Out of the (Black)Box: AI as Conditional Probability *

Hui Chen[†], Antoine Didisheim[‡], Luciano Somoza[§]

March, 2025

Abstract

The core technology powering modern Large Language Models (LLMs) estimates the distribution of probable answers conditional on the prompt. Using a financial news and returns dataset, we find that these conditional probabilities are interpretable and contain valuable economic information. Conversely, measures of declared confidence used in the literature are opaque, structurally biased, unstable, and more model-dependent, indicating that LLMs cannot assess their own confidence. Using conditional probabilities, we analyze LLM biases and provide insights into the internal mechanisms driving model decisions. Our results indicate that conditional probabilities provide a reliable and transparent reflection of LLM priors, particularly for economic applications.

*We thank Dongyihai Peng, Simon Scheidegger, Hanqing Tian, Nitin Yadav, for their valuable comments.
[†]MIT Sloan and NBER. Email: huichen@mit.edu
[‡]University of Melbourne. Email: antoine.didisheim@unimelb.edu.au
[§]ESSEC Business School. Email: somoza@essec.edu

I. Introduction

The recent drastic improvements in the field of artificial intelligence (AI) have prompted a rapid integration into financial markets. AI in general, and LLMs in particular, are impacting trading (Cheng, Lin, and Zhao, 2024), driving firm-level innovation (Babina, Fedyk, He, and Hodson, 2024), and changing the economics of knowledge production (Abis and Veldkamp, 2024). However, as the performance of LLMs improves, so does their opacity. Researchers using LLMs must accept a steep trade-off between effectiveness and interpretability. Indeed, the generated text is the result of the interaction between two black boxes. The first is a model, called *transformer*, that estimates the conditional distribution of the next token.[1] The second is the *decoding strategy*, which decides which tokens to select, based on opaque and often undisclosed algorithms, compounding the lack of transparency of the transformer. Hence, while LLM generated texts appear full of meaning, their opaque generation process makes them challenging to use in academic research. In this paper, we argue that focusing on conditional probabilities, leaving aside all the additional opacity coming from the token selection algorithm, drastically increases transparency and interpretability. We show that these distributions are less noisy than point estimates, carry intuitive economic meaning, and can be used to understand the belief-formation process within LLMs.

The transformer model is composed of billions of parameters that are impossible to interpret at an individual level, but it nevertheless follows a well-defined and intuitive mathematical structure. More importantly, its output is a straightforward mathematical object: the distribution of the next token conditional on the context (i.e., prompt) provided to the model. For clarity, we refer to these conditional probabilities as *inner probabilities*, as they capture the inner beliefs of the transformer model. By contrast, the generated tokens are a point estimate of a distribution spanned by the interaction of the inner beliefs and the decoding strategy.

To test whether inner probabilities carry economic meaning, we use a sample of 10,000 financial news about individual firms and their associated contemporaneous stock returns. We ask the LLM to classify each news as positive or negative. We use OpenAI's API to obtain both the generated token and the inner probabilities from GPT-4, and use the inner probabilities, rather than the point estimates, for the classification. Next, we define

[1] A token can represent a single character such as "A" or 3, a fragment of a word like *ing* or *bel*, or even an entire word. Tokens serve as the basic units that AI models use to encode and processed text. One can think of tokens as the "building blocks" of language for the model—similar to how individual data points are fundamental in economic analysis. By decomposing text into these smaller units, the model can handle and generate language more efficiently, especially when dealing with complex words or varied linguistic structures. This tokenization allows the AI to take into account context, grammar, and meaning at a granular level, enabling more accurate predictions and interpretations.

inner confidence as how distant the inner probabilities are from a 50%-50% distribution. We measure accuracy by comparing the LLM classification with the market return on the same day. We observe a clear relationship between inner confidence and classification accuracy. Specifically, for predictions in the highest decile of inner confidence, the LLM achieves an accuracy rate of approximately 72.5%, significantly surpassing the baseline accuracy below 50% observed in the lower confidence deciles. A Logit regression where the dependent variable is a binary indicator of correct classification shows that these differences are statistically significant. This relationship and its magnitude persist even when isolating positive versus negative news, based on realized returns. This finding implies that when the LLM assigns high inner confidence to a classification, it is substantially more likely to match the actual market reaction, underscoring the economic relevance of inner probabilities as indicators of predictive precision.

A natural alternative to the inner-confidence measure defined above is to design a prompt to "ask" the LLM to generate first an answer and then generate a metric of confidence that can then be parsed out of the generated text (see, Bybee, 2023, e.g.,). We called this measure *declared confidence*. Unlike the inner confidence which captures the transformers actual "belief," the declared confidence captures, at best, the LLM's best "guess" at its own certainty. We demonstrate that this guess is: i) significantly biased by the decoding strategy, ii) sensitive to model and prompt choice, and iii) fairly uncorrelated to the inner beliefs.

Recall that LLM's generated tokens are samples drawn from a conditional distribution spanned by the transformer's estimated conditional probabilities and the decoding strategy. Hence, the distribution of the tokens representing declared confidence is functionally biased by the previously generated tokens. To illustrate the concept, assume a simplified scenario where we generate two tokens s_1, capturing the LLM's choice, and s_2, capturing the LLM's declared confidence in s_1. The first token is drawn from a distribution $p(s_1|P)$, where P is the prompt. The second token is drawn from $p(s_2|s_1, P)$. Hence, unlike the inner-confidence metrics, declared confidence is a function of earlier generated tokens. This is potentially problematic if, everything else being equal, $p(s_2|\text{“}A\text{”}, P) \neq p(s_2|\text{“}B\text{”}, P)$—that is, if the probability of the second token is biased by the draw of the first tokens. We show that this mechanical effect is empirically relevant and translates into a significant bias of the declared confidence, when faced with an economic problem. Controlling for prompt fixed effects, we show that the declared confidence is statistically significantly higher when the model randomly selected a positive prediction "A". This effect is robust to the sign of the realized return and the inner probabilities. Note that the inner confidence is mechanically not impacted by the choice of the first token, as the first token is a function of the inner probabilities and not the other way around. The instability of declared confidence is further

evidenced by model behavior under different configurations. In tests using GPT-3.5, declared confidence often performed no better than random chance, with results resembling a coin flip. Additionally, flipping the evaluation scale — defining declared confidence between "1" and "0" instead of "0" and "1" — greatly disrupts declared confidence scores, while inner probabilities remain relatively unaffected. This contrast shows that inner probabilities are more robust indicators, as they are less impacted by superficial changes in model prompts or scale reversals, unlike declared confidence, which is susceptible to distortions introduced by the decoding process.

Given the discussed bias and instabilities of declared confidence, it's perhaps not surprising that the declared confidence is only mildly correlated to inner confidence. In our experiments, the bottom twenty percent of declared confidence is associated with inner confidence significantly higher than that in the next two quintiles of declared confidence. And in a logit regression with both predictors, both the inner and declared confidence keep their statistical significance when jointly predicting the model's accuracy. Furthermore, unlike with inner-probabilities, the distribution of the declared confidence tends to cluster around similar numbers, which prevents grouping the answers into deciles of declared confidence. GPT-4 assigns exactly 0.6, 0.7, or 0.8 to 85% of its predictions, and GPT-3.5 assigns exactly 0.8 to 77% of its declared confidence. While expected, these results point to an interesting pattern of state-of-the-art LLMs: a lack of self-understanding.

When declaring a degree of self-confidence, LLMs are generating tokens out of a distribution of likely answers. This distribution is calibrated to mimic patterns observed on the training sample and to have a "not bland or repetitive" feel (Holtzman, Buys, Du, Forbes, and Choi, 2019). Indeed, to a large extent, the primary objective of LLMs can be understood as "passing the Turing Test". To reach this objective, the decoding strategy adds randomness and, as we've shown, biases to the inner probabilities of the LLMs. Hence, when generating a declared confidence following a news article and its own the primary goal of the LLM is to produce a declared confidence that appears human, as opposed to the most economically meaningful number. The inner-probability in contrast stands as a far more transparent, and economically meaningful measure of inner-beliefs. Declared confidence is akin to declared preference on a survey, inner-confidence is akin to measured behavior, or even a brain scan of the LLM.

Finally, we leverage inner probabilities to examine the internal mechanics of LLM decision-making through various empirical tests. These inner probabilities allow us to observe patterns and biases deep within the model, before the decoding process, offering a clearer understanding of the LLM. One notable finding from this analysis is that LLMs exhibit biases for

seemingly arbitrary choices, such as favoring the number "7" when asked to randomly select a number between "0" and "9". This result should not be surprising. LLMs are trained to produce text with a human-feel to it, not to be unbiased number generators. Similarly, when we design a game in which the rational strategy for the LLM is to generate the token "A" 50% of the time, independently of game history, we see that the inner-probabilities are highly biased toward "A" and strongly influenced by the game history. Once again, this pattern should not be surprising given the nature of LLMs. The LLM's response in the game, as in random number generation, reflects patterns and frequencies within the dataset, rather than an inherent understanding of the concepts involved. Still, these results should be of interest to economists as it suggests a lack of economic rationality at the micro level. This pattern is especially interesting as it stands at odds with patterns observed at the aggregate level. Indeed, Bybee (2023) showed that GPT can match existing expert survey and Fedyk, Kakhbod, Li, and Malmendier (2024) that it mimics human financial reasoning.

Finally, we use inner-probabilities to study the decision process of LLMs. We feed news articles to the LLM incrementally, presenting fragments of text one at a time. This approach allows us to track how the model's classification of news as either "positive" or "negative" changes as more information becomes available. The results highlight a surprisingly strong economic significance of inner probabilities. First, the inner confidence increases on average as the model is fed a larger portion of the article. Second, high inner confidence (measured over the full article) strongly negatively correlates with the number of times the LLM reverses its classification through the article. Finally, while the number of fragments is a good predictor of the model's accuracy, in a joint logit regression predicting accuracy, the number of sentences has no residual explanatory power over the model's accuracy.

Our main contribution to the literature is the introduction of inner probabilities and inner confidence as a novel method to enhance LLMs' interpretability. Unlike declared confidence, which is prone to biases and instability due to additional conditioning from the model's decoding strategy, inner probabilities provide a direct, stable reflection of the model's underlying confidence. By empirically demonstrating that inner probabilities are reliable indicators of classification accuracy in simple tasks involving economic mechanisms, our work provides a concrete alternative to declared confidence measures. We further contribute to the understanding of LLM decision-making by using inner probabilities to unblackbox model behaviors, revealing biases and dependencies in token generation. Our findings suggest that inner probabilities can enhance model transparency and provide a more interpretable metric for researchers, enabling more accurate and trustworthy applications of LLMs in economics and finance.

Our paper is related to the literature that studies and assesses applications of LLMs in economic and financial research. This literature is recent but fast growing (see e.g., Korinek, 2023; Eisfeldt and Schubert, 2024). More specifically, we contribute to two strands: the one using LLMs for classification purposes and the one using LLMs for simulating agent behavior.

While machine learning methods, like LDA, have been widely used in research for years (see e.g., Bybee, Kelly, Manela, and Xiu, 2020), applications of LLMs are relatively more recent. Notably, Chang, Dong, Martin, and Zhou (2023) use LLM to identify earning calls sentiment, Krockenberger, Saunders, Steffen, and Verhoff (2024) for covenants violations, and Caragea, Chen, Cojoianu, Dobri, Glandt, and Mihaila (2020) for patents classifications. We contribute to this growing literature by showing that inner probabilities can be used as a measure of classification accuracy and they are stable and easy to interpret.

LLMs are also finding applications in simulating agent behavior and expectations. Bybee (2023) introduces a survey of economic expectations formed by querying a LLMs. He finds the resulting expectations closely match major existing surveys. In the same spirit, Fedyk et al. (2024) investigates whether AI models accurately capture investment preferences across different demographics, shedding light on the models' ability to replicate nuanced human decision-making processes in financial contexts. Ashokkumar, Hewitt, Ghezae, and Willer (2024) tests the predictive capabilities of LLMs concerning the outcomes of social science experiments, finding strong results. Overall, this strand of the literature finds that LLMs perform well in capturing aggregate human choices and belief. Conversely, Ludwig and Mullainathan (2024) exploit the difference between human cognition and LLMs. They propose a systematic procedure to generate novel hypotheses about human behavior, which uses the capacity of machine learning algorithms to notice patterns that might not be obvious to humans.

In this context, we address the problem of interpretability of LLMs, which is a central theme in recent research. Korinek (2023) discusses the implications of generative AI in economics, emphasizing the necessity for methods that illuminate the internal reasoning processes of LLMs to enhance their reliability and acceptance in academic research. The black-box nature of these models presents challenges for researchers aiming to understand and trust their outputs, especially when using them to simulate agent behavior.

The rest of the paper is organized as follows: Section II provides a discussion of the LLM's opacity source and the role of inner-probabilities in increasing transparency, Section III discusses the methodology, Section IV presents the sample, Section V analyses the economic

significance of inner probabilities, Section VI compares inner probabilities with declared confidence, VII uses inner probabilities to shed light on LLM token generation process, and Section VIII concludes.

II. Understanding Inner Probabilities

A. *Unpacking LLMs: Transformer and Decoding Strategy*

Colloquially, the term LLM refers to the complete generating process that takes a prompt as input and yields generated text as output. This process is in fact composed of two separate parts, that are developed separately. The two parts are the *transformer model* and the *decoding strategy*. The transformer is a probabilistic model (Movellan and Gabbur, 2020) which estimates the next token's probabilities conditional on the context. The decoding strategy is a combination of rules and algorithms that determines the procedure to query the transformer in order to sample a generated text out of a complex combination of the transformer's conditional probabilities. The key takeaway of this paper is that by focusing only on the transformer, researchers can greatly enhance the transparency, replicability, and understanding of their LLM applications.

A transformer model is a type of large neural network that relies on self-attention mechanisms to capture dependencies in sequential data, enabling parallel processing and achieving state-of-the-art performance in natural language processing Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin (2017). In other words, it is a highly parameterized function that captures patterns in texts (see Section II.B for an overview). During the training, it receives samples of tokenized texts, with the last token of the sample masked. The model then learns to minimize the negative log-likelihood of the correct next masked token, i.e., it is trained to guess the last (masked) token:

$$\text{Loss} = -\sum_{t=1}^{T} \log p(y_t | y_1, y_2, \ldots, y_{t-1}; \theta), \tag{1}$$

where y_t is the last masked token and y_1 to y_{t-1} is the set of preceding tokens forming the context. The model iteratively adjusts its weights (parameters in ML lingo), striving to assign higher probabilities to the correct next tokens, thereby honing its predictive capabilities. Training the transformer means estimating a highly complex function capturing human language by guessing one word at a time.

In this paper, we'll use P to denote a prompt. Similarly, we denote s_i as the token

selected in the i^{th} position in the answer generated by the LLMs.

After training, the transformer model does not *generate* text autonomously. Rather, the output is a distribution of conditional probabilities for each possible next token, given the context. As an example, given the prompt "The cat is on the", a well-trained transformer should provide a set of conditional probabilities, among which the word "table" would have a relatively high probability, while inappropriate words like "stock" should be assigned a very low probability. Formally, for any prompt P composed of N individual tokens: $P = [p_N, p_{N-1}, \cdots, p_2, p_1]$, the transformer's output is:

$$p(s_1|P) = p(s_1|p_N, p_{N-1}, \cdots, p_2, p_1), \tag{2}$$

where we denote s_i as the token selected in the i^{th} position in the answer generated by the LLMs.

The role of the decoding strategy is to use the transformer model to select not one, but a series of tokens that will make the generated text look the most human-like.

A natural first attempt to translate the function $p(\cdot|P)$ into a generated text is to use the likelihood of the next token as a decoding objective. The decoding strategy would simply select the most likely token every time. Formally the first token is given by,

$$s_1 = \arg\max_{s_1} \left(p(s_1|p_N, p_{N-1}, \cdots, p_2, p_1)\right) \tag{3}$$

Then the second token is selected to maximize the conditional probability, taking into account not only the original prompt, but also the token just selected by the decoding strategy,

$$s_2 = \arg\max_{s_2} \left(p(s_2|s_1, p_N, p_{N-1}, \cdots, p_2, p_1)\right). \tag{4}$$

The process continues until some mechanism stops the text generation. While always selecting the most likely token is very intuitive, no major LLM uses it as a decoding strategy, because the result is bland and strangely repetitive (Holtzman et al., 2019).

To address these issues, researchers have developed a variety of decoding strategies.[2] A complete review and discussion of these many strategies is beyond the scope of this paper. Nevertheless, they can create a thick layer of opacity stacked on top of the transformer as:

1. For example, the beam search strategy (Holtzman et al., 2019) evaluates multiple possible sequences by selecting the most probable ones at each decoding step, rather

[2] e.g., greedy decoding and beam search (Sutskever, Vinyals, and Le, 2014), and sampling methods such as top-k (Fan, Lewis, and Dauphin, 2018) and nucleus sampling (Holtzman et al., 2019).

than focusing on the likelihood of individual tokens. It uses heuristics to limit the number of sequences considered, managing the search space more efficiently.

2. Decoding strategies can be chosen independently of the transformer model. One can change them independently.
3. As the final decoding strategy is made of a set of rules, heuristics, and algorithms, developers have many degrees of freedom when selecting the strategy.
4. Retraining a new transformer is an extremely costly operation (see, Dubey et al., 2024, e.g.,), by contrast tweaking the decoding strategy is akin to tweaking parameters in software.
5. Most of the big players in AI do not disclose their decoding strategies, and they update them far more frequently than their transformers (OpenAI's GPT, Alphabet's Gemini, Anthropic's Claude, etc.).

B. One (and a Half) Black Box

Many researchers are rightfully concerned with the opacity of LLMs and often refer to them as black boxes. We argue that the opacity is the product of the two distinct parts discussed in the last subsection– transformer and decoding strategy– but that their respective contributions are not the same.

Conceptually, transformer models are relatively easy to interpret. As summarized in Figure 1, the model takes as input the tokenized text and positional encoding,[3]. Next, the model encodes the relationship between every token in the prompt using *attention heads*. These special functional forms produce as output a matrix of dimension N by N, where N is the number of tokens in the context. Note that most state-of-the-art LLMs can process contexts of up to 128,000 tokens, making the attention heads potentially 128,000 by 128,000. Every off-diagonal entry of these matrices models the relationship between two tokens. The stacked output of these attention heads constitutes a latent representation of the prompt (or context) that is then fed to a neural network (often a single linear layer) in charge of predicting the likelihood of the next token.

What makes transformer models opaque is not their structure at the abstract level. At its core, it is simply a model that provides probabilities for the next token based on

[3]Positional encoding adds information about the position of each token in the sequence, enabling the model to capture sequential order. Using positions like 1, 2, 3, 4 can be problematic because they are unbounded and may lead to numerical instability in neural networks. Moreover, simple positional indices do not capture the relative distances between tokens effectively, whereas sinusoidal positional encodings provide a bounded and continuous representation that allows the model to generalize to sequences of different lengths (Vaswani et al., 2017).

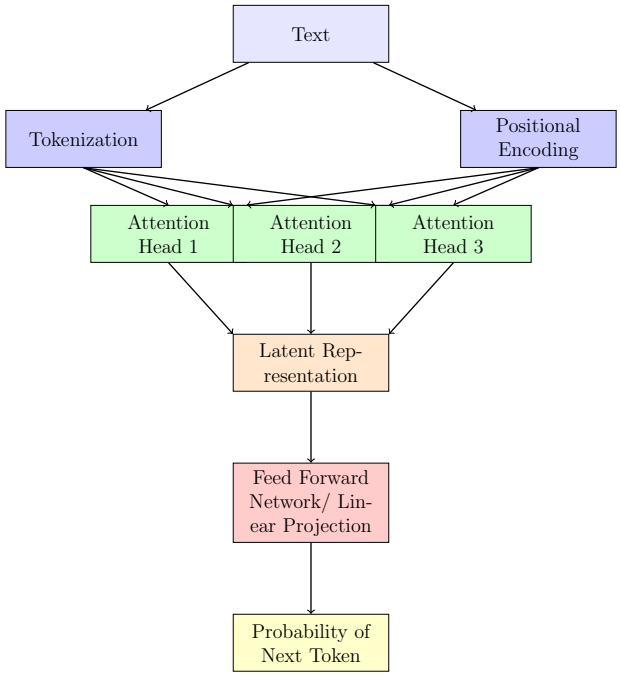

Figure 1. Transformer's Diagram
The diagram above provides an abstract representation of the computational flow in a standard transformer model with three attention heads. The diagram is simplified for clarity and intuitive understanding.

its training sample and the provided context. The opacity comes purely from the scale. Transformer models have not one, but multiple attention heads and a total of billions of parameters.[4] Such a scale makes direct interpretation of individual parameters impossible. Nevertheless, while the role of each parameter is opaque, the conceptual role of the attention heads, latent representation, and feed-forward network is fairly intuitive. Furthermore, and more importantly, the output has a clear and straightforward mathematical definition: the probability of the next token conditional on context.

The decoding strategy, on the other hand, amplifies the opacity coming from the transformer scale by an order of magnitude. As all decoding strategies rely on some iterative usage of the transformer to generate text, some of the probabilities will be functions not only of the prompt, but also of the text generated previously. As this text will be generated differently by different decoding strategies, it follows that only the first token's distribution will be conditional on the prompt. All other tokens will be conditional on the prompt *and* on the decoding strategy itself.

Furthermore, decoding strategies are typically less transparent than the transformer model. These strategies are rarely disclosed. Most commercially provided LLMs (Ope-

[4]LLaMA 3.1 largest model, performing at par with older versions of GPT-4, has 20 attention heads and a total of 405 billion parameters (Dubey et al., 2024)

nAI's GPT, Alphabet's Gemini, Anthropic's Claude, etc.) have little to no incentive to be transparent in their decoding strategies. First, they are a trade secret that makes it more difficult for competitors to match an LLM. Second, they create the illusion that their superiority comes from impossible-to-replicate secret transformer models. This aura of secrecy and perceived unreachable performances help companies' marketing and/or fundraising efforts. Hence, while the transformer model is a known but hard-to-interpret process, the decoding strategy is both unknown and hard to interpret.

C. Generated Text vs Inner Probabilities

To the best of our knowledge, LLM applications in economic research have focused on analyzing tokens generated by LLMs (see e.g., Bybee, 2023; Fedyk et al., 2024). Yet, generated text is only a point estimate from the distribution of probabilities generated by the transformer. In the case of complex generated text made up of multiple tokens, this conditional distribution is itself a complex combination of the conditional distribution of the next token given context by the transformer model, and the decoding strategy which decides how those distributions are truncated, combined, and drawn from.

Hence, we focus most of our experiments in the remainder of this paper on single-token answers. With one single token, the conditional distribution is simply the transformer's predicted distribution of the next token conditional exclusively on the prompt and the single token drawn from the distribution: $p(s_1|P)$ and s_1. While reducing the answer to single token may look like a strong constraint, one can design prompts that impart complex meanings to a single token.

Take Prompt 1 as an example. Given an article and its headline, we ask the model to assess the contemporaneous return of the firm. Specifically, we ask the model to produce a single token: "A" for a positive return or B for a negative one. For any given transformer model with a high enough capacity, the two most likely candidates for the next generated token will necessarily be "A" or B. Formally,

$$p(s_1 = A \mid P) \geq p(s_1 = X \mid P) \quad \text{and} \quad p(s_1 = B \mid P) \geq p(s_1 = X \mid P) \quad \forall X \neq A, B. \tag{5}$$

To remove the residual probabilities of inappropriate tokens (e.g., C or *The*) we define the inner probability of "A" as the normalized conditional probability of "A" over the set of acceptable answers:

$$\tilde{p}(s_1 = A \mid P) = \frac{p(s_1 = A \mid P)}{p(s_1 = A \mid P) + p(s_1 = B \mid P)}, \tag{6}$$

We stress that $\tilde{p}(s_1 = A \mid P)$ contains more information than $p(s_1 = A \mid P)$, a binary variable equal to 1 if the first generated token is "A". First, recall that $\tilde{p}(s_1 = A \mid P) > 0.5$ is not always equal to $p(s_1 = A \mid P)$. Indeed, state-of-the-art LLM models do not select the most highly likely token, but instead sample from the conditional distribution of the transformer models based on their decoding strategy (Holtzman et al., 2019; Dubey et al., 2024). Hence, the first generated token s_1 is a binary and noisy point estimate of the more nuanced distribution $\tilde{p}(s_1 = A \mid P)$. Second, the probability $\tilde{p}(s_1 = A \mid P)$ carries an implied measure of confidence.

As an illustrative example, assume $s_1 = B$ and $\tilde{p}(s_1 = A \mid P) = 0.99$. In this case, the model would have predicted "A" 99% of the time, implying a very high level of confidence, yet a tail event selected the 1% event and the generated text is B. Compare this outcome to a second illustrative example where $s_1 = A$ and $\tilde{p}(s_1 = A \mid P) = 0.6$. In this second scenario, the generated prediction aligns with the first moments of the inner probability, yet this prediction suggests a model far less confident in its choice of "A". The model would still select B 40% of the time. These simple examples serve to highlight the type of nuances that $\tilde{p}(s_1 = A \mid P)$ captures, which the single point estimate s_1 misses.

Concretely, in this paper we argue that inner probabilities are more suitable than point estimates for economic research, for four main reasons:

1. By using the inner probabilities of a single-token answer, the researcher minimizes the opacity of any given LLM model. We discussed in Section II.B that the opacity of LLMs comes from three sources: the scale of the transformer model, the secret decoding strategy, and the interaction of the decoding strategy with the transformer model. By forcing the answer into a single token and looking at the inner probabilities, the only opacity of the LLM comes from the scale of the transformer model, making the answer much more interpretable than the one going through the decoding strategy.
2. Using the inner probability to assign predictions (e.g., "A" iff $\tilde{p}(s_1 = A \mid P) > 0.5$) is mechanically less noisy than taking a random draw from this distribution.
3. The inner probability $\tilde{p}(s_1 \mid P)$ carries an implied measure of confidence. We show in Section V that these probabilities are not uninterpretable numbers within a black box, but rather they do carry clear economic significance.
4. Finally, we argue that the inner probabilities are useful for un-blackboxing exercises, by pinpointing when a model takes or reverses its output.

III. Methodology

A. *Default Models and Mixture of Experts*

For all our tests, we use the API provided by OpenAI. Our default model is *gpt-4-2024-08-06*, which we compare to GPT-3.5 using *gpt-3.5-turbo-0125*. We obtain the inner probabilities, $\tilde{p}(s_1 \mid P)$, directly from the API. While OpenAI has neither confirmed nor denied it, industry speculation suggests that the latest GPT models rely on Mixture of Experts (MoE) architectures—a concept that predates LLMs but has since been successfully applied in this context (Jacobs, Jordan, Nowlan, and Hinton, 1991; Artetxe, Bhosale, Goyal, Mihaylov, Ott, Shleifer, Lin, Du, Iyer, Pasunuru, et al., 2022).[5] During inference with a Mixture of Experts, a gating mechanism dynamically selects a subset of these experts (different transformer models) to process each input. Here, we define an expert as a distinct textual model that processes the prompt. In the context of LLMs, each expert corresponds to a different transformer model. Experts may differ in architecture, training data, or training procedures (Lo, Huang, Qiu, Wang, and Fu, 2024). This dynamic selection can introduce slight variations in the model's outputs, including the inner probability of the next token. However, this variation does not affect any of the experiments or arguments in this paper. It merely adds noise, which we mitigate by conducting sufficiently large tests.

B. *Parsing and Failed Prompts*

In the remainder of the paper, we present various tests using different prompts. To facilitate token parsing, we sometimes request that results be generated within special symbols, such as $[\cdot]$ or $< \cdot >$. This structure helps automate the processing of large volumes of generated text.

However, some prompts occasionally fail to produce parsable text. This can occur due to a rare token draw or a poor conditional distribution of the model. When this happens, we simply discard the observation. As a result, the total number of observations may not always sum precisely to 10,000 (the original number of runs). These discrepancies are minimal and unlikely to affect the significance of our results.

[5] See, e.g., https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked/

IV. Data

For our empirical analysis, we use a sample of news articles from Reuters.com. We randomly select a subsample of articles that meet the following criteria:

1. The news must relate to a single company. This ensures that the firm is directly affected by the news rather than mentioned only indirectly. We apply the data-cleaning procedure described in Chen, Kelly, and Xiu (2022).
2. The news must have a timestamp between 10 a.m. and 2 p.m. (ET) to increase the likelihood that contemporaneous daily returns reflect the news impact.
3. The news must contain between 100 and 5,000 characters.
4. The news must concern firms with a market capitalization of at least 10 billion USD.

From this population, we select 1,000 articles per year from 2013 to 2022, resulting in a total of 10,000 news articles. We supplement this subsample with daily returns and market capitalization data from the CRSP database. Table I presents the main summary statistics for the sample.

Table I. Summary Statistics
The table below presents the summary statistics of the dataset, including the mean, standard deviation, and decile cutoff points for article headline and body length (in characters), firm market capitalization (in billions), and contemporaneous returns.

		mean	std	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Article	Len. Headline	67.310	19.323	15	47	53	57	61	64	68	73	79	90	241
	Len. Body	1226	1032	120	296	426	573	687	819	1051	1473	1911	2769	4992
Stock	Market Cap.	125.298	220.632	10.010	13.627	18.589	25.839	35.803	49.561	70.805	106.420	184.252	284.830	2686.711
	Ret	0.002	0.036	-0.436	-0.028	-0.015	-0.008	-0.003	0.001	0.005	0.010	0.018	0.033	0.412

V. The economic significance of inner probabilities

In this section, we examine the economic significance of the inner probabilities. Specifically, we assess whether the inner probabilities assigned by the LLM are related to the accuracy of the **model's economic claim.** To do this, we use the sample of news articles and stock returns described in Section IV. We classify the news using Prompt 1, which provides the LLM's assessment of the news' impact on the firm's stock price. The prompt is designed to generate a single letter: A for positive news and B for negative news. This structure simplifies the extraction of the model's conditional distribution of responses, i.e., its inner probabilities.

```
Based on the news below, please say if you think the return for the stock
 {target} will be A (positive) or B (negative).

Please write only a letter A or B. Add no other formatting or bolding.

{headline}
{body}
```

Prompt 1. Classifying the news:
The prompt above is designed to extract the LLM's inner probabilities in a news classification task. The brackets $\{\cdots\}$ indicate dynamic inserts where we place the *target* (firm's ticker), *headline* (article's headline), and *body* (main text of the news article).

The generated token, $s_1 \in [A, B]$, is a noisy point estimate drawn from the hidden distribution $\tilde{p}(s_1 \mid P)$ of inner probabilities. If the LLM generates the token "A" as an answer, this token could come from a highly confident distribution (e.g., $\tilde{p}(s_1 = A \mid P) = 0.99$), a less confident distribution (e.g., $\tilde{p}(s_1 = A \mid P) = 0.6$), or even be an outlier from a distribution leaning toward negative sentiment (e.g., $\tilde{p}(s_1 = A \mid P) = 0.01$).

Thus, using the LLM's output—a point estimate—instead of the inner probability inherently introduces noise into the classification. A natural follow-up question is whether the LLM's inner probabilities carry economic significance. On average, are predictions with high inner probabilities more accurate than those with lower ones?

To address this question, we define a confidence measure that remains robust regardless of the forecast's direction:

$$C = |\tilde{p}(s_1 = A \mid P) - 0.5|, \tag{7}$$

As before, let $\tilde{p}(s_1 = A \mid P)$ denote the conditional probability that the first generated token is "A". Thus, a confidence measure of $C = 0.45$ implies that the LLM assigned 95% probability to either $s_1 = A$ or $s_1 = B$.

We extract the inner conditional probability $\tilde{p}(s_1 = A \mid P)$ for each news article in our sample. We classify news as positive if the probability of generating "A" as the first token is at least 0.5 ($\tilde{p}(s_1 = A \mid P) \geq 0.5$). Conversely, we classify news as negative (B) if $\tilde{p}(s_1 = A \mid P) < 0.5$.

We use the realized return as the *truth*, defining an event as positive if the daily close-to-close contemporaneous return is positive. We then measure accuracy as the share of

news for which the return's sign matches the LLM's point estimate. Finally, we divide the 10,000 news articles into deciles based on confidence, ranging from 1 (lowest confidence) to 10 (highest confidence).

Table II presents the results. Higher inner probabilities correspond to greater accuracy, suggesting that inner probabilities reflect the underlying economic reality rather than being an uninterpretable component of a black-box model. Moreover, the accuracy gains are substantial. At low confidence levels, the LLM is correct about 50% of the time, no better than a coin toss, indicating its inability to classify with certainty. In contrast, at high confidence levels, accuracy reaches 72.5%, showing that the LLM's inner probabilities align with its actual confidence.

Table II. Inner Confidence Economic Meaning

The table below uses the output of Prompt 1 to estimate sentiment and the associated inner probability for 10,000 randomly selected news articles. Sentiment is classified as positive when the probability of generating the first token as "A" exceeds 0.5 ($\tilde{p}(s_1 = A \mid P) \geq 0.5$). Inner accuracy is defined as $|\tilde{p}(s_1 = A \mid P) - 0.5|$, and its distribution is divided into 10 groups. For each group, the table reports the average accuracy, average return, and standard deviation of contemporaneous firm returns, conditional on positive or negative sentiment. It also provides the average accuracy conditional on positive sentiment. The final column lists the number of unique news articles in each confidence cluster. Deviations from a perfect 10-way split (1,000 articles per group) arise due to: (a) prompt generation errors that reduced the sample size and (b) clustering in the inner confidence distribution, which prevented exact equal splits.

	Accuracy	Average Return		Std Return		Accuracy		Count
Inner Confidence Group		$\tilde{p}(s_1 = A \mid P) \leq 0.5$	$\tilde{p}(s_1 = A \mid P) > 0.5$	$\tilde{p}(s_1 = A \mid P) \leq 0.5$	$\tilde{p}(s_1 = A \mid P) > 0.5$	$\tilde{p}(s_1 = A \mid P) \leq 0.5$	$\tilde{p}(s_1 = A \mid P) > 0.5$	
1	0.496	-0	0.002	0.027	0.027	0.506	0.485	994
2	0.521	-0.001	0.001	0.027	0.029	0.514	0.531	996
3	0.502	0.002	0.002	0.033	0.029	0.469	0.535	994
4	0.526	-0.001	0.002	0.029	0.029	0.519	0.531	994
5	0.531	-0.002	0.002	0.030	0.026	0.561	0.518	994
6	0.542	-0.007	0.004	0.032	0.029	0.586	0.525	991
7	0.617	-0.015	0.009	0.036	0.033	0.677	0.596	994
8	0.615	-0.013	0.011	0.044	0.039	0.640	0.606	996
9	0.676	-0.018	0.016	0.045	0.041	0.703	0.664	987
10	0.725	-0.031	0.019	0.053	0.042	0.762	0.709	1002

To assess the statistical significance of the trend shown in Table II, we estimate variations of the following regression:

$$y_i = \alpha + \sum_{k=1}^{10} \beta_k G_i(k) + \epsilon_i, \tag{8}$$

where $G_i(k)$ is a binary variable equal to 1 if news i belongs to confidence decile k.

Table III presents the results, where y_i equals 1 if the LLM's sentiment classification ($\tilde{p}(s_1 = A \mid P) > 0.5$) matches the sign of the realized contemporaneous return. Since the

dependent variable is binary, we use a Logit model. All deciles are individually statistically significant: buckets 1 to 6 correlate negatively with the LLM's accuracy, while buckets 7 to 10 correlate positively. Jointly, only buckets 4 to 10 are statistically significant and positively correlated.

Tables VIII and IX use the firm's contemporaneous return r_i as the dependent variable. Table VIII focuses on the subsample of news classified as positive by the LLM ($\tilde{p}(s_1 = A \mid P) > 0.5$), while Table IX examines news classified as negative ($\tilde{p}(s_1 = A \mid P) \leq 0.5$). In both cases, the results show strong significance. In each table, all but one decile are individually statistically significant. For news classified as positive (negative), higher-confidence deciles are associated with more positive (negative) returns on average. Appendix B presents results for subsamples split by positive and negative news.

Table III. Deciles of Inner Confidence

The table below presents the results of estimating equation 8 using a Logit model. The dependent variable is accuracy, a binary indicator equal to 1 when the LLM's highest inner probability aligns with the market reaction. $G_i(n)$ represents the decile of inner confidence. 7. For each estimation we report the point estimate of the parameters and its standard error (SE), and the sample size. *, **, and *** denote statistical significance at the 10%, 5%, and 1% level, respectively.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
$G_i(1)$	-0.355***										-0.0161
	(0.067)										(0.0634)
$G_i(2)$		-0.2431***									0.0844
		(0.067)									(0.0634)
$G_i(3)$			-0.3281***								0.008
			(0.067)								(0.0634)
$G_i(4)$				-0.2204***							0.1047*
				(0.067)							(0.0635)
$G_i(5)$					-0.1979***						0.1249**
					(0.0671)						(0.0636)
$G_i(6)$						-0.15**					0.1679***
						(0.0673)					(0.0638)
$G_i(7)$							0.1916***				0.4756***
							(0.0686)				(0.0652)
$G_i(8)$								0.1858***			0.4703***
								(0.0685)			(0.0651)
$G_i(9)$									0.4769***		0.7345***
									(0.0713)		(0.068)
$G_i(10)$										0.7325***	0.9671***
										(0.0739)	(0.0707)
Constant	True	True	True	True	True	True	True	True	True	True	False
Observations	9,942	9,942	9,942	9,942	9,942	9,942	9,942	9,942	9,942	9,942	9,942

VI. Declared confidence

A natural alternative approach is to directly ask the LLM about its confidence in a generated answer. Several studies have explored variations of this approach (Bybee, 2023; Fedyk et al., 2024). While declared confidence and inner probabilities may appear identical at first glance, the extraction method introduces two key sources of discrepancy. First, the LLM cannot directly observe its own inner confidence, as it simply predicts the next token based on its training data. When asked about its confidence, it must estimate it without direct access to the underlying probability distribution. Second, obtaining declared confidence requires the model to generate an answer first. This means that declared confidence is conditional not only on the original prompt but also on the generated response (see Section 2). This added layer of conditioning makes declared confidence less interpretable and more susceptible to algorithmic biases (discussed in Section II.B), further increasing its divergence from inner probabilities.

In this section, we evaluate the accuracy of declared confidence and examine the effects of these two sources of discrepancy. First, we isolate the impact of additional conditioning. If declared confidence depends on the generated response, it may be influenced by that response. To test this, we use the following prompt to construct a dataset of matched inner probabilities, answers, and declared confidence.

```
Based on the news below, please say if you think the return for the stock
 {target} will be A (positive) or B (negative).
Please start by writing only a letter A or B. Add no other formatting or
bolding.

Then write how confident you are in your choice by providing a number
between 1 (very confident) and 0 (not confident).
Please put the confidence number between square brackets [].

{headline}
{body}
```

Prompt 2. Joint declared and inner sentiment:
The prompt is a news classification exercise designed to extract matched pairs of the LLM's inner probabilities and declared confidence.
The brackets $\{\cdots\}$ indicate dynamic inserts where we place the *target* (firm's ticker), *headline* (article's headline), and *body* (main text of the news article).

Once we obtain a matched dataset of inner probabilities, point estimates, and declared

confidence using Prompt 2, we conduct the following experiment:

- We select 500 news articles where the inner confidence is $C \leq 0.2$, meaning the inner probability falls within $0.4 \leq \tilde{p}(s_1 = A \mid P) \leq 0.6$. These are cases where the LLM has low inner confidence.
- We rerun the prompt 100 times for each of the 500 news articles, generating a total of 50,000 observations.
- We estimate the following regression:

$$D_{i,j} = \gamma_j + \beta_0, A_{i,j} + \epsilon_{i,j}, \tag{9}$$

where $D_{i,j}$ represents the declared confidence, i.e, the generated token representing the model's confidence. $A_{i,j}$ is a binary variable equal to 1 if the first generated token is "A". This point estimate does not necessarily correspond to the most likely token based on the inner probability. The index i denotes the individual observation, while j refers to the specific prompt. The key coefficient, β_0, captures whether the LLM systematically declares higher confidence when the first generated token is "A", given the same inner probabilities. As a robustness check, we perform sample splits based on realized returns and inner probabilities. In all regressions, we cluster standard errors at the prompt level.

Table IV presents the results. The LLM consistently reports significantly higher confidence when the point estimate is "A", even in cases where the inner probability is actually higher for B (column 3) or when the market reaction is negative (column 5). This finding highlights a bias in declared confidence, introduced by its dependence on the point estimate. Unlike inner probabilities, which are conditional only on the prompt, declared confidence is also influenced by the generated response. In this case, the bias may stem from the LLM arbitrarily favoring the letter "A" over B or from a general tendency to express higher confidence when predicting positive outcomes. Regardless of the cause, this experiment empirically demonstrates the impact of additional conditioning.

Table IV. Identifying the Effect of Conditioning

The table below presents the results of estimating equation 9. The sample consists of matched observations of inner probabilities, point estimates, and declared probabilities, obtained following the steps described in the main text. The dependent variable is the declared confidence. The independent variable is $S_1 = A$, a binary indicator equal to 1 if the point estimate is A, as defined in equation 7. For each estimation we report the point estimate of the parameters and its standard error (SE), and the sample size. *, **, and *** denote statistical significance at the 10%, 5%, and 1% level, respectively.

	(1)	(2)	(3)	(4)	(5)
$A_{i,j}$	0.0337***	0.0432***	0.0216***	0.0367***	0.0307***
	(0.0038)	(0.0047)	(0.0048)	(0.0055)	(0.0052)
Prompt FE	YES	YES	YES	YES	YES
Sample	Full	$\hat{s}_1(A) > 0.5$	$\hat{s}_1(A) \leq 0.5$	$r_{i,t} > 0$	$r_{i,t} \leq 0$
Observations	49,999	22,944	27,055	25,000	24,999
R^2	0.568400	0.515400	0.599300	0.569500	0.567300

Next, we examine the distribution of declared confidence, computed using both GPT-3.5 and GPT-4, as shown in Figure 2. The figure reveals two key observations. First, the LLM exhibits a strong preference for integer values clustered around 0.6, 0.7, and 0.8. Nearly all declared confidences estimates are whole numbers, with only a few exceptions at 0.75 and 0.85 for GPT-3.5. This suggests that the tendency became even more pronounced in GPT-4. Second, the distribution of declared confidence differs sharply between GPT-3.5 and GPT-4, highlighting its strong dependence on the specific model used. This variability underscores the lack of consistency in declared confidence across model versions.

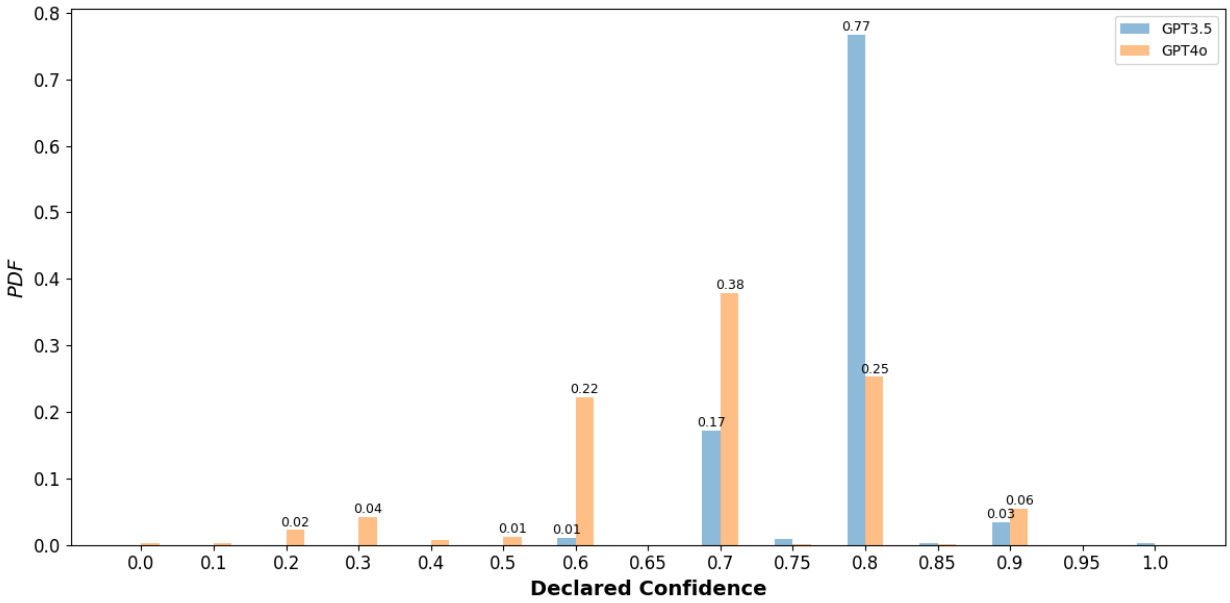

Figure 2. Distribution of Declared Confidence
The figure displays the distribution of declared confidence, defined as the LLM's self-assessed confidence in its output on a scale from 0 to 1. The blue bars represent the distribution for GPT-3.5, while the orange bars represent the distribution for GPT-4.0.

To evaluate the economic significance of declared confidence, we replicate the exercise from Section V, dividing observations into four buckets based on accuracy. While this approach works seamlessly for inner probabilities, it poses challenges for declared confidence due to its bias toward integer values, which results in one empty bucket. Table V presents the aggregate statistics.

Table V. Distribution of Declared Confidence
Using Prompt 1, we obtain inner confidence, point estimates, and declared confidence for the news in our sample. The table below reports accuracy levels for each quartile of confidence, comparing inner confidence and declared confidence for GPT-3.5 and GPT-4.

Metric	Confidence Group	Inner GPT3.5	Inner GPT4o	Declared GPT3.5	Declared GPT4o
Mean	1	0.501	0.494	0.517	0.479
	2	0.524	0.522	-	0.489
	3	0.547	0.581	-	0.543
	4	0.657	0.684	0.549	0.685
Count	1	2501	2533	2118	886
	2	2500	2489	-	2226
	3	2500	2488	-	3791
	4	2499	2490	7882	3097

For GPT-3.5, inner probabilities are strongly correlated with accuracy, ranging from a coin toss in the lowest confidence bucket to 66% accuracy in the highest. In contrast, declared confidence remains close to random, suggesting that the LLM is unable to reliably assess its own confidence. With GPT-4, both inner probabilities and declared confidence perform well, capturing the underlying economic reality. However, these results highlight that inner probabilities are more stable than declared confidence.

Nevertheless, the fact that GPT-4's declared confidence can achieve accuracy levels comparable to inner probabilities raises an important question: Does the LLM have an embedded model of itself, or are inner probabilities and declared confidence merely two unrelated estimators of the same underlying concept? While the LLM still predicts the next token without direct access to its previous inner probabilities, its strong performance may suggest that it has an effective internal model of its own functioning. To distinguish between these two interpretations, we estimate the following regression:

$$\text{Accuracy}_i = \alpha + \beta G(i) + \beta D(i) + \epsilon_i, \tag{10}$$

where Accuracy_i is a binary variable equal to 1 if the model accurately predicted the sign of contemporaneous return. $G(i)$ is a variable equal to 1 if prompt i falls into the first decile of inner confidence, 2 if it is in the second decile, and so on. Similarly, $D(i)$ captures the declared confidence decile of prompt i. Table VI presents the results. The explanatory power of both metrics declines when included together in the regression, yet they remain economically and statistically significant. This suggests that inner probabilities and declared confidence are not mere substitutes but capture distinct aspects of the model's confidence estimation.

Table VI. Deciles of Inner vs Declared Confidence

The table below presents the results of estimating equation 10. The dependent variable is a binary indicator equal to 1 when the prediction matches the market reaction to the news. $G(i)$ is an indicator variable equal to 1 if the inner confidence of prompt i falls in the first decile, 2 if it falls in the second decile, and so on. Similarly, $D(i)$ captures the decile of declared confidence for prompt i. For each estimation we report the point estimate of the parameters and its standard error (SE), and the sample size. *, **, and *** denote statistical significance at the 10%, 5%, and 1% level, respectively.

	(1)	(2)	(3)
$G(i)$	0.1068***		0.069***
	(0.0072)		(0.0099)
$D(i)$		0.1042***	0.0555***
		(0.0073)	(0.0101)
Constant	-0.2993***	-0.4133***	-0.4636***
	(0.0438)	(0.0526)	(0.0531)
Observations	9,958	9,958	9,958

To further support this finding, we analyze the distribution of declared confidence and compute the average inner probability for each interval. Figure 3 presents the results.

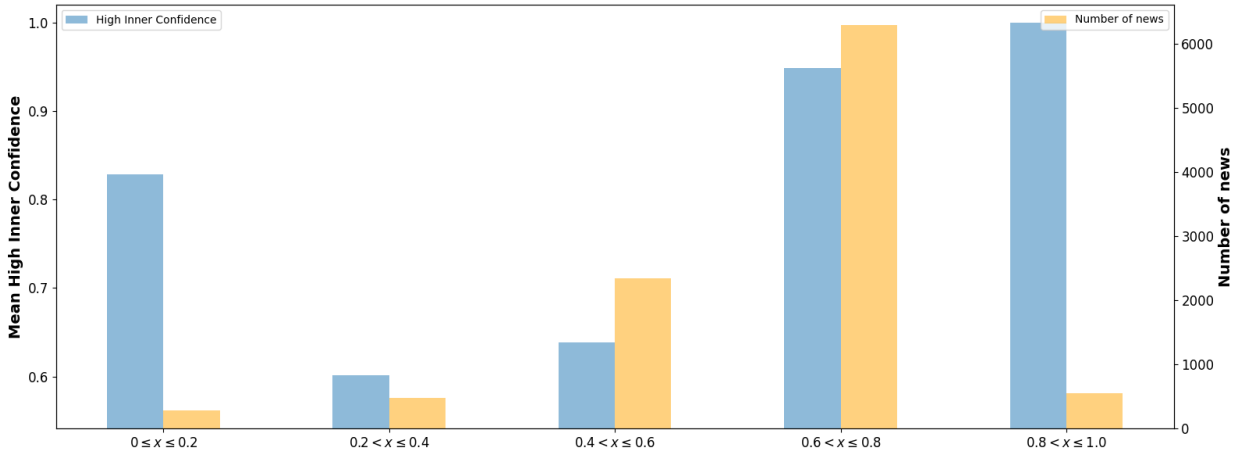

Figure 3. LLM Self Representation

The figure displays the distribution of declared confidence (orange bars) and the corresponding average inner confidence for each bucket (blue bars). The prompt instructs the LLM to classify news as positive or negative.

We observe that high declared confidence corresponds to a high average inner probability, and both decrease in a roughly linear fashion. However, a significant mismatch appears at

extremely low declared confidence levels, where inner probabilities remain relatively high. This pattern suggests a fundamental misalignment between declared confidence and inner probabilities, supporting the idea that they are two unrelated estimators of the same underlying object rather than evidence that the LLM has an accurate internal model of itself. In other words, not only can the LLM not directly observe its own internal functioning, but it also lacks a reliable model of it.

VII. Un(black)boxing LLMs

In this section, we use inner probabilities to shed light on the LLM's black box and help the reader understand its internal processes. First, we evaluate how closely the LLM's answers align with those of a rational agent. Second, we examine the process by which the LLM forms its answers.

For the first experiment, we instruct the LLM to generate a random number between 0 and 9 using the following prompt:

```
Give me a random number between 0
and 9.
```

```
Give me a random number between 9
and 0.
```

Prompt 3. Random Numbers:
The prompt above is designed to extract the LLM's inner probabilities when generating a single random token representing a number. The first prompt (left) requests numbers between 0 and 9, while the second prompt (right) requests numbers between 9 and 0.

Figure 4 presents the resulting inner probabilities, revealing a distribution that is far from uniform. A rational agent would assign equal probability to each number before making a random selection. In contrast, the LLM never selects the extremes, with most probabilities concentrated between 3 and 7—excluding 5. The number 7 has the highest inner probability at 75%. Notably, the LLM lacks an inherent understanding of numbers, treating them like any other token. This suggests that the high probability of 7 reflects its prominence in the training data rather than an inherent numerical preference. Moreover, the distribution is highly sensitive to the prompt; for instance, reversing the order of the interval's extremes produces a radically different distribution.

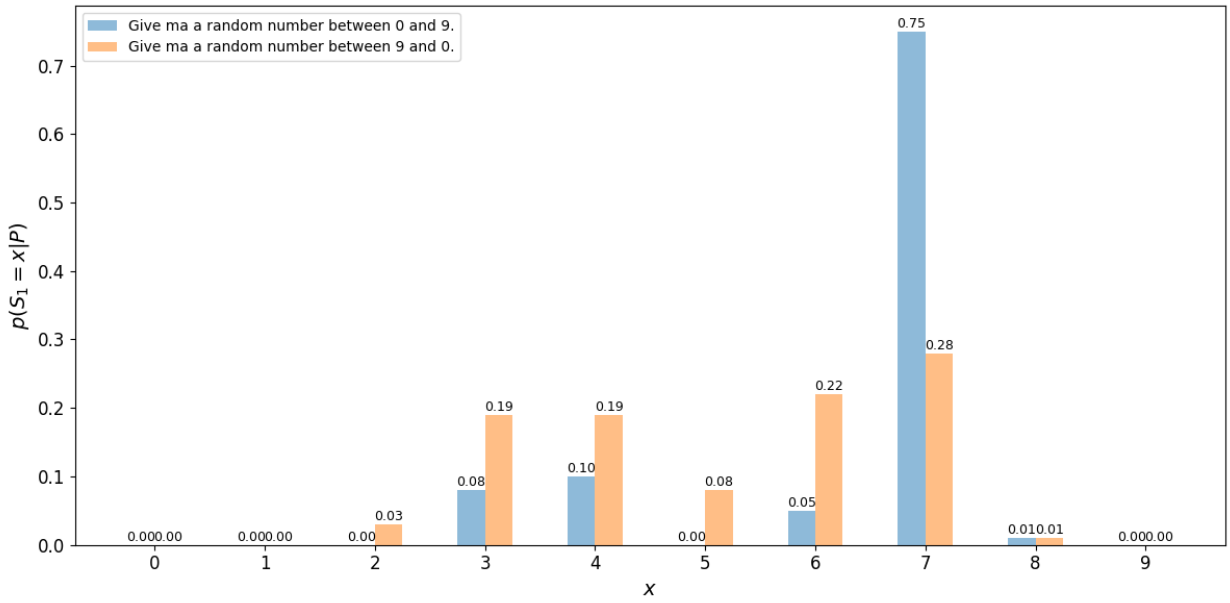

Figure 4. LLM Random Draw
The figure displays the distribution of inner probabilities resulting from Prompt 3. The LLM is instructed to choose a random number between 0 and 9.

Next, we design a game where the LLM generates a letter for the user to guess. The prompt also includes additional context: the number of times the game has been played and the LLM's previous answers. We use the following prompt:

```
Let's play a game. I write on a piece of paper A or B.
Then you write either A or B. If I guess right, you lose, if I guess
wrong, you win.
We've played this game {nb_game_played} times already.
Here is the history of your choices: {game_history}

I've now written my guess for the next round. Please write your choice.

Just write the letter you chose, nothing else.
```

Prompt 4.
Guess A/B Game:
The prompt above is designed to extract the LLM's inner probabilities in a game where the LLM generates a token, A or B, and the user attempts to guess it. The brackets $\{\cdots\}$ indicate dynamic inserts where we place the *target* (firm's ticker), *headline* (article's headline), and *body* (main text of the news article).

Figure 5 presents the results. A rational agent would commit to a strategy of playing both options with a 50% chance. In contrast, the LLM starts by selecting A with 96%

confidence but abruptly switches to B after three responses. This sudden shift suggests that rather than committing to a random stratefy, the LLM follows a more rigid pattern, likely influenced by implicit biases in its training data.

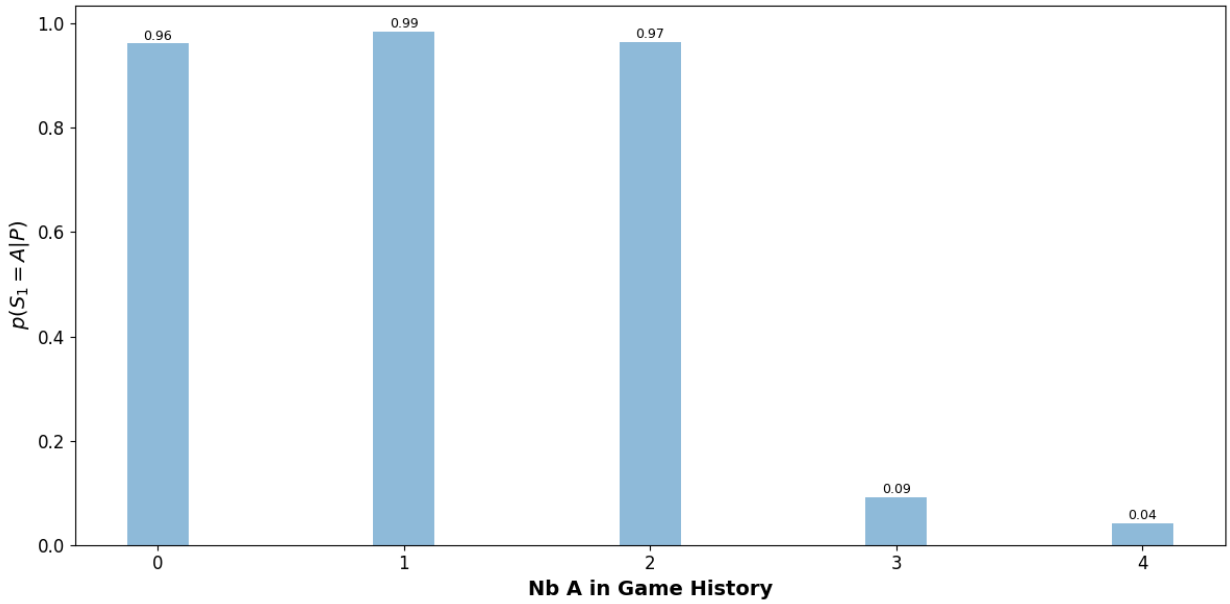

Figure 5. LLM Random Draw
The figure displays the distribution of inner probabilities resulting from Prompt 4. The LLM is instructed to choose A or B while trying to avoid selecting the same option as the user. The prompt also specifies how many times the LLM has already chosen A to observe how inner probabilities evolve. The number of previous A selections is denoted by $N\_A\_in\_Game\_History$.

To provide intuition on how LLMs make up their minds, we conduct another simple illustrative experiment. We ask the LLM to assess whether a piece of news is positive or negative, revealing one token at a time while extracting inner probabilities. To implement this, we use the following prompt:

Figure 6 presents the results. The interpretation is as follows: when assessing whether a piece of news is positive or negative, the LLM assigns a 22% probability of being positive to the incomplete sentence "Apple stock fluctuated today" As the sentence becomes more complete, the pattern of inner probabilities becomes more interpretable. Notably, certain words trigger abrupt shifts in probabilities. For instance, the word "groundbreaking" causes the probability of a "but" immediately reduces it to 38%. This experiment highlights how the LLM's answer generation process is characterized by sudden probability jumps rather than smooth, incremental adjustments.

```
    Based on the news below, please say if you think the return for the
    stock {target} will be A (positive) or B (negative).

Apple stock fluctuated today following the announcement of its
groundbreaking but costly new technology, leaving investors torn between
excitement for innovation and concern over the steep investments
required.
```

Prompt 5. Decomposed single sentence:
The prompt above is manually designed to include a change in sentiment in the middle of the sentence.

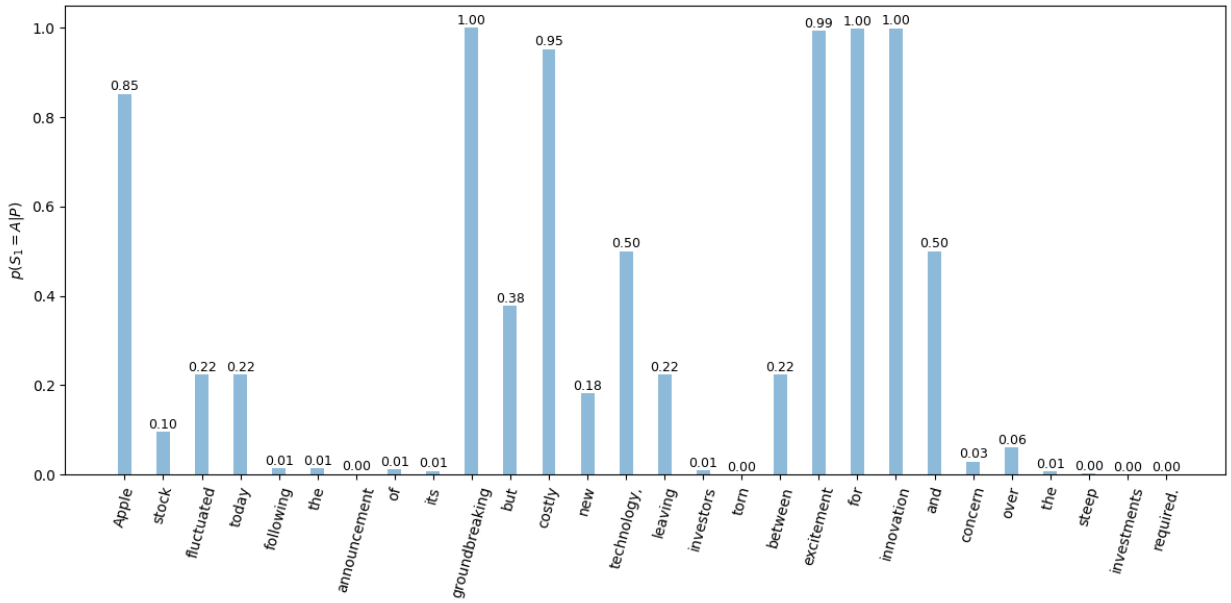

Figure 6. Word by Word Inner Probability
The figure displays the distribution of inner probabilities resulting from Prompt 5. The LLM is instructed to assess whether a piece of news is positive or negative and to extract inner probabilities. The news is provided to the LLM one token at a time to observe how the inner probabilities evolve with each token.

As this exercise suggests a pattern of radical probability changes, we further examine the answer generation process through a more comprehensive test. To improve computational efficiency, instead of prompting the LLM word by word as in Figure 6, we divide the input into fragments based on punctuation. We define punctuation as the following symbols: *.!?,:;*. For example, the sentence "The nimble cats, with tails twitching and eyes wide, leapt across the room, chasing after a bouncing ball" is split into four separate prompts:

- *The nimble cats*

- *with tails twitching and eyes wide*
- *leapt across the room*
- *chasing after a bouncing ball*

We randomly select 1,000 news articles where the headline and body combined contain between 10 and 20 fragments. This threshold ensures that the news samples are complex enough while keeping computational costs manageable. Using this sample, we conduct a classification experiment with Prompt 6.

```
Based on the truncated news below, please say if you think the return for
 the stock {target} will be A (positive) or B (negative).
Please start by writing only a letter A or B. Add no other formatting or
bolding.

Then write how confident you are in your choice by providing a number
between 1 (very confident) and 0 (not confident).
Please put the confidence number between square brackets [].

{truncated_body}
```

Prompt 6. De-Blackbox, Segment By Segment:
The prompt above is designed to extract the LLM's inner probabilities in a news classification exercise. This prompt reverses the A and B labels used in Prompt 1. The brackets $\{\cdots\}$ indicate dynamic inserts where we place the *target* (firm's ticker), *headline* (article's headline), and *body* (main text of the news article).

The resulting dataset contains the point estimate, inner confidence, and accuracy for each fragment. This allows us to track how often the model changes its classification as new information is introduced—specifically, when $\tilde{p}(s_1 = A \mid P)$ shifts from above 0.5 to below 0.5 (or vice versa) after adding a fragment. We refer to these shifts as reversals. For each news article, we count the number of fragments required to reach each reversal. Figure 7 illustrates the distribution of these changes. We find that in 65% of cases, the model does not reverse its classification at all. However, in 4% of cases, the LLM changes its classification three times while processing the news fragment by fragment.

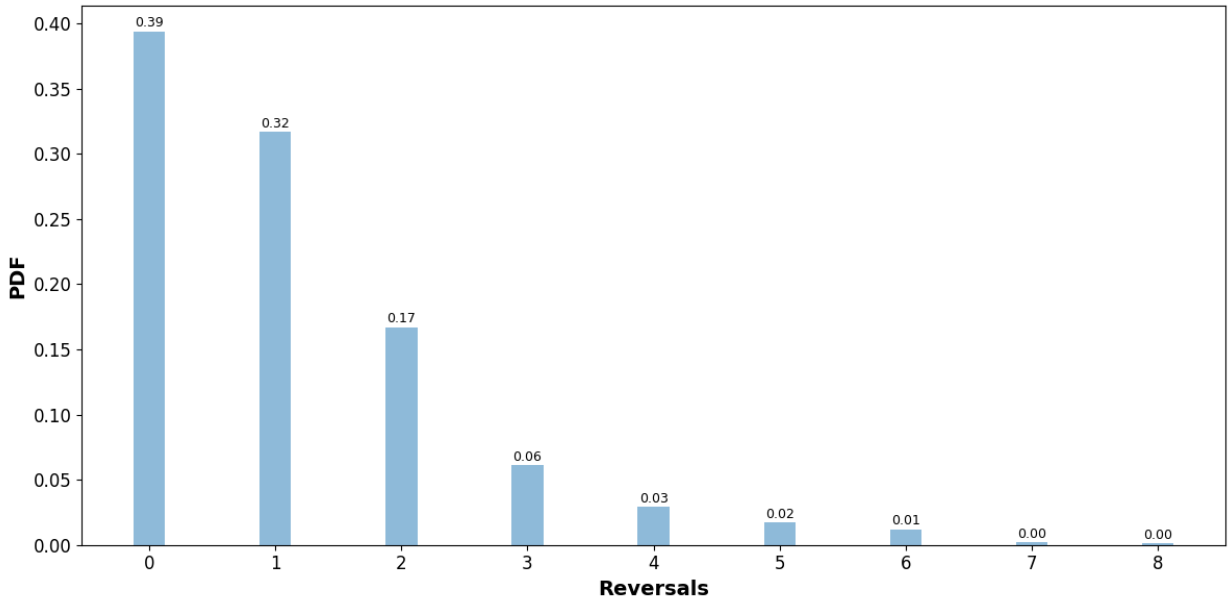

Figure 7. Reaching the Point Estimate
We divide each news article into fragments, defined as sections of text separated by punctuation, including commas and semicolons. The figure shows the number of fragments required for the LLM to reach its final classification, determining whether the news is positive or negative.

For each group of news articles with n reversals during fragment-by-fragment processing, we compute the average inner confidence. Figure 8 presents the results. We observe that the fewer the reversals, the higher the LLM's inner confidence. In other words, when the model maintains a consistent classification, it tends to express greater certainty. This pattern resembles human decision-making, where confidence is typically higher when conclusions remain stable despite new information.

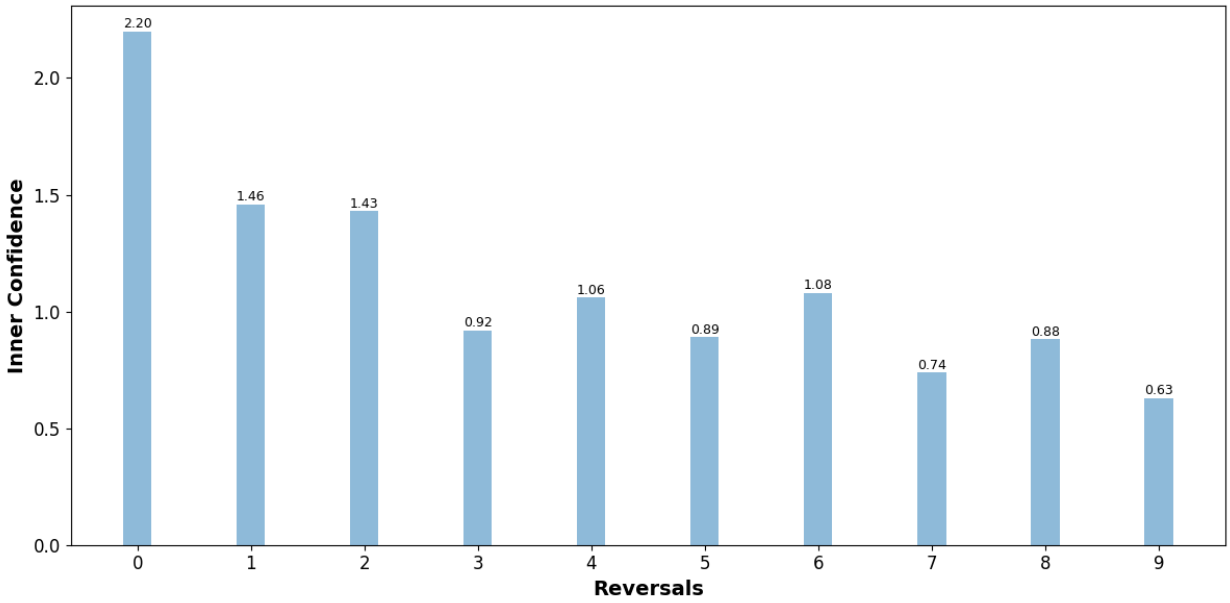

Figure 8. Reversals and Confidence in the Point Estimate
We divide each news article into fragments, defined as sections of text separated by punctuation, including commas and semicolons. The x-axis represents the number of fragments required for the LLM to reach its final classification, determining whether the news is positive or negative. The y-axis shows the average inner confidence.

Next, we examine the relationship between accuracy and the number of fragments processed. Using Prompt 6, we sequentially feed the LLM one fragment at a time, measuring both the average inner confidence of its point estimate and its accuracy. Figure 9 presents the results. The relationship is both linear and increasing, indicating that as the LLM receives more context, its confidence grows, and its classification accuracy improves. This suggests that the model effectively incorporates additional information to refine its predictions.

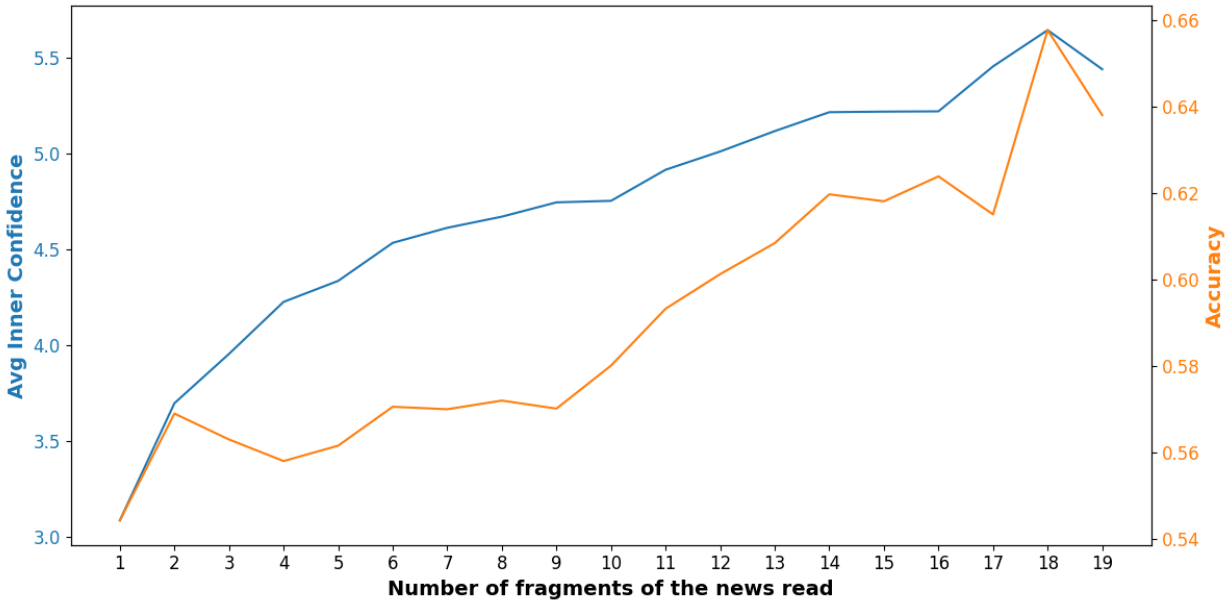

Figure 9. Amount of context, Inner Confidence, and Accuracy
We divide each news article into fragments, defined as sections of text separated by punctuation, including commas and semicolons. The LLM processes one fragment at a time, as indicated on the x-axis. The left y-axis represents inner confidence, while the right y-axis shows the accuracy of the point estimate.

This result suggests that both the length of the news piece and inner confidence predict accuracy. But how do they relate to each other? And which is the better predictor? To answer these questions, we estimate different versions of the following regression:

$$Accuracy_i = \sum_{j=2}^{10} \beta_j^C G_i(j) + \sum_{k=2}^{10} \beta_k^F Fragment_i(k) + \epsilon_i \tag{11}$$

where $G_i(j)$ is a binary variable equal to 1 if prompt i falls into confidence bucket j (with 1 being the lowest confidence level). Similarly, $Fragment_i(k)$ is a binary variable equal to 1 if prompt i includes the k^{th} fragment of text. Table XI in Appendix B presents the results. While both inner confidence and the number of fragments predict accuracy, the regression including both (column 3) shows that inner confidence accounts for nearly all the explanatory power. For readability, we report results up to 10 fragments, but extending the analysis to 20 fragments does not alter the findings.

VIII. Conclusion

In this paper, we examine inner probabilities in LLMs and their implications for transparency in research using these models. We argue that inner probabilities, unlike declared confidence, provide a more reliable measure of model accuracy. Because inner probabilities reflect the conditional probabilities assigned during token generation, they offer an unbiased representation of the LLM's decision-making process. In contrast, declared confidence is influenced by the model's decoding strategy, introducing opacity, uncertainty, and bias that reduce its interpretability and alignment with actual performance.

Using a dataset of financial news and stock returns, we empirically assess the economic significance of inner probabilities. We find that when inner probabilities are high, prediction accuracy improves significantly, closely aligning with actual market outcomes. Conversely, low inner probabilities correspond to random-like performance, highlighting the risks associated with low-confidence outputs.

A comparative analysis shows that while GPT-4 improves on GPT-3.5 in terms of declared confidence accuracy, both models exhibit systematic biases in declared confidence. For example, declared confidence values are often rounded integers, reflecting model-specific artifacts that undermine their reliability. In contrast, inner probabilities remain stable and perform consistently, even under changes in prompt structure.

We also explore whether LLMs can assess their own performance. Our findings indicate that they lack true self-awareness, as declared confidence does not strongly correlate with inner probabilities. Instead, declared confidence is influenced by the generated output itself, introducing bias and detaching it from the underlying probabilities that drive model decisions. This suggests that LLMs effectively "guess" their own confidence rather than internally assessing it, making declared confidence an unreliable measure of self-reflection.

Finally, we use inner probabilities to analyze the LLM's answer generation process, uncovering biases and patterns in token selection. Our results show that inner probabilities fluctuate based on prompt structure and context accumulation, shifting as new information is processed. Observing these changes provides insight into how LLMs prioritize responses and reveals the mechanisms behind perceived biases in their outputs. This approach offers a deeper understanding of model decision-making at a granular level.

Overall, our findings underscore the economic significance of inner probabilities and their potential to enhance LLM transparency, particularly in applications requiring interpretability of model confidence. By leveraging inner probabilities, researchers and practitioners can improve the reliability of LLMs in economic and financial contexts.

REFERENCES

Abis, Simona, and Laura Veldkamp, 2024, The changing economics of knowledge production, *The Review of Financial Studies* 37, 89–118.

Artetxe, Mikel, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, et al., 2022, Efficient large scale language modeling with mixtures of experts, in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 11699–11732, Association for Computational Linguistics.

Ashokkumar, Ashwini, Luke Hewitt, Isaias Ghezae, and Robb Willer, 2024, Predicting results of social science experiments using large language models, Technical report, Working Paper.

Babina, Tania, Anastassia Fedyk, Alex He, and James Hodson, 2024, Artificial intelligence, firm growth, and product innovation, *Journal of Financial Economics* 151, 103745.

Bybee, Leland, 2023, Surveying generative ai's economic expectations, arXiv preprint arXiv:2305.02823.

Bybee, Leland, Bryan T Kelly, Asaf Manela, and Dacheng Xiu, 2020, The structure of economic news, NBER Working Paper.

Caragea, Doina, Mark Chen, Theodor Cojoianu, Mihai Dobri, Kyle Glandt, and George Mihaila, 2020, Identifying fintech innovations using bert, in *2020 IEEE International Conference on Big Data (Big Data)*, 1117–1126, IEEE.

Chang, Anne, Xi Dong, Xiumin Martin, and Changyun Zhou, 2023, Ai democratization, return predictability, and trading inequality, Available at SSRN 4543999.

Chen, Yifei, Bryan T Kelly, and Dacheng Xiu, 2022, Expected returns and large language models, Available at SSRN 4416687.

Cheng, Qiang, Pengkai Lin, and Yue Zhao, 2024, Does generative ai facilitate investor trading? evidence from chatgpt outages, Evidence from ChatGPT Outages (June 21, 2024).

Dubey, Abhimanyu, et al., 2024, The llama 3 herd of models.

Eisfeldt, Andrea L, and Gregor Schubert, 2024, Ai and finance, Available at SSRN.

Fan, Angela, Mike Lewis, and Yann Dauphin, 2018, Hierarchical neural story generation, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 889–898.

Fedyk, Anastassia, Ali Kakhbod, Peiyao Li, and Ulrike Malmendier, 2024, Chatgpt and perception biases in investments: An experimental study, Available at SSRN 4787249.

Holtzman, Ari, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi, 2019, The curious case of neural text degeneration, arXiv preprint arXiv:1904.09751.

Jacobs, Robert A., Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton, 1991, Adaptive mixture of local experts, in *Advances in Neural Information Processing Systems*, volume 3, 633–640 (Morgan-Kaufmann).

Korinek, Anton, 2023, Generative ai for economic research: Use cases and implications for economists, *Journal of Economic Literature* 61, 1281–1317.

Krockenberger, Vanessa S, Anthony Saunders, Sascha Steffen, and Paulina M Verhoff, 2024, Covenantai-new insights into covenant violations, Working Paper.

Lo, Ka Man, Zeyu Huang, Zihan Qiu, Zili Wang, and Jie Fu, 2024, A closer look into mixture-of-experts in large language models, arXiv preprint arXiv:2406.18219.

Ludwig, Jens, and Sendhil Mullainathan, 2024, Machine learning as a tool for hypothesis generation, *The Quarterly Journal of Economics* 139, 751–827.

Movellan, Javier R, and Prasad Gabbur, 2020, Probabilistic transformers, arXiv preprint arXiv:2010.15583.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le, 2014, Sequence to sequence learning with neural networks, in *Advances in Neural Information Processing Systems*, volume 27, 3104–3112.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, 2017, Attention is all you need, in *Advances in neural information processing systems*, volume 30.

Appendix A. Additional Prompts

```
Based on the news below, please say if you think the return for the stock
 {target} will be B (positive) or A (negative).
Please start by writing only a letter B or A. Add no other formatting or
bolding. \

Then write how confident you are in your choice by providing a number
between 0 (very confident) and 1 (not confident).
Please put the confidence number between square brackets [].

{headline}
{body}
```

Prompt 7. Get Sentiment Inverted:
The prompt above is designed to extract the LLM's inner probabilities in a news classification exercise. This prompt inverts the A and B labels of Prompt 1. The squiggly brackets $\{\cdots\}$ indicate dynamic inserts where we place the *target* (firm's ticker), *headline* (article's headline), and *body* (main text of the news article).

Appendix B. Robustness checks

Table VII. Inner Confidence Economic Meaning with GPT 3.5

The table below uses the output of Prompt 1 to estimate sentiment and the inner probability of the estimation for 10,000 randomly selected news articles using GPT-3.5. Sentiment is classified as positive when the probability of generating the first token as A exceeds 0.5 ($\hat{s}_1(A) \geq 0.5$). Inner accuracy is defined as $|\hat{s}_1(A) - 0.5|$, and its distribution is divided into 10 quintile groups. For each quintile group, the table reports the average accuracy, average return, and standard deviation of contemporaneous firm returns, conditional on positive or negative sentiment. It also provides the average accuracy conditional on positive sentiment. The final column lists the number of unique news articles in each confidence cluster. Deviations from a perfect 10-way split (1,000 articles per group) arise due to: (a) prompt generation errors that reduced the sample size and (b) clustering in the inner confidence distribution, which prevented exact equal splits.

	Accuracy	Average Return		Std Return		Accuracy		Count
Inner Confidence Group		$\hat{S}_1(A) \geq 0.5$	$\hat{S}_1(A) < 0.5$	$\hat{S}_1(A) \geq 0.5$	$\hat{S}_1(A) < 0.5$	$\hat{S}_1(A) \geq 0.5$	$\hat{S}_1(A) < 0.5$	
1	0.477	0.004	0	0.030	0.031	0.440	0.517	1086
2	0.506	-0.001	-0	0.024	0.029	0.525	0.493	984
3	0.536	-0.002	0.003	0.034	0.031	0.553	0.526	985
4	0.538	-0.004	0.003	0.037	0.032	0.532	0.541	983
5	0.532	-0.003	0.004	0.029	0.028	0.535	0.530	984
6	0.575	-0.007	0.005	0.029	0.028	0.600	0.564	984
7	0.574	-0.010	0.004	0.036	0.028	0.611	0.559	984
8	0.607	-0.018	0.008	0.055	0.035	0.687	0.572	984
9	0.617	-0.020	0.010	0.043	0.038	0.701	0.586	984
10	0.714	-0.031	0.021	0.046	0.045	0.774	0.705	984

Table VIII. Accuracy and Inner Confidence for Positive News

The table below presents the results of estimating equation 8 using a Logit model. The sample includes only news to which the market reacted positively. The dependent variable is accuracy, a binary indicator equal to 1 when the LLM's highest inner probability aligns with the market reaction. $G_i(n)$ represents the decile of inner confidence, as defined by equation 7. For each estimation we report the point estimate of the parameters and its standard error (SE), and the sample size. *, **, and *** denote statistical significance at the 10%, 5%, and 1% level, respectively.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
$G_i(1)$	-0.0057***										0.0022
	(0.0017)										(0.0016)
$G_i(2)$		-0.0073***									0.0007
		(0.0017)									(0.0016)
$G_i(3)$			-0.0064***								0.0016
			(0.0016)								(0.0015)
$G_i(4)$				-0.0062***							0.0019
				(0.0015)							(0.0014)
$G_i(5)$					-0.0066***						0.0017
					(0.0014)						(0.0013)
$G_i(6)$						-0.0036***					0.0044***
						(0.0014)					(0.0013)
$G_i(7)$							0.0013				0.0087***
							(0.0013)				(0.0012)
$G_i(8)$								0.0039***			0.011***
								(0.0013)			(0.0012)
$G_i(9)$									0.0098***		0.0163***
									(0.0014)		(0.0013)
$G_i(10)$										0.0134***	0.0194***
										(0.0014)	(0.0013)
Constant	True	True	True	True	True	True	True	True	True	True	False
Observations	6,205	6,205	6,205	6,205	6,205	6,205	6,205	6,205	6,205	6,205	6,205
R^2	0.001800	0.002800	0.002400	0.002700	0.003500	0.001000	-0.000000	0.001200	0.008000	0.015400	0.034700

Table IX. Accuracy and Inner Confidence for Negative News

The table below presents the results of estimating equation 8 using a Logit model. The sample includes only news to which the market reacted negatively. The dependent variable is accuracy, a binary indicator equal to 1 when the LLM's highest inner probability aligns with the market reaction. $G_i(n)$ represents the decile of inner confidence, as defined by equation 7. For each estimation we report the point estimate of the parameters and its standard error (SE), and the sample size. *, **, and *** denote statistical significance at the 10%, 5%, and 1% level, respectively.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
$G_i(1)$	0.0072***										-0.0004
	(0.0017)										(0.0015)
$G_i(2)$		0.0071***									-0.0006
		(0.0016)									(0.0015)
$G_i(3)$			0.01***								0.0021
			(0.0017)								(0.0016)
$G_i(4)$				0.0057***							-0.0015
				(0.0019)							(0.0017)
$G_i(5)$					0.0046**						-0.0024
					(0.0021)						(0.002)
$G_i(6)$						-0.0005					-0.007***
						(0.0023)					(0.0021)
$G_i(7)$							-0.0087***				-0.0147***
							(0.0023)				(0.0022)
$G_i(8)$								-0.0066***			-0.0127***
								(0.0023)			(0.0021)
$G_i(9)$									-0.0126***		-0.0182***
									(0.0021)		(0.002)
$G_i(10)$										-0.0268***	-0.0312***
										(0.0021)	(0.002)
Constant	True	True	True	True	True	True	True	True	True	True	False
Observations	3,737	3,737	3,737	3,737	3,737	3,737	3,737	3,737	3,737	3,737	3,737
R^2	0.004700	0.004800	0.008700	0.002200	0.001000	-0.000300	0.003400	0.002000	0.008900	0.039800	0.068300

Metric	Confidence Group	Inner GPT3.5	Inner GPT4o	Declared GPT3.5	Declared GPT4o
Mean	1	0.503	0.513	0.616	0.754
	2	0.531	0.519	-	0.607
	3	0.575	0.580	0.528	0.536
	4	0.660	0.689	0.488	0.499
Count	1	2577	2500	508	1285
	2	2474	2500	-	2824
	3	2475	2499	6779	3023
	4	2474	2501	2713	2868

Table X. Distribution of Inverted Declared Confidence

Using Prompt 7, we obtain inner confidence, point estimates, and declared confidence for the news in our sample. The table below reports accuracy levels for each quartile of confidence, comparing inner confidence and declared confidence for GPT-3.5 and GPT-4.

	(1)	(2)	(3)
Fragments(2)	0.2778***		0.0042
	(0.0639)		(0.0692)
Fragments(3)	0.2533***		-0.0477
	(0.0638)		(0.0693)
Fragments(4)	0.233***		-0.1033
	(0.0637)		(0.0696)
Fragments(5)	0.2475***		-0.1043
	(0.0638)		(0.0698)
Fragments(6)	0.2842***		-0.0973
	(0.0639)		(0.0701)
Fragments(7)	0.2819***		-0.1126
	(0.0639)		(0.0701)
Fragments(8)	0.29***		-0.1127
	(0.0639)		(0.0703)
Fragments(9)	0.2824***		-0.1362*
	(0.0639)		(0.0704)
Fragments(10)	0.3232***		-0.0996
	(0.0706)		(0.0767)
$G(2)$		-0.0815	-0.0241
		(0.0571)	(0.0617)
$G(3)$		0.168***	0.2252***
		(0.0556)	(0.0599)
$G(4)$		0.1386**	0.197***
		(0.0553)	(0.0598)
$G(5)$		0.2378***	0.2999***
		(0.0548)	(0.0595)
$G(6)$		0.2095***	0.2704***
		(0.0541)	(0.0589)
$G(7)$		0.3394***	0.3983***
		(0.0541)	(0.0587)
$G(8)$		0.3677***	0.4289***
		(0.0542)	(0.0591)
$G(9)$		0.6451***	0.7046***
		(0.0558)	(0.06)
$G(10)$		1.2545***	1.312***
		(0.0634)	(0.0665)
Observations	13,493	13,493	13,493

Table XI. Inner confidence vs Number of Fragments

The table below presents the results of estimating Equation 11. The dependent variable is a binary indicator equal to 1 when the LLM's point estimate correctly classifies the news as positive or negative, meaning the classification aligns with the market return. $Fragment_i(k)$ represents the number of fragments included in the data point. $G(j)$ denotes the decile of inner confidence of the classification across all data points. For each estimation we report the point estimate of the parameters and its standard error (SE), and the sample size. *, **, and *** denote statistical significance at the 10%, 5%, and 1% level, respectively.