Title

Anonymous ACL submission

Abstract

TO DO

1 Introduction

2 Related Works

3 Methodology

3.1 Creating the Phrase Dataset

3.1.1 Scraping

We collected COPOM (Central Bank of Brazil's Monetary Policy Committee) minutes using Python and Selenium. We accessed `https://www.bcb.gov.br/publicacoes/atascopom/cronologicos`, which contains the listing of all of them. For each minute, we downloaded both the HTML and PDF content when available.

We ended up with a dataset C containing 251 COPOM minutes from January 1996 to July 2025. Each minute c in C has an associated date d_i and may have one or both HTML and PDF versions of the content.

3.1.2 Parsing

For each COPOM minute c in C:

1. Type-Specific Pre-Processing

   HTML file: if it exists, we extracted only the content inside the body tag. Tags such as `strong`, `i`, and `br` were removed while preserving their inner content. Other tags were removed along with their content.

   PDF file: if it exists, we used SpaCyLayout with the `pt_core_news_lg` model to extract individual phrases from PDF documents.

   After that, we created two separate phrase lists: one from the HTML source P_c^{html} and another from the PDF source P_c^{pdf}.

2. General Pre-Processing

   For each phrase in both P_c^{html} and P_c^{pdf}, we applied the following steps in that order: (1) Removed newlines and tabs; (2) Removed remaining tag entities (e.g., ` `); (3) Reduced multiple consecutive spaces, commas, and periods to single characters; (4) Added a period at the end if it did not exist.

3. Length Filtering

   For both P_c^{html} and P_c^{pdf} sets, we applied the following steps in that order: (1) Discarded single-word phrases; (2) Discarded phrases with character count below μ, the mean character count of phrases from the respective source P_c^{x}.

4. Blacklist Filtering

   We removed phrases containing at least one of the words from the following list: (1) *javascript*; (2) *cookies*; (3) *expand_less*; (4) *content_copy*; (5) *Garantir a estabilidade do poder de compra da moeda*.

   While terms (1) to (4) are related to web page elements and scripts, term (5) is the Brazilian Central Bank's motto, which often appears in the minutes and is not relevant for sentiment analysis.

Finally, we compared the number of phrases between sets P_c^{html} and P_c^{pdf} for each minute c. We selected the set with the most phrases; if both sets had equal size, we chose the PDF version as it appeared to have an overall superior phrase quality. When either source was unavailable or contained insufficient information, this step ensured we obtained the most reliable set for each minute.

At the end we obtained a set F made of smaller sets F_{d_i} for each date d_i, where d_i is the associated date of minute c. Each F_{d_i} contained 20 to 70 phrases.

3.1.3 Phrase Selection

We flattened the set F into a single list of phrases while preserving each phrase date labels, creating a list L of tuples $(\text{phrase}, \text{date})$.

We performed **dense passage retrieval** using semantic similarity filtering. We computed dense vector representations (embeddings) for all phrases using the **Qwen3-Embedding-0.6B** model and computed the cosine similarity between each phrase embedding and the embedding of the target concept "inflation". We retained only phrases with a cosine similarity score exceeding a threshold of 0.6, thereby selecting phrases semantically related to inflation concepts.

The implementation utilized PyTorch for GPU acceleration, pandas for data manipulation, scikit-learn for similarity computations, and the LangChain HuggingFace integration for embedding generation.

We then constructed a set of tuples $(\text{phrase}, \text{date})$ containing only the selected phrases named F^{infl}.

F^{infl} is the final phrase dataset used in subsequent steps. It contains 9378 phrases related to inflation across 251 dates (or COPOM minutes), an average of approximately 37.4 phrases per date.

3.2 Creating the Sentiment Datasets

3.2.1 LLM Evaluation Dataset

We evaluated the sentiment of the phrases using nine different Large Language Models (LLMs), each one made from a different company:

1. *openai/gpt-5*
2. *anthropic/claude-sonnet-4*
3. *google/gemini-2.5-pro*
4. *x-ai/grok-4-fast*
5. *openai/gpt-oss-120b*
6. *meta-llama/llama-4-maverick*
7. *google/gemma-3-27b-it*
8. *microsoft/phi-4*
9. *deepseek/deepseek-chat-v3.1*

For each model in the list above, we made one independent request **for each phrase** of the dataset F^{infl}, without providing previous context.

The prompt was formulated in Brazilian Portuguese by our specialist economist Cézio Luiz Ferreira Junior. It contained a fixed text that explained the task and the phrase to be evaluated concatenated at the end:

> **DEFINIÇÃO DE OTIMISMO:** Ocorre quando as projeções indicam que a inflação ficará abaixo da meta ou dentro do intervalo de tolerância com folga. Isso pode sinalizar que o Banco Central vê espaço para reduzir juros ou manter uma política monetária mais acomodatícia.
>
> **DEFINIÇÃO DE PESSIMISMO:** Ocorre quando as projeções apontam para inflação acima da meta ou próxima do teto do intervalo de tolerância. Isso sugere preocupação com pressões inflacionárias e pode justificar uma política monetária mais restritiva.
>
> **AVALIE A FRASE COMO:** O para OTIMISTA, N para NEUTRA, P para PESSIMISTA. SUA RESPOSTA DEVE SER APENAS UMA LETRA, SEM QUALQUER OUTRO TEXTO.
>
> **FRASE A SER AVALIADA:** ««PHRASE»»

In the prompt we asked the model to classify each phrase as optimistic, neutral, or pessimistic based on the provided definitions. Model responses (O, N, P) were converted to numerical values: 1 for optimistic, 0 for neutral, and -1 for pessimistic. Responses that could not be parsed were labeled as -2, but such cases were rare.

Inference was performed using the OpenRouter API to unify model access and each model was assigned a maximum token limit determined through initial testing.

The maximum token limit was determined by testing the models on the phrases from the first $F^{infl}_{d_i} \in F^{infl}$. With an initial token limit of 1, if any phrase received a -2 score in this first set, the limit was doubled and the test was repeated until the model could process all the set's phrases successfully.

The resulting maximum token limits were shown in Table 1. Interestingly, OpenAI's models needed considerably higher token limits compared to other models, followed by Google's.

To ensure consistency and more reliable results, we discarded any evaluations where the sentiment wasn't 1 and -1.

Finally, we concatenated the results into sets named E_m for each model m. Each E_m contained tuples of the form $(\text{phrase}, \text{date}, \text{sentiment})$.

The set that contains all sets E_m is named E_{Models}.

Model	Token Limit
openai/gpt-5	1024
openai/gpt-oss-120b	512
google/gemini-2.5-pro	128
google/gemma-3-27b-it	8
deepseek/deepseek-chat-v3.1	4
others	1

Table 1: Maximum token limits per LLM model.

3.3 Human Evaluation Dataset

Similar to the previous section, we created three different human evaluation datasets:

1. *Open*

   We created a website featuring the same evaluation system used for LLMs presented in section 3.2.1, but adapted for humans to select between O (optimistic), N (neutral), and P (pessimistic) options instead of reading API responses. The phrases were randomly selected from the set F^{infl} and each browser was limited to evaluating 10 phrases per 24-hour period.

   We requested collaborating universities (USP and Unicamp) to share the website with their economics-related graduate students. It is publicly accessible at `https://inflation-form.luvas.io`.

2. *Specialist*

   We created a subset named $F^{infl-350}$ consisting of 350 randomly selected phrases from F^{infl}. The date labels were encoded in Base64 to prevent human bias.

   Then, our specialist economist Cézio Luiz Ferreira Junior, manually labeled each phrase as: 1 for optimistic, 0 for neutral, -1 for pessimistic, -2 for non-related phrase, or -3 for did not understand. The definitions used were also the same as those presented in the prompt for LLMs in section 3.2.1.

3. *Consolidated*

   The $F^{infl-350}$ dataset and its sentiment labels from the Specialist evaluation was re-analyzed by the specialist and two additional professors in conjunction. They discussed each phrase and attempted to reach consensus.

To ensure consistency and more reliable results, we discarded any evaluations where the sentiment wasn't 1 and -1 for all labels produced by humans.

Finally, we created a dataset E_h for each human evaluation method h presented. Each E_h contained tuples of the form $(\text{phrase}, \text{date}, \text{sentiment})$, where sentiment is the label assigned by the humans in the respective evaluation method. In the end, E_{Open} had 278 tuples, $E_{Specialist}$ had 350 and $E_{Consolidated}$ had 220.

The set that contains all sets E_h is named E_{Humans}.

3.4 Testing Inflation Prediction Performance

We will test two of the most common inflation prediction models: (1) **ARIMA** and (2) **LSTM**.

The goal is to check whether adding sentiment variables derived from LLM evaluations can reduce RMSE compared to using only historical inflation data and also if bias correction based on human evaluations can further improve performance.

3.4.1 Creating the Input Datasets

For each set of the power set of E_{Models}, except for the empty one, we will concatenate the tuples of the selected E_m sets into a single set named U_i.

For each U_i created, we will create j more tuples in the form (U_i, V_j), where V_j is one of the three human evaluation datasets in E_{Humans}.

For each tuple (U_i, V_j) created, we will create k more tuples in the form (U_i, V_j, eq_k), where eq_k is one of the equations to be used for bias correction later.

The tuple (U_i, V_j, eq_k) represents the sentiment evaluations from the selected LLM models combined with the human evaluation dataset V_j for bias correction using equation eq_k.

The possible equation forms for eq_k are: linear $(x + a)$, affine $(bx + a)$, quadratic $(cx^2 + bx + a)$, and cubic $(dx^3 + cx^2 + bx + a)$.

For each tuple (U_i, V_j, eq_k), we will create three different input datasets for inflation prediction models, each one of them will provide a list of tuples in the form of $(Inflation, Sentiment)$:

1. *Only Inflation (Baseline)*

   IPCA monthly (Series 433). The series can be accessed here: `https://api.bcb.gov.br/dados/serie/bcdata.sgs.433/dados?formato=json`.

   The sentiment variable will be set to 0 for associated inflation values.

2. *Inflation + Sentiment (Without Correction)*

   IPCA monthly (Series 433) + Sentiment variable created as an average grade per date of the evaluations in U_i (interpolated by cubic spline and fitted to the available IPCA dates)

3. *Inflation + Sentiment (With Correction)*

   IPCA monthly (Series 433) + Sentiment variable created as an average grade per date of the evaluations in U_i (interpolated by cubic spline and fitted to the available IPCA dates) corrected based on the bias measured from V_j.

   The correction process works as follows:

   First, both LLM sentiment scores from U_i and human evaluations from V_j are averaged by date and interpolated using cubic spline to create continuous daily time series.

   Then, we try to find a single set of parameters of the transformation equation eq_k that when applied to all dates individually minimize the mean squared error (MSE).

   The equation is applied per date with the variable x representing the average LLM sentiment score in that date, and the resulting value representing the bias-corrected sentiment score.

   The optimization uses gradient descent with the Adam optimizer (1000 epochs, learning rate 0.01) implemented in PyTorch.

   These optimized parameters are then applied to the equation to transform the LLM sentiment score for each individual date in U_i, producing bias-corrected values aligned with human judgment from V_j.

Finally, for each tuple (U_i, V_j, eq_k) created, we will have 3 new associated lists of tuples in the form of $(Inflation, Sentiment)$ which will be called IN_{ijkm}, where i is the LLM model combination used; j is the human evaluation dataset used for bias correction; k is the equation type used for bias correction; and $m \in$ {*Baseline*, *Without Correction*, or *With Correction.*}

The set that contains all sets IN_{ijkm} is named IN.

3.4.2 Running the Tests

Looking at the IN, it is obvious that this approach implies a lot of apparent unnecessary repetition of IN_{ijkm} datasets, since, for example, *Baseline* is the same for all tuples (U_i, V_j, eq_k).

While this is bad from a computational efficiency perspective, it provides a control for every experiment: *Baseline* should be a control *Without Correction* and *With Correction*, while *Without Correction* should be a control for *With Correction*.

For each IN_{ijkm} in IN, we will run both ARIMA and LSTM inflation prediction models using the respective dataset as input. Data was split 70/30 for training and testing.

We employed ARIMA with sentiment as exogenous variable using walk-forward validation, and LSTM with 5000 neurons trained with NAdam optimizer (learning rate 0.001, max 10,000 epochs, early stopping patience 10). Both models were evaluated using Root Mean Squared Error (RMSE).

In total, we conducted 36,792 tests: $(2^9 - 1)$ LLM combinations $\times$ 3 human datasets $\times$ 4 equation types $\times$ 3 dataset types $\times$ 2 models.

References

A. Author and B. Author. 2025. Placeholder article title. *Journal Name*, 1:1–10.