5 show hard samples that both GPT-4.1 and XGBoost misclassify. The green box illustrates the input sample and the red box shows GPT-4.1's output, which matches XGBoost's prediction. The LLM prediction is based on the configuration of the first experiment group with few-shot prompting and verbalized reasoning. For these two samples, the final prediction is not influenced by the verbalized reasoning but provides valuable insights into the model behavior.

The false positive sample in Figure 4 shows very low engagement, not even reaching one minute of active playtime. There is a gap of about five and a half days after the last session, covering most of the observation period. All data point toward the player's lack of interest in the server. GPT-4.1 and XGBoost classify the sample as a churner. However, the player returns within the following seven days, possibly without much engagement.

The false negative sample in Figure 5 shows extraordinary engagement, with multiple hours of active gameplay, multiple map finishes, and social interaction via chat messages. However, the sample also has a large gap of about five days after the last session, and the player has been seen for the first time.

 # Player Behavior Data
n_sessions=2
longest_session_gap_days=1.13
max_session_duration_mins=0.37
days_since_first_seen=37.54
total_active_mins=0.73
avg_active_mins=0.37
gap_after_last_session_days=5.52
rescue_counter=0
var_session_start_hour=4.75
var_active_mins=0.00
messages=0
long_finish=0
kill_counter=2
medium_finish=0

GPT-4.1: This player has very low engagement: only 2 sessions, extremely short playtime, no chat messages, no map finishes, and a long gap after the last session, all of which are strong indicators of churning based on the examples.
Churning

Fig. 4. False Positive Prediction

 # Player Behavior Data
n_sessions=5
longest_session_gap_days=0.64
max_session_duration_mins=211.83
days_since_first_seen=5.87
total_active_mins=357.77
avg_active_mins=71.55
gap_after_last_session_days=4.94
rescue_counter=3418
var_session_start_hour=10.95
var_active_mins=1870.04
messages=581
long_finish=5
kill_counter=111
medium_finish=7

GPT-4.1: This player has a high number of sessions, very high total and average active minutes, significant engagement in chat and map completions, and high use of game mechanics, all of which are strong indicators of returning players in the examples; however, the gap after the last session is relatively high at 4.94 days, which is a slight risk factor, but not enough to outweigh the strong engagement signals.
Returning

Fig. 5. False Negative Prediction

This indicates that this is a new player who played excessively for one or two days and did not reappear until the end of the observation period. Based on the provided data, both GPT-4.1 and XGBoost classify this sample as returning. However, the player churns and does not return in the following seven days.

V. Discussion

A. Dataset

The consistent performance results between the validation and test sets for the traditional models indicate that the dataset splits are representative. We observe a dependency between the optimal number of features k and model performance. MLP performs best with all 14 available features, highlighting that all selected features convey relevant information.

The imbalance between the obtained F1 scores for the returning and churning classes highlights the structural difficulty of the underlying churn prediction task. The dataset includes some churning and returning samples that exhibit similar feature values, making them difficult to distinguish. Figures 4 and 5 show two such samples that are challenging to classify correctly for traditional machine learning models and LLMs. Because the returning class is underrepresented in the dataset, models will learn to favor the churning majority class, leading to lower performance for the returning class.

Predicting human behavior is generally a non-trivial task, especially when limited to some aggregated features. However, this also points out that some samples might be challenging to classify due to missing information. Also, the strict binary classification based on whether players return to the server during the churn period might be suboptimal. Returning players are not differentiated by their engagement, which also classifies players who join for a few seconds as returning. We propose various extensions in the future work section to overcome these challenges.

Despite the challenges, the weighted F1 scores highlight that most samples in the dataset can be correctly classified. The results show that our novel churn prediction dataset is challenging, making it suitable for comparing traditional models and LLMs.

B. LLM Performance

The initial default configuration in the first experiment group, which neither provides the LLM with few-shot examples nor feature statistics, significantly outperforms the naive baseline. The identical configuration only reaches a weighted F1 score of 0.119 in the obfuscated experiments, highlighting the LLM's use of internal knowledge to solve the task. Providing either feature statistics or examples in a few-shot setting allows the model to calibrate to our specific dataset. When using few-shot prompting with seven examples for each class, the LLM achieves practically identical performance to a traditional XGBoost model, which is trained on thousands of labeled samples. The model obtains competitive performance using the default configuration, which is heavily limited in information. Therefore, the performance can primarily be attributed to the LLM's extensive internal knowledge. However, the improvement achieved by calibrating the LLM with few-shot examples from the training set should be primarily attributed to the LLM's in-context learning and problem-solving abilities, highlighting the model's internal reasoning capabilities. These

findings suggest that both internal knowledge and reasoning contribute to the LLM's strong performance.

In the second experiment group, especially the last experiment, we deprive the LLM of all information except the few-shot examples and churn prediction task description. The LLM still achieves competitive performance, highlighting that it can substitute explicitly provided knowledge as part of the prompt with internal knowledge about churn prediction that is part of the model. However, the results of the second experiment group also show that LLMs can exploit further information about a dataset to achieve optimal performance.

In the third experiment group, we transform the churn prediction task into a generic binary classification task with various obfuscation steps to restrict the LLM's use of internal knowledge about the underlying churn prediction task. Regardless of what information the LLM is provided, it tries to find a way to exploit the available information to solve the task at hand, as highlighted by the experiment in which the LLM is only provided with the feature distributions. Aligning with the previous experiments, few-shot prompting consistently improves model performance. The LLM performs best when it also receives information about the feature distributions. In comparison, if the LLM is aware of the task at hand, we observe that providing additional information, such as feature distributions, decreases performance. This suggests that the LLM can leverage even more and less relevant information when it is not supplementing internal knowledge based on the task.

In our experiments, verbalized reasoning did not lead to consistent improvements and in several cases even reduced performance. This outcome is surprising, as it is often reported to enhance model performance [29]–[31]. However, it aligns with recent observations that even strong models can display inconsistent and counterintuitive reasoning [32] and that reasoning can increase the risk of undesirable outputs [33].

Overall, the experiments highlight that LLMs can deliver competitive performance either by compensating for missing information with internal knowledge or exploiting explicitly provided information with their strong internal reasoning capabilities. We obtain optimal results when the model can use both by providing the LLM with a task description and crucial information extracted from the training set, such as few-shot examples, feature importance, and class imbalance.

VI. Conclusion

In this work, we evaluated the impact of internal LLM knowledge and reasoning capabilities on the structured industry-specific downstream task of player churn prediction using a novel dataset unseen during LLM pretraining. We find that an LLM, specifically OpenAI's GPT-4.1 [2], can match a traditional machine learning model in performance using few-shot prompting. These results are particularly relevant, as unlike traditional models, which are trained on thousands of labeled training samples, the LLM does not require parameter updates. However, it can match the performance on a task it was not explicitly trained on, highlighting LLMs' potential as foundation models for structured industry-specific downstream tasks, such as behavioral prediction tasks, including churn prediction.

Providing the LLM with further information about the dataset, such as class imbalance, feature ordering, or feature statistics, is helpful, especially in obfuscated experiments, where the LLM is deprived of any task-specific information. Across all experiments, **few-shot prompting was crucial to obtain the optimal LLM performance**. Our experiments showed that the LLM was **effectively leveraging internal knowledge about human churning behavior**, obtaining competitive performance without any explicit information about the feature values of the dataset. Also, we showed that in obfuscated experiments, where the LLM could not leverage its prior knowledge about human churning behavior, it still achieved competitive performance by exploiting provided dataset information, highlighting its **strong internal reasoning and problem-solving capabilities**. The best LLM performance was achieved by providing the LLM with relevant dataset statistics, few-shot examples, and a task description, enabling the LLM to **leverage both its extensive world knowledge and strong internal reasoning capabilities**.

VII. Future Work

This work assessed the impact of internal LLM knowledge and reasoning capabilities on player churn prediction using OpenAI's state-of-the-art model GPT-4.1 [2]. Future work could focus on comparing different LLMs, such as older models of OpenAI or other closed-source proprietary LLMs by different vendors. Also, comparing the results with open-source models and analyzing how they react to the various experiments performed in this work would provide valuable insights. As open-source models can be self-hosted, their inference costs are significantly lower, allowing one to perform more experiments and use larger sample sizes. This also makes it feasible to assess the feature importance specifically for LLMs, compared to traditional models.

As few-shot prompting was crucial for strong LLM performance, the few-shot approach should be further optimized. For example, the number of few-shot examples, their class distribution, or the method by which they are obtained could be modified. In our experiments, enabling verbalized reasoning in the prompts did not noticeably impact performance while extensively increasing prompting cost. Experiments with guided chain-of-thought prompting should be performed to further assess the impact of explicit verbalized reasoning [34]. Methods such as *Auto-CoT* [35] could be utilized to generate reasoning chain demonstrations automatically.

We observed that some samples in the dataset are challenging to classify correctly. Additional enhancements to the dataset include adding more behavioral features, which might help differentiate between similar churning and returning players correctly. Possible features include the difficulty of the played maps, interaction with other players, the number of points collected on the servers, or even features outside the recorded servers, such as placement on the leaderboards of

the official DDNet servers. Further features would be mostly game-specific, which are particularly interesting as they are less likely to be known by the LLMs.

Our experiments used a dataset with an observation period and churn period of seven days and a strict binary definition for churning. One possible refinement is to relax the strict binary classification. As we want to capture active engagement, a player could be labeled as returning only if they return and show active engagement. This could be determined by applying a threshold to the active playtime in the churn period. The churn prediction problem could also be transformed into a multi-class prediction task, differentiating between players who return shortly and players who return after a longer time but do not churn forever. Future work could evaluate the impact of different churn prediction definitions on the model's performance and how they change the LLM's usage of its internal knowledge and reasoning capabilities.

REFERENCES

[1] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, and S. von Arx et al., "On the opportunities and risks of foundation models," 2022. [Online]. Available: https://arxiv.org/abs/2108.07258

[2] OpenAI, "Introducing GPT-4.1 in the API," April 2025, (Accessed: April 28, 2025). [Online]. Available: https://openai.com/index/gpt-4-1/

[3] F. Hadiji, R. Sifa, A. Drachen, C. Thurau, K. Kersting, and C. Bauckhage, "Predicting player churn in the wild," in *2014 IEEE Conference on Computational Intelligence and Games*. IEEE, Aug. 2014.

[4] M. Chajia and E. Nfaoui, "Customer churn prediction approach based on llm embeddings and logistic regression," *Future Internet*, vol. 16, p. 453, 12 2024.

[5] "DDraceNetwork website," (Accessed: April 28, 2025). [Online]. Available: https://ddnet.org/

[6] H. Xu, A. Sharaf, Y. Chen, W. Tan, L. Shen, B. V. Durme, K. Murray, and Y. J. Kim, "Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation," 2024. [Online]. Available: https://arxiv.org/abs/2401.08417

[7] Y. Zhuang, Y. Yu, K. Wang, H. Sun, and C. Zhang, "ToolQA: A dataset for LLM question answering with external tools," in *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. [Online]. Available: https://openreview.net/forum?id=pV1xV2RK6I

[8] S. Hegselmann, A. Buendia, H. Lang, M. Agrawal, X. Jiang, and D. Sontag, "Tabllm: Few-shot classification of tabular data with large language models," in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, F. Ruiz, J. Dy, and J.-W. van de Meent, Eds., vol. 206. PMLR, 25–27 Apr 2023, pp. 5549–5581. [Online]. Available: https://proceedings.mlr.press/v206/hegselmann23a.html

[9] A. Berger, L. Hillebrand, D. Leonhard, T. Deußer, T. B. F. De Oliveira, T. Dilmaghani, M. Khaled, B. Kliem, R. Loitz, C. Bauckhage *et al.*, "Towards automated regulatory compliance verification in financial auditing with large language models," in *2023 IEEE International Conference on Big Data (BigData)*. IEEE, 2023, pp. 4626–4635.

[10] F. Wei, R. Keeling, N. Huber-Fliflet, J. Zhang, A. Dabrowski, J. Yang, Q. Mao, and H. Qin, "Empirical study of llm fine-tuning for text classification in legal document review," in *2023 IEEE international conference on big data (BigData)*. IEEE, 2023, pp. 2786–2792.

[11] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, and P. D. et al., "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[12] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," 2021. [Online]. Available: https://arxiv.org/abs/2009.03300

[13] S. Nolfi, "On the unexpected abilities of large language models," 2023. [Online]. Available: https://arxiv.org/abs/2308.09720

[14] P. Burelli, "Predicting customer lifetime value in free-to-play games," in *Data analytics applications in gaming and entertainment*. Auerbach Publications, 2019, pp. 79–107.

[15] R. Sifa, F. Hadiji, J. Runge, A. Drachen, K. Kersting, and C. Bauckhage, "Predicting purchase decisions in mobile free-to-play games," in *proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment*, vol. 11, no. 1, 2015, pp. 79–85.

[16] A. Drachen, M. Pastor, A. Liu, D. J. Fontaine, Y. Chang, J. Runge, R. Sifa, and D. Klabjan, "To be or not to be... social: Incorporating simple social features in mobile game customer lifetime value predictions," in *proceedings of the australasian computer science week multiconference*, 2018, pp. 1–10.

[17] Y. Suh, "Machine learning based customer churn prediction in home appliance rental business," *Journal of Big Data*, vol. 10, 2023.

[18] R. Sifa, "Predicting player churn with echo state networks," in *2021 IEEE Conference on Games (CoG)*. IEEE, 2021, pp. 1–5.

[19] J. Runge, P. Gao, F. Garcin, and B. Faltings, "Churn prediction for high-value players in casual social games," in *2014 IEEE conference on Computational Intelligence and Games*. IEEE, 2014, pp. 1–8.

[20] J. T. Kristensen and P. Burelli, "Combining sequential and aggregated data for churn prediction in casual freemium games," in *2019 IEEE Conference on Games (CoG)*. IEEE, 2019, pp. 1–8.

[21] B. Huang, M. T. Kechadi, and B. Buckley, "Customer churn prediction in telecommunications," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1414–1425, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417411011353

[22] V. Lazarov and M. Capota, "Churn prediction," *Bus. Anal. Course. TUM Comput. Sci*, vol. 33, p. 34, 2007.

[23] K. Matuszelański and K. Kopczewska, "Customer churn in retail e-commerce business: Spatial and machine learning approach," *Journal of Theoretical and Applied Electronic Commerce Research*, vol. 17, no. 1, pp. 165–198, 2022. [Online]. Available: https://www.mdpi.com/0718-1876/17/1/9

[24] A. Perišić, D. Šišak Jung, and M. Pahor, "Churn in the mobile gaming field: Establishing churn definitions and measuring classification similarities," *Expert Systems with Applications*, vol. 191, p. 116277, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417421015852

[25] R. Sifa, S. Srikanth, A. Drachen, C. Ojeda, and C. Bauckhage, "Predicting retention in sandbox games with tensor factorization-based representation learning," 2016.

[26] R. Sifa, M. Fedell, N. Franklin, D. Klabjan, S. Ram, A. Venugopal, S. Demediuk, and A. Drachen, "Retention prediction in sandbox games with bipartite tensor factorization," in *Science and Information Conference*. Springer, 2020, pp. 297–308.

[27] J. Saias, L. Rato, and T. Gonçalves, "An approach to churn prediction for cloud services recommendation and user retention," *Information*, vol. 13, no. 5, 2022. [Online]. Available: https://www.mdpi.com/2078-2489/13/5/227

[28] "Teehistorian documentation," (Accessed: April 28, 2025). [Online]. Available: https://ddnet.org/libtw2-doc/teehistorian/

[29] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2022.

[30] L. Wang, W. Xu, Y. Lan, Z. Hu, Y. Lan, R. K.-W. Lee, and E.-P. Lim, "Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models," 2023. [Online]. Available: https://arxiv.org/abs/2305.04091

[31] H. S. Zheng, S. Mishra, X. Chen, H.-T. Cheng, E. H. Chi, Q. V. Le, and D. Zhou, "Take a step back: Evoking reasoning via abstraction in large language models," 2024. [Online]. Available: https://arxiv.org/abs/2310.06117

[32] W. H. Holliday, M. Mandelkern, and C. E. Zhang, "Conditional and modal reasoning in large language models," 2024. [Online]. Available: https://arxiv.org/abs/2401.17169

[33] O. Shaikh, H. Zhang, W. Held, M. Bernstein, and D. Yang, "On second thought, let's not think step by step! bias and toxicity in zero-shot reasoning," 2023. [Online]. Available: https://arxiv.org/abs/2212.08061

[34] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023. [Online]. Available: https://arxiv.org/abs/2201.11903

[35] Z. Zhang, A. Zhang, M. Li, and A. Smola, "Automatic chain of thought prompting in large language models," 2022. [Online]. Available: https://arxiv.org/abs/2210.03493