Causal Graph Fuzzy LLMs: A First Introduction and Applications in Time Series Forecasting

Omid Orang¶, Patrícia O. Lucas†‡¶, Gabriel I. F. Paiva§¶, Petrônio C. L. Silva*¶,
Felipe Augusto Rocha da Silva‡¶, Adriano Alonso Veloso¶ and Frederico Gadelha Guimarães¶
* *Federal Institute of Northern Minas Gerais*, Januária, MG, Brazil
† *Federal Institute of Northern Minas Gerais*, Salinas, MG, Brazil
‡ *Graduate Program in Electrical Engineering*, *Universidade Federal de Minas Gerais*,
Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brazil
§ *Graduate Program in Computer Science*, *Federal University of Minas Gerais*, Belo Horizonte, Brazil
¶ *Future Lab*, *Department of Computer Science*, *Federal University of Minas Gerais*, Belo Horizonte, Brazil
Email: omid.orang@dcc.ufmg.br, {patricia.lucas, petronio.candido}@ifnmg.edu.br,
gabrielpaiva18@gmail.com, fars@ufmg.br, adrianov@dcc.ufmg.br and fredericoguimaraes@ufmg.br

***Abstract*—In recent years, the application of Large Language Models (LLMs) to time series forecasting (TSF) has garnered significant attention among researchers. This study presents a new frame of LLMs named CGF-LLM using GPT-2 combined with fuzzy time series (FTS) and causal graph to predict multivariate time series, marking the first such architecture in the literature. The key objective is to convert numerical time series into interpretable forms through the parallel application of fuzzification and causal analysis, enabling both semantic understanding and structural insight as input for the pretrained GPT-2 model. The resulting textual representation offers a more interpretable view of the complex dynamics underlying the original time series. The reported results confirm the effectiveness of our proposed LLM-based time series forecasting model, as demonstrated across four different multivariate time series datasets. This initiative paves promising future directions in the domain of TSF using LLMs based on FTS.**

***Index Terms*—Time Series Forecasting, Large Language Models, Fuzzy Time Series, Causal Graph.**

I. Introduction

Time series forecasting (TSF) and analysis play a pivotal role in many real-world applications, such as energy, traffic, healthcare, finance, and meteorology [1], [2]. A variety of forecasting methods have been proposed in the literature, generally classified into three main categories: statistical, machine learning (ML), and deep learning (DL) approaches [3]. Despite their successes, these methods still face challenges such as high dimensionality, missing values, limited data availability, and the need to capture long-term dependencies—all of which are critical for accurate temporal modeling [1].

Large Language Models (LLMs), built on transformers composed of billions of parameters, have emerged to revolutionize DL models [3]. Although LLMs were originally developed for natural language processing (NLP) tasks, their application in time series (TS) analysis and forecasting has recently gained significant momentum. This shift is largely due to their ability to leverage self-attention mechanisms to capture temporal dependencies and complex dynamics, thereby effectively modeling input–output relationships [4]. This surge is driven by their remarkable ability to capture long-range dependencies and complex sequential patterns through the attention mechanism inherent in the transformers [1], [5].

According to the reviewed literature [1], [6], a number of methods leveraging LLMs have been proposed for TSF, differing in input types, model integration strategies, and LLM architectures. For instance, Time-LLM [7] uses LLaMA and GPT-2 to process multimodal TS (MTS) and relies on tokenization and fine-tuning strategies. TEMPO [8], which also employs GPT-2, handles univariate TS data using tokenization and prompt-based modeling without fine-tuning. In contrast, PromptCast [9] uses BART and BERT, focusing on prompt engineering without modifying LLM weights. Chronos [10] incorporates GPT-2 and T5, combining prompt design with model fine-tuning. LLMTIME [11] utilizes GPT-3 and LLaMA-2 to handle multimodal inputs with prompt tuning and partial integration of LLMs into the downstream task. GPT4MTS [12], employing GPT-2, and UniTime [13], also with GPT-2, both use prompt-based methods without integrating the LLM as part of the model. Meanwhile, S^2IP-LLM [14] relies on GPT-2 and fully integrates it into the forecasting pipeline. Several methods, such as LAMP [15] (using GPT-3 variants and LLaMA-2) and the model proposed in [16] (with GPT-4 and Open-LLaMA), explore newer LLMs and fine-tuning for multivariate forecasting in specific domains. These diverse strategies reflect the flexibility of LLMs in modeling sequential dependencies in TS, originally designed for text but increasingly adapted to structured temporal data. Notably, methods like Time-LLM, TEMPO, Chronos, and S^2IP-LLM provide open-source code, fostering reproducibility and further research.

In more domain-specific scenarios, the authors in [17] use ChatGPT to query multimodal data for financial forecasting without fine-tuning or integration. In the healthcare domain, Liu et al. [18] apply PaLM for MTS both forecasting and classification, although without integration or tokenization. For mobility forecasting, AuxMobLCast [19] leverages LLMs

such as BERT, RoBERTa, GPT-2, and XLNet, combining tokenization and fine-tuning strategies. LLM-Mob [20] builds on GPT-3.5, using token-based modeling without integration. Finally, in traffic forecasting, ST-LLM [21] employs LLaMA and GPT-2, utilizing tokenization and prompt engineering with full integration into the final model. These models demonstrate that LLMs can be effectively adapted beyond general domains, extending their capabilities to temporal, multimodal, and spatio-temporal forecasting tasks across sectors.

This research pioneers a new multiple-input single-output (MISO) LLM-based forecasting model termed CGF-LLM. This method combines the concepts of fuzzy time series (FTS) [22], causal graphs, and LLMs. The central objective of this work is to transform numerical time series into interpretable linguistic representations through the parallel application of fuzzification and causal analysis. This dual approach enables both semantic understanding of variable behavior and structural insight into their temporal dependencies. Specifically, the framework integrates FTS modeling with causal discovery using the PCMCI algorithm [23]. By combining these two perspectives, the method constructs a fuzzy causal text that is both data-driven and interpretable, which serves as input for GPT-2. In other words, it extracts meaningful knowledge, providing a clearer understanding of the complex dynamics within the original time series and revealing the causal relationships among variables. The results confirm that the proposed CGF-LLM technique surpasses the standard LLM in both accuracy and computational efficiency.

The remainder of this paper is structured as follows: Section II provides the basics of LLMs, FTS, and causal graphs. Section III outlines the details of the proposed CGF-LLM method. Section IV covers the case studies, results, and discussion. Finally, Section V concludes the paper and highlights the future research avenues.

II. Preliminary Concepts

A. Large Language Models

The advent of Large Language Models (LLMs) has significantly advanced natural language processing (NLP), with early landmark models such as BERT [24], GPT-2 [25], and RoBERTa [26]. These models are built upon the Transformer architecture introduced in [27], which replaced recurrence with self-attention, enabling more effective handling of sequential data.

A major leap occurred with GPT-3 [28], a 175 B-parameter model that demonstrated remarkable few-shot capabilities across diverse tasks. Its success, however, also highlighted issues such as limited transparency, bias, and occasional unreliability. Subsequent models sought to improve on these limitations. Google's PaLM [29] introduced the Pathways approach for better reasoning and coherence at scale (540B parameters), while DeepMind's Chinchilla [30] showed that smaller models (70B) can outperform larger ones when trained more efficiently. Recent models such as GPT-4 [31], LaMDA [32], LLaMA [33], and DeepSeek [34], among others, further expand capabilities.

Alongside performance gains, efforts to enhance efficiency (e.g., pruning, quantization) and interpretability (e.g., attention maps, chain-of-thought prompting) have become central to LLM research, enabling broader and safer application across domains, including time series analysis.

B. Fuzzy time series

The concept of Fuzzy Time Series (FTS) was originally proposed by [35]–[37], based on the fuzzy set theory of [38]. The main idea of the model is to convert numerical time series into linguistic representations to describe and forecast their behavior through fuzzy relation rules or matrices.

Consider a univariate time series $Y \in \mathbb{R}$, composed of observations $y(t)$ for $t = 0, 1, \ldots, T$. Each value $y(t)$ can be associated with a fuzzy set $A_i \in \tilde{A}$ through a membership function $\mu_{A_i} : \mathbb{R} \to [0, 1]$, which measures the degree of membership of $y(t)$ in A_i. The most common membership functions are triangular, trapezoidal, sigmoid, and Gaussian.

According to the approach of [37], the training process of an FTS model is divided into three main steps:

a) *Partitioning*: Universe of Discourse (UoD), represented by $U = [\min(Y), \max(Y)]$, is segmented into k overlapping subintervals. For each subinterval, a fuzzy set A_i is defined with its respective membership function μ_{A_i} and central point c_i. From these sets, the linguistic variable $\tilde{A}$ is constructed, whose terms are the A_i, with $i = 1, \ldots, k$.

b) *Fuzzification*: transforms the series Y into a fuzzy sequence F, in which each element $f(t) \in F$ corresponds to a k-component vector, representing the degrees of association of the value $y(t)$ with the fuzzy sets A_i of the linguistic variable $\tilde{A}$.

c) *Rule Generation*: from the fuzzified series F, transition patterns are identified between consecutive pairs $(f(t-1), f(t))$, from which rules of the type $A_i \to A_j, A_k, \ldots$ are extracted. These rules represent relationships such as: "IF $f(t)$ is A_i, THEN $f(t+1)$ is A_j, A_k, etc.".

The FTS model, therefore, is composed of both the linguistic variable $\tilde{A}$ and the extracted set of fuzzy rules. To perform forecasting, the model follows these steps:

a) *Input Fuzzification*: the current value $y(t)$ is converted into its fuzzy representation $f(t)$ using the previously defined membership functions.

b) *Rule Activation*: the subset of rules R is identified whose antecedents contain the fuzzy set corresponding to $f(t)$. The activation degree μ_r of each rule $r \in R$ is given by the corresponding membership value.

c) *Defuzzification*: the predicted value $y(t+1)$ is calculated based on a weighted average of the central points mp_r of the activated rules. The central point of a rule is given by:

$$mp_r = \sum_{i \in consequent} c_i \tag{1}$$

The final forecast is then obtained by:

$$y(t+1) = \frac{\sum_{r \in R} \mu_r \cdot mp_r}{\sum_{r \in R} \mu_r} \tag{2}$$

C. Causal discovery in multivariate time series

In this part the PCMCI method developed by [23] is discussed. The PCMCI is a causal discovery method that generates a causal graph directly from multivariate time series. It consists of two steps: first, using the PC_1 algorithm to identify the parents $\hat{\mathcal{P}}(X_t^j)$ for all variables in the time series $X_t^j \in X_t^1, \cdots, X_t^N$, and second, applying the momentary conditional independence (MCI) test to test for indirect links $X_{t-\tau}^i \longrightarrow X_t^j$.

PC_1 is an algorithm that uses iterative independence tests for discovering Markov sets. For each variable X_t^j the initialization of the initial parents $\hat{\mathcal{P}}(X_t^j) = (X_{t-1}, X_{t-2}, \cdots, X_{t-\tau_{max}})$ is performed. First, unconditional independence tests are applied to remove $X_{t-\tau}^i$ from $\hat{\mathcal{P}}(X_t^j)$ if the null hypothesis $X_{t-\tau}^i \perp\!\!\!\perp X_t^j$ is not rejected given a certain significance level α_{PC}. The initial parents are ranked by their absolute test statistic value.

Figure 1 illustrates PC1 for two variables X_1 and X_3 where the color intensity represents the absolute test statistic value of the dependent variables (darker color indicates higher value). Gray nodes represent independent variables. Then, conditional independence tests $X_{t-\tau}^i \perp\!\!\!\perp X_t^j | \mathcal{L}$ are performed, where $\mathcal{L}$ represents the strongest parents in $\hat{\mathcal{P}}(X_t^j) \setminus X_{t-\tau}^i$, and the independent parents are removed from $\hat{\mathcal{P}}(X_t^j)$. In this way, PC_1 converges to only a few relevant conditions that include causally linked parents with high probability (dark pink/dark blue) and potentially some false positives (dashed arrows).

In step (2), the MCI test uses the parents estimated by PC_1 to identify indirect causes. In the example illustrated in Figure 1, the conditions $\hat{\mathcal{P}}(X_t^3)$ are sufficient to establish conditional independence and test $X_{t-2}^1 \longrightarrow X_t^3$. Additionally, lagged parents from $\hat{\mathcal{P}}(X_{t-2}^1)$ are included as additional conditions, and they are responsible for maintaining the false positive rate at an expected level.

The PCMCI method is based on the assumptions of causal sufficiency[1], the Causal Markov Condition, and the faithfulness assumption. It also does not assume contemporaneous causal effects and assumes stationarity [23]. The PCMCI has a polynomial complexity in the number of variables N and τ_{max}. In the worst-case scenario, where the graph is fully connected, the computational complexity of the PC1 condition selection stage for N variables equates to $N^3 \tau max^2$. The MCI step involves additional tests of $N^2 \tau_{max}$ (for $\tau > 0$). Therefore, the total worst-case computational complexity in terms of the number of variables is polynomial and given by $N^3 \tau_{max}^2 + N^2 \tau_{max}$.

III. Causal Graph Fuzzy LLM

This section introduces CGF-LLM, a novel multivariate time series forecasting framework that integrates FTS, causal graphs, and LLMs. The method begins by converting numerical time series into linguistic variables through fuzzification. Next, a causal graph is constructed using the PCMCI algorithm [23] to capture temporal and causal relationships among variables. Specifically, the framework integrates FTS modeling with causal discovery using the PCMCI algorithm, where PCMCI identifies which variables are causally relevant, and fuzzification captures how those variables evolve over time. The resulting fuzzy linguistic descriptions and causal relations are combined into interpretable textual representations, which are provided as input to a pretrained GPT-2 model. The LLM generates a linguistic forecast, which is then defuzzified to obtain the final numerical prediction.

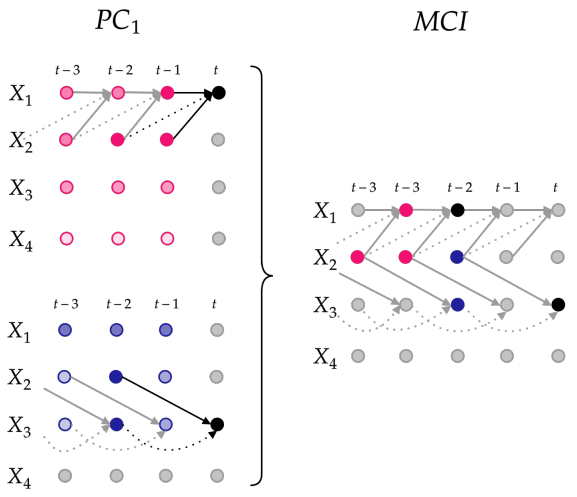

Fig. 1. Illustration of the PCMCI method.

As illustrated in Figure 2, CGF-LLM frames the forecasting task as a MISO (Multiple Input Single Output) system, where the n time series variables are treated as inputs $(Y^0, Y^1, Y^2, \ldots, Y^n)$, and the output corresponds to the endogenous variable Y^0, which is to be predicted at time step $t+1$.

The CGF-LLM methodology comprises four main steps:

1) ***Fuzzification:*** The Universe of Discourse of the n time series is divided into K overlapping intervals, with each fuzzy set f_i^j defined by its membership function $\mu_{f_i^j}$ and central point $c_{f_i^j}$, where $i = 1, \ldots, K$ and $j = 0, \ldots, n$. Each time series is then converted into a fuzzy time series F^j, where each element $f^j(t) \in F^j$ is a K-dimensional vector representing the degrees of membership of the value $y^j(t)$ in the fuzzy sets f_i^j.
2) ***Causal Graph Construction:*** The PCMCI algorithm is used to identify causal relationships among variables, resulting in a causal graph G for the endogenous variable Y^0. This graph is constructed based on all input variables, considering a maximum lag window of size $\tau_{\max}$.
3) ***Text Generation:*** Based on the fuzzy series f^j and the

[1]Implying that all common drivers are among the observed variables.

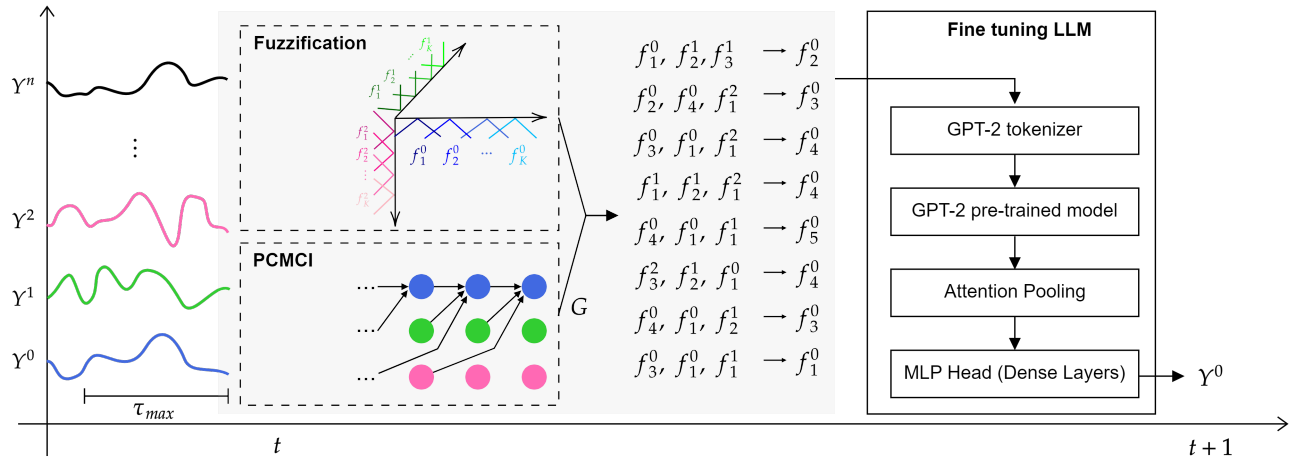

Fig. 2. Schematic overview of the proposed CGF-LLM forecasting model. Y^j denotes the input multivariate time series, where $j = 0, \ldots, n$. Y^0 corresponds to the endogenous time series. τ_{max} represents the size of the lag window observed by the PCMCI method. The fuzzification transforms each Y^j into a FTS f^j using K fuzzy sets. Then, the temporal linguistic patterns are generated using parallel application of fuzzification and PCMCI. For example, the causal relation $f_1^0, f_2^1, f_3^1 \rightarrow f_2^0$ illustrates that the fuzzy representation f_1^0 (value of Y^0 in fuzzy set 1), f_2^1 and f_3^1 (values of Y^1 in fuzzy sets 2 and 3) influence f_2^0, as identified by PCMCI. Arrows in the causal graph (G) denote directed causal relationships among fuzzy sets, such that f_1^0 to f_2^0, f_2^1 to f_2^0 and others.

causal graph G, temporal linguistic patterns are extracted that reflect the identified dependencies. For example, if G contains the links $Y^0(t-1) \rightarrow Y^0(t)$ and $Y^1(t-1) \rightarrow Y^0(t)$, the resulting pattern will be: $F(Y^0(t-1)), F(Y^1(t-1)) \rightarrow F(Y^0(t))$.

4) ***Forecasting:*** The linguistic patterns derived from the FTS and causal graph are used to fine tune the GPT-2 model, originally designed for text generation, for numerical TSF. In more details, pretrained GPT-2 is integrated with additional layers that convert its textual output into a numerical value. This conversion involves a self-attention-based pooling step to aggregate token embeddings into a single representative vector that captures the global context of the sequence. This vector is then processed through a series of linear layers to produce the final output: the numerical forecast of variable Y^0 at time $t+1$.

IV. Computational Experiments

A. Case studies

The datasets were named according to their respective application domains: economic, energy, Internet of Things (IoT), and climate. To support the reproducibility of this research, all datasets are publicly available at Datasets. Table I provides a summary of the main details of the datasets. Accordingly, the proposed method is tested on these datasets to predict solar radiation in Brazil, wind power generation, household electricity consumption in Mexico, and Bitcoin prices in the United States.

B. Experimental Methodology

The experiments were conducted as an ablation study to assess the impact of the proposed technique (CGF-LLM) on the performance of the GPT-2 model in multivariate time series forecasting. To this end, we compared the methods in terms of prediction accuracy and the number of tokens generated. Three configurations were evaluated: (i) the complete proposed method (CGF-LLM), (ii) a variation without the fuzzyfication step (CG-LLM), and (iii) a baseline LLM, in which numerical time series values were simply converted into text.

Each configuration was trained using two strategies: with freezing, where GPT-2's internal parameters were frozen and only an additional output layer was trained, and no freezing, where both GPT-2 and the additional layers were fine-tuned.

CGF-LLM was configured with the following hyperparameters: lag window size $\tau_{max} = 20$, $\alpha_{PC} = 0.1$, number of partitions set to 30, grid-based partitioning method, and triangular membership function. These values were empirically defined, taking into account the computational cost of running the experiments. The CG-LLM and LLM approaches used the same hyperparameters, adjusted according to their respective architectures. In all three approaches, the GPT-2 model was fine-tuned for 20 epochs.

For the comparative analysis, four datasets from distinct domains and with varying dimensionalities were employed. The experiments were conducted over 10 time windows, each with a size of $0.3 \cdot |D|$ and an overlap of 30% along the multivariate time series of length $|D|$. Each window was split into training and test subsets, with 20% of the data reserved for testing.

Prediction accuracy was assessed using the average Normalized Root Mean Square Error (NRMSE) computed across the ten windows, as defined in Equation (3). Here, $y_{\max}$ and $y_{\min}$ denote the maximum and minimum values within the test set, respectively.

$$\text{NRMSE} = \frac{\sqrt{\sum_{t=0}^{n}(y(t) - \hat{y}(t))^2}}{y_{\max} - y_{\min}} \tag{3}$$

TABLE I
SUMMARY OF DATASETS USED IN THE EXPERIMENTS

Dataset	Description	Target Variable	Samples	Variables	Frequency
ECONOMIC	Bitcoin Daily Price Index	AVG	2.970	6	Daily
ENERGY	Wind Power Production	Power	43.800	9	Hourly
IoT	Household Electricity	active_power	100.000	14	Minutely
CLIMATIC	Solar Radiation	glo_avg	35.000	12	Minutely

It is worth noting that all the experiments are implemented and tested in Python 3, using open source libraries including PyFTS [39], Tigramite[2] and Hugging Face[3]

To promote transparency and reproducibility of the proposed model, source code and datasets are available at https://github.com/CBIC2025.

C. Results and Discussion

Table II summarizes the means and standard deviations of the NRMSE for one-step-ahead forecasts across all datasets using three models: CGF-LLM, CG-LLM, and LLM. Two training strategies are compared: no freezing, where all the parameters of the GPT-2 model are fine-tuned for the forecasting task, and freezing, where only the last two layers (including attention pooling and MLP head) are fine-tuned to reduce the computational demands.

Across all datasets, the no-freezing strategy consistently achieves lower values of average NRMSE compared to those with freezing, indicating that full fine-tuning of the GPT-2 model improves the forecasting accuracy. For example, in the Energy dataset, CGF-LLM records an NRMSE of 0.066 ± 0.006 without freezing versus 0.153 ± 0.009 with freezing, highlighting a substantial performance gap. This trend holds across the other three datasets as well. The superior performance of the no-freezing strategy likely stems from its potential to fully adjust the model to the dynamic patterns of datasets, though it needs greater computational resources.

Comparing the models, CGF-LLM outperforms CG-LLM and LLM in all scenarios, with or without freezing. The exception occurs in the CLIMATIC with freezing, where CG-LLM reports the lowest forecasting error in comparison to CGF-LLM and LLM. In the ECONOMIC dataset, performance differences among the models are minimal, with overlapping standard deviations (e.g., 0.094 ± 0.018 for CGF-LLM vs. 0.104 ± 0.018 for LLM without freezing), suggesting no statistically significant distinction.

Thus, the obtained results underscore the efficacy of the proposed CGF-LLM for MISO time series forecasting, particularly under the no-freezing strategy. CG-LLM ranks second in most cases, while LLM generally performs the least effectively. These findings highlight a trade-off between computational efficiency and forecasting precision, with CGF-LLM offering the most robust solution when resources permit fine-tuning the full model.

In addition to the accuracy enhancement, Table III reveals the computational efficiency of the proposed CGF-LMM method regarding the number of tokens. Although the total text size produced by the combined use of causal graphs and FTS is greater than that of CG-LLM without fuzzy logic, CGF-LLM results in significantly fewer tokens compared to both CG-LLM and standard LLM approaches. The reason is that fuzzy labels tokenize into fewer units with the GPT-2 tokenizer due to their repetitive nature and simpler structure compared to numerical strings. For instance, a label like "f_1" often results in a single token, whereas "23.5" may split into multiple tokens (e.g., "23" and ".5"). Thus, CGF-LLM substantially reduces the number of tokens. For instance, on the IoT dataset, CGF-LLM produces 36 times fewer tokens than the standard LLM (839,412 vs. 30,699,520), leading to a drastic reduction in both memory usage and computational requirements.

In summary, CGF-LLM not only minimizes tokenization overhead and computational costs but also delivers the highest prediction accuracy, leveraging FTS and causal graphs to capture essential temporal relationships effectively. This evidence suggests that strategic pre-processing, such as that used by the CGF-LLM method, is critical for optimizing the scalability and cost-effectiveness of LLM-based forecasting methods.

V. CONCLUSION

This research introduces an LLM-based TSF method named CGF-LLM, which combines the concepts of causal graphs, FTS, and LLMs (fine-tuned GPT-2), applied to MISO applications. The key innovation of this technique relies on converting numerical time series into fuzzy causal text as input for the GPT-2 model, achieved through the parallel use of fuzzification and causality. The generated text provides an interpretable representation of the complex dynamics within the original time series. This method was tested across four different datasets, and the obtained results highlight the efficacy of the proposed CGF-LLM in terms of accuracy and computational costs compared to CG-LLM and regular LLM.

Despite the success of CGF-LLM, the current study utilized predefined values of hyperparameters. Thus, one possible future direction would be to optimize the model's hyperparameters via auto ML or other advanced optimization techniques. Another research avenue is to explore other alternatives such as GPT-3, GPT-4, and others. Also, future work may extend CGF-LLM to support multiple-output forecasting tasks.

ACKNOWLEDGMENT

This work has been supported by the Brazilian agencies (i) National Council for Scientific and Technological Development (CNPq), Grant no. 304856/2025-8, "*Aprendizado de Máquina Colaborativo e Proteção à Privacidade*"; (ii)

[2]https://jakobrunge.github.io/tigramite/

[3]https://huggingface.co/docs/transformers/model_doc/gpt2

TABLE II

COMPARISON OF FUZZY LLM AND LLM MODELS, WITH AND WITHOUT PARAMETER FREEZING, ACROSS DATASETS. ALL RESULTS ARE BASED ON ONE-STEP-AHEAD FORECASTING USING THE NRMSE (NORMALIZED ROOT MEAN SQUARED ERROR) METRIC.

Datasets	No Freezing			With Freezing		
	CGF-LLM	CG-LLM	LLM	CGF-LLM	CG-LLM	LLM
ECONOMICS	0.094 ± 0.018	0.113 ± 0.017	0.104 ± 0.018	0.201 ± 0.032	0.238 ± 0.029	0.222 ± 0.034
ENERGY	0.066 ± 0.006	0.083 ± 0.003	0.104 ± 0.004	0.153 ± 0.009	0.212 ± 0.012	0.246 ± 0.004
IoT	0.027 ± 0.003	0.030 ± 0.008	0.075 ± 0.005	0.061 ± 0.006	0.077 ± 0.067	0.082 ± 0.004
CLIMATIC	0.029 ± 0.002	0.063 ± 0.006	0.077 ± 0.006	0.135 ± 0.015	0.116 ± 0.011	0.125 ± 0.019

TABLE III

COMPARISON OF TEXT SIZE AND TOKEN METRICS FOR CGF-LLM, CG-LLM, AND LLM ACROSS DATASETS

Dataset	Metric	CGF-LLM	CG-LLM	LLM
ECONOMICS	Total Text Size	43,875	40,643	1,335,281
	Train Text Size	35,034	32,627	1,069,999
	Test Text Size	8,841	8,016	265,282
	Total Tokens	17,340	32,984	846,300
	Train Tokens	13,880	26,410	677,625
	Test Tokens	3,460	6,574	168,675
ENERGY	Total Text Size	1,276,952	1,190,523	23,194,823
	Train Text Size	1,021,690	952,364	18,564,729
	Test Text Size	255,262	238,159	4,630,094
	Total Tokens	380,451	839,680	13,434,880
	Train Tokens	304,384	671,744	10,747,904
	Test Tokens	76,067	167,936	2,686,976
IoT	Total Text Size	3,147,800	2,064,031	94,793,482
	Train Text Size	2,518,273	1,651,208	75,825,454
	Test Text Size	629,527	412,823	18,968,028
	Total Tokens	839,412	1,738,840	30,699,520
	Train Tokens	671,552	810,007	24,559,616
	Test Tokens	167,860	347,768	6,139,904
CLIMATIC	Total Text Size	786,258	660,437	29,934,389
	Train Text Size	629,000	528,553	23,949,302
	Test Text Size	157,258	131,884	5,985,087
	Total Tokens	461,076	576,400	10,731,520
	Train Tokens	368,896	461,120	8,585,216
	Test Tokens	92,180	115,280	2,146,304

Coordination for the Improvement of Higher Education Personnel (CAPES) through the Academic Excellence Program (PROEX).

Partially funded by the R&D&I agreement between UFMG and Fundep – Research Development Foundation: "Federated Machine Learning Program for Connected Vehicles".

The authors acknowledge partial support from Kunumi and Embrapii, project PDCC-2412.0030.

REFERENCES

[1] S. Abdullahi, K. Usman Danyaro, A. Zakari, I. Abdul Aziz, N. Amila Wan Abdullah Zawawi, and S. Adamu, "Time-series large language models: A systematic review of state-of-the-art," *IEEE Access*, vol. 13, pp. 30 235–30 261, 2025.

[2] H. Liu, H. Kamarthi, Z. Zhao, S. Xu, S. Wang, Q. Wen, T. Hartvigsen, F. Wang, and B. A. Prakash, "How can time series analysis benefit from multiple modalities? a survey and outlook," 2025. [Online]. Available: https://arxiv.org/abs/2503.11835

[3] J. Kim, H. Kim, H. Kim, D. Lee, and S. Yoon, "A comprehensive survey of deep learning for time series forecasting: Architectural diversity and open challenges," 2025. [Online]. Available: https://arxiv.org/abs/2411.05793

[4] W. Chow, L. Gardiner, H. T. Hallgrímsson, M. A. Xu, and S. Y. Ren, "Towards time series reasoning with llms," 2024. [Online]. Available: https://arxiv.org/abs/2409.11376

[5] V. Ekambaram, A. Jati, P. Dayama, S. Mukherjee, N. H. Nguyen, W. M. Gifford, C. Reddy, and J. Kalagnanam, "Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series," 2024. [Online]. Available: https://arxiv.org/abs/2401.03955

[6] Y. Jiang, Z. Pan, X. Zhang, S. Garg, A. Schneider, Y. Nevmyvaka, and D. Song, "Empowering time series analysis with large language models: A survey," *arXiv preprint arXiv:2402.03182*, 2024.

[7] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan *et al.*, "Time-llm: Time series forecasting by reprogramming large language models," *arXiv preprint arXiv:2310.01728*, 2023.

[8] D. Cao, F. Jia, S. O. Arik, T. Pfister, Y. Zheng, W. Ye, and Y. Liu, "Tempo: Prompt-based generative pre-trained transformer for time series forecasting," *arXiv preprint arXiv:2310.04948*, 2023.

[9] H. Xue and F. D. Salim, "Promptcast: A new prompt-based learning paradigm for time series forecasting," *IEEE Trans. on Knowl. and Data Eng.*, vol. 36, no. 11, p. 6851–6864, Nov. 2024. [Online]. Available: https://doi.org/10.1109/TKDE.2023.3342137

[10] A. F. Ansari, L. Stella, C. Turkmen, X. Zhang, P. Mercado, H. Shen, O. Shchur, S. S. Rangapuram, S. P. Arango, S. Kapoor, J. Zschiegner, D. C. Maddix, H. Wang, M. W. Mahoney, K. Torkkola, A. G. Wilson, M. Bohlke-Schneider, and Y. Wang, "Chronos: Learning the language of time series," 2024. [Online]. Available: https://arxiv.org/abs/2403.07815

[11] N. Gruver, M. Finzi, S. Qiu, and A. G. Wilson, "Large language models are zero-shot time series forecasters," 2024. [Online]. Available: https://arxiv.org/abs/2310.07820

[12] F. Jia, K. Wang, Y. Zheng, D. Cao, and Y. Liu, "Gpt4mts: prompt-based large language model for multimodal time-series forecasting," ser. AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. [Online]. Available: https://doi.org/10.1609/aaai.v38i21.30383

[13] X. Liu, J. Hu, Y. Li, S. Diao, Y. Liang, B. Hooi, and R. Zimmermann, "Unitime: A language-empowered unified model for cross-domain time series forecasting," in *Proceedings of the ACM Web Conference 2024*, ser. WWW '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 4095–4106. [Online]. Available: https://doi.org/10.1145/3589334.3645434

[14] Z. Pan, Y. Jiang, S. Garg, A. Schneider, Y. Nevmyvaka, and D. Song, "S2ip-llm: semantic space informed prompt learning with llm for time series forecasting," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24. JMLR.org, 2024.

[15] X. Shi, S. Xue, K. Wang, F. Zhou, J. Zhang, J. Zhou, C. Tan, and H. Mei, "Language models can improve event prediction by few-shot abductive reasoning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 29 532–29 557, 2023.

[16] X. Yu, Z. Chen, Y. Ling, S. Dong, Z. Liu, and Y. Lu, "Temporal data meets llm–explainable financial time series forecasting," *arXiv preprint arXiv:2306.11025*, 2023.

[17] A. Lopez-Lira and Y. Tang, "Can chatgpt forecast stock price movements? return predictability and large language models," 2024. [Online]. Available: https://arxiv.org/abs/2304.07619

[18] X. Liu, D. McDuff, G. Kovacs, I. Galatzer-Levy, J. Sunshine, J. Zhan, M.-Z. Poh, S. Liao, P. Di Achille, and S. Patel, "Large language models are few-shot health learners," *arXiv preprint arXiv:2305.15525*, 2023.

[19] H. Xue, B. P. Voutharoja, and F. D. Salim, "Leveraging language foundation models for human mobility forecasting," in *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, 2022, pp. 1–9.

[20] X. Wang, M. Fang, Z. Zeng, and T. Cheng, "Where would i go next? large language models as human mobility predictors," 2024. [Online]. Available: https://arxiv.org/abs/2308.15197

[21] C. Liu, S. Yang, Q. Xu, Z. Li, C. Long, Z. Li, and R. Zhao, "Spatial-temporal large language model for traffic prediction," in *2024 25th IEEE*

International Conference on Mobile Data Management (MDM). IEEE, 2024, pp. 31–40.

[22] P. O. Lucas, O. Orang, P. C. Silva, E. Mendes, and F. G. Guimaraes, "A tutorial on fuzzy time series forecasting models: recent advances and challenges," *Learn Nonlinear Models*, vol. 19, pp. 29–50, 2022.

[23] J. Runge, P. Nowack, M. Kretschmer, S. Flaxman, and D. Sejdinovic, "Detecting and quantifying causal associations in large nonlinear time series datasets," *Science Advances*, vol. 5, no. 11, pp. 4996–5023, 11 2019. [Online]. Available: https://www.science.org/doi/10.1126/sciadv.aau4996

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[25] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[26] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[28] T. B. Brown *et al.*, "Language models are few-shot learners," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[29] A. Chowdhery *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 1, pp. 240:1–240:113, 2023. [Online]. Available: https://www.jmlr.org/papers/volume24/21-1378/21-1378.pdf

[30] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark *et al.*, "Training compute-optimal large language models," *arXiv preprint arXiv:2203.15556*, 2022.

[31] O. et al., "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[32] R. T. et al., "Lamda: Language models for dialog applications," 2022. [Online]. Available: https://arxiv.org/abs/2201.08239

[33] T. et al., "Llama: Open and efficient foundation language models," 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

[34] D.-A. et al., "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," 2025. [Online]. Available: https://arxiv.org/abs/2501.12948

[35] Q. Song and B. S. Chissom, "Forecasting enrollments with fuzzy time series—part i," *Fuzzy sets and systems*, vol. 54, no. 1, pp. 1–9, 1993.

[36] ——, "Forecasting enrollments with fuzzy time series—part ii," *Fuzzy sets and systems*, vol. 62, no. 1, pp. 1–8, 1994.

[37] S.-M. Chen, "Forecasting enrollments based on fuzzy time series," *Fuzzy sets and systems*, vol. 81, no. 3, pp. 311–319, 1996.

[38] L. A. Zadeh, "Fuzzy sets," *Information and control*, vol. 8, no. 3, pp. 338–353, 1965.

[39] P. C. e Lima E Silva, C. A. S. Júnior, M. A. Alves, R. C. P. Silva, G. L. Vieira, P. De Oliveira E Lucas, H. J. Sadaei, and F. G. Guimarães, "PYFTS/pyFTS: Stable version 1.7," 2019.