Using Transformers and Reinforcement Learning to Narrate the Business Cycle*

Vegard H. Larsen† Leif Anders Thorsrud‡

This version: December 15, 2025

Abstract

Building on recent advances in Natural Language Processing and modeling of sequences, we study how a multimodal Transformer-based deep learning architecture can be used to measure the business cycle and provide narrative attribution of its underlying structural drivers. The framework we propose combines (news) text and (macroeconomic) time series information using cross-attention mechanisms and incorporates important macroeconomic features, such as potential differences in data frequencies and reporting delays. In the process, we describe and document how the architecture can be used together with Reinforcement Learning to produce summaries of high-frequency news flows. Applied and tested on both simulated and real-world data out-of-sample, the results we obtain are encouraging.

JEL-codes: C45, C55, E32, E37

Keywords: Multimodal Transformer, Business cycle narration, News text analytics, Macro nowcasting

*Comments from Hilde C. Bjørnland, Peter Winker, Rune Sørensen, and seminar participants at BI Norwegian Business School, SSB, improved the quality of this paper considerably. This work is part of the research activities at the Centre for Applied Macroeconomics and Commodity Prices (CAMP) at the BI Norwegian Business School.

†Centre for Applied Macroeconomics and Commodity Prices, BI Norwegian Business School. Email: vegard.h.larsen@bi.no

‡Corresponding author. Centre for Applied Macroeconomics and Commodity Prices, BI Norwegian Business School, Nydalsveien 37, 0484 Oslo, Norway. Email: leif.a.thorsrud@bi.no

1 Introduction

> *"It would be nice to point to recognizable events, of the type that is reported by newspapers, as the source of economic fluctuations, rather than to residuals from some equations."* Cochrane (1994, p. 298)

A fundamental problem in macroeconomics is measuring the business cycle and identifying its underlying drivers. Early work, in the tradition of Burns and Mitchell (1946), focused foremost on measurement of empirical co-movement among individual economic variables and the division of business cycles into separate phases. More modern approaches analyze the question using a particular statistical model, with the vector autoregressions, introduced by Sims (1980), and later extensions incorporating factor structures or larger systems (Stock and Watson, 1989; Bernanke et al., 2005; Bańbura et al., 2010), having become standard frameworks. Within this tradition, however, the fundamental driving forces of the business cycle are residuals from the equations defining the assumed statistical model. Although successful, this has motivated developments of methods, starting already with Friedman and Schwartz (1963), that use text and historical sources to add narrative content to the statistical decompositions (Romer and Romer, 1989; Ramey, 2011; Stock and Watson, 2012; Antolín-Díaz and Rubio-Ramírez, 2018).

This paper takes up the same theme from a different perspective, leveraging Transformer-based neural architectures (Vaswani et al., 2017). These types of architectures underlie recent advances in Natural Language Processing (NLP) and Large Language Models (LLMs), but are designed for sequence data and increasingly used to analyze time series dynamics as well as multimodal learning using both text and time series variables (Wen et al., 2022; Xu et al., 2023).

In particular, we explore how a Transformer-based architecture can be combined with a user-specific economic view to build a model called the *Narrater*, whose purpose is to provide a reliable estimate of the business cycle and its underlying drivers by simultaneously modeling text and time series data via joint contextualized representations of the two data modalities. In the process, we show how the model can be used together with Reinforcement Learning (RL) techniques to produce summaries of the high-frequency flow of textual information. Accordingly, more than training a deep neural network solely to optimize predictive performance, we explore how RL and a multimodal Transformer architecture can be used as a structural narrative filter.

To fix ideas, the problem we study is based on a common and broadly defined hypothesis where text, here economic news, provides timely information about the most important events affecting the aggregate business cycle. Specifically, we assume that both the texts and economic time series are (partly) driven by the same underlying causes and events,

and that their joint contextualized representations are informative about a structural decomposition of the business cycle, akin to the historical shock decomposition commonly used to understand economic fluctuations, and the class and sentiment of the associated texts.[1]

These arguments do not imply that the Transformer architecture alone can identify the underlying sources of aggregate fluctuations. Identifying causal relationships from observational, aggregated data requires imposing (strong) assumptions. Thus, what we explore is how a Transformer-based architecture can filter multimodal information in line with a particular structural view. Fundamentally, this structural view must be encoded in the training data, which we simulate using assumptions about how news is selected and aligned with aggregate time-series dynamics. To mitigate in-sample circularity and assess usefulness, we strictly separate training, validation, and a quasi real-time out-of-sample evaluation on non-synthetic data. This prevents re-use of the same simulated sequences and allows testing whether the supervised narrative filter remains informative when confronted with unseen news and time series data. Importantly, this separation neither provides model-free identification nor validates the structural view; it only ensures that the reported performance is not an artifact of evaluating on training data.

In our benchmark configuration, we consider a setting with structural demand, supply, and noise components, where the business cycle is driven by the sum of the two fundamentals. The practical usefulness of this configuration is easy to motivate. From, e.g., an inflation-targeting central bank perspective having accurate business cycle estimates is important, but knowing that the business cycle is growing because of higher demand, rather than supply, might be equally important because the policy rate response will easily be very different in these two cases.

Still, the framework we propose is general and not tied to any specific structural implementation. It can, for example, be implemented under assumptions ranging from a single common business-cycle shock (e.g., Angeletos et al. (2020)) to much richer descriptions of macroeconomic fluctuations (e.g., Smets and Wouters (2007)). To demonstrate the framework's flexibility, we also evaluate two alternative, less-structural model configurations. One identifies news topics relevant to business-cycle fluctuations (as first proposed by Thorsrud (2018) and Larsen and Thorsrud (2019)). The other focuses on the model's

[1] See, e.g., Veldkamp (2011), Shiller (2017) and Chahrour et al. (2021) for information choice mechanisms where the news-media-economics linkages are described more explicitly within either a behavioral or rational expectations framework. The proposed framework can in principle also be used for narrative attribution of shocks. This would however require construction of training data where the mapping between economic variables and the texts is such that the latter contains unpredictable news. We find it easier to construct training data under the less restrictive assumption of predictability, as captured by the structural decomposition of the cycle.

ability to handle mixed-frequency data and directly generate extractive news summaries.

The design of the deep neural network we propose builds on the attention mechanism (Vaswani et al., 2017), and has an encoder-decoder structure using Transformer layers. This facilitates processing multimodal data and producing estimates of the underlying business cycle in a generative manner. To be precise, the model we design combines two encoders, one for time series of text data and one for multivariate economic time series. At a given time point, the language encoder inputs a text sequence, which is processed via a Bidirectional Encoder Representations from Transformers (BERT) architecture (Devlin et al., 2018), and outputs an embedding representation of the whole sequence. The BERT weights are shared across time points and potential time dependencies across the embedding representations are modeled using a single Transformer block. Similarly, the multivariate time series encoder consist of one Transformer block which outputs embedding representations of the multivariate time series data. The text and time series embeddings are then fused with the embedding representation of the underlying business cycle in the decoder using cross-attention layers. The end product is a network that simultaneously processes text and time series data and performs multi-task learning (Caruana, 1997). I.e., the model autoregressively outputs not only a structural decomposition of the cycle, but also performs narrative attribution by outputting the associated sentiment and class of the underlying news flow.

While the *Narrater* is large compared to traditional econometric models it is deliberately designed to be very parsimonious relative to typical LLMs used for text generation. This, in combination with using pre-trained language encoders and domain specific fine-tuning, facilitates efficient training and cost-effective usage. Besides capturing potentially complex and long-range dependencies within a sequence, the Transformer architecture enables these layers to capture non-linear features of the underlying data via non-linear activation functions and multi-head attention with softmax weighting. Moreover, the usage of the cross-attention layers permits modeling sequences measured at potentially different frequencies and with different lengths - features that are prominent when modeling macroeconomic data.

Based on similar cost and efficiency arguments the model is trained under the assumption of one text per time period. When confronted with multiple news items within the same period, we show that the model can be combined with RL to produce (extractive) summaries. In this setup, policy functions are estimated based on reward functions favoring low predictive loss, while optionally incorporating terms that encourage narrative priors and coherence. A simple special case is greedy selection, which requires no additional estimation or learning.

Empirical results from both simulation and real-world data demonstrate the model's

value. On synthetic validation data, the model achieves near-perfect classification accuracy and outperforms simple baseline methods and LLMs on classification and sentiment prediction. Permutation tests confirm that both modalities contribute to predictive performance, while manual audits on the simulated training data suggest strong alignment between model-generated labels and human judgment.

Ablation studies reveal that multi-task learning improves predictive performance, removing either encoder significantly reduces performance, while experimenting with settings determining the model's embedding dimensions suggests that larger embeddings encode more relevant information. Fine-tuning the BERT encoder alongside the other model parameters is also essential: performance degrades by up to 80% if the pre-trained parameters are kept fixed.

We test the *Narrater* on real-world Norwegian macroeconomic data and news from Dagens Næringsliv, Norway's leading business newspaper. In this quasi real-time experiment the model's business cycle estimates closely match those from traditional methods and the extracted sentiment is highly correlated with the cycle. Moreover, although the model and RL routine has to process thousands of news articles every quarter, the news summaries and classified news flow accord well with conventional (ex-post) human beliefs and narratives, while the model's nowcasting performance for GDP growth is comparable or slightly superior to that of a standard Dynamic Factor Model.

Finally, using the alternative model configurations, we demonstrate that the model performs well in identifying user-defined news topics and efficiently processes mixed-frequency information. Thus, the proposed framework appears to be easily adaptable and capable of addressing diverse user needs.

The remainder of the paper is structured as follows. Section 2 reviews related literature and situates our contribution. Section 3 presents the *Narrater* model, while Section 4 details the simulation-based training data and estimation procedure and Section 5 reports validation results. In Section 6 we describe how to use the model together with RL to generate extractive summaries and present application results as well as model extensions. Section 7 concludes.

2 Related literature and contribution

This study intersects with research in both economics and computer science. To our knowledge, it is the first to propose a multimodal Transformer-based architecture for economic analysis, and connects to three main strands of the economics literature.

First, we build on the growing body of work using NLP methods in macroeconomics (Baker et al., 2016; Hansen and McMahon, 2016; Hansen et al., 2018; Larsen et al., 2021;

Bybee et al., 2024). More recent studies incorporate Transformer-based models such as BERT for text classification and feature extraction (Liu et al., 2021; Gorodnichenko et al., 2023; Dell, 2024; Gambacorta et al., 2024). Unlike these approaches, which treat text as a standalone feature set, we adopt a fully multimodal framework in which text and time series data share a joint representation, estimated simultaneously through a Transformer architecture. This design enables supervised structural decomposition of the cycle alongside sentiment and class predictions in a multi-task setup, rather than treating text as a generic control. It also natively handles mixed frequencies and reporting lags through the cross-attention mechanism, and can be used to produce real-time extractive summaries that tie numerical movements to concrete news items.

Second, our work relates to the macroeconomic literature on signal extraction, business cycle measurement, and nowcasting. This literature often relies on large datasets (Stock and Watson, 1989; Evans, 2005; Giannone et al., 2008; Banbura et al., 2011), handles mixed-frequency inputs (Mariano and Murasawa, 2003; Aruoba et al., 2009), and increasingly incorporates non-linear or text-based features (Marcellino et al., 2016; Thorsrud, 2018; Burri and Kaufmann, 2020; Shapiro et al., 2022). While our model shares these features, it differs from traditional statistical filtering methods by enabling supervised structural decomposition via deep learning.

Third, we connect to the literature on narrative identification, which dates back to Friedman and Schwartz (1963) and uses historical documents to interpret and identify macroeconomic shocks (Romer and Romer, 1989; Ramey, 2011; Stock and Watson, 2012). Although we do not identify shocks per se, but rather propose a framework for narrative attribution consistent with a pre-specified structural view, our contribution builds on this tradition by automating the classification of narrative content and integrating it into a unified estimation framework that jointly infers the business cycle.

We do not want to argue, however, that the proposed model necessarily is better than existing technologies used in economics. Rather, we view our contribution as an exploration of how recent advances in language modeling and deep learning can provide a scalable framework for interpreting multimodal information through a consistent economic lens. Going forward, we envision numerous improvements and extensions to this type of model architecture. Beyond narrating the business cycle, the framework is potentially useful in any setting where high-frequency text (or other unstructured streams) co-evolves with numeric time series. By swapping in alternative targets and structural views, it can, e.g., be used to; monitor financial stability by attributing stress to liquidity, solvency, or policy shocks; track energy/commodity markets by tying price and quantity movements to supply disruptions or policy news; generate extractive summaries for event studies.

At a more general level our study speaks to large fields, foremost within computer sci-

ence, studying multimodal neural network architectures, LLMs for time series modeling, and so-called foundational models (FM). In terms of the former, Xu et al. (2023) survey multimodal learning with Transformers, highlighting their intrinsic advantages and scalability in modeling different modalities and tasks with fewer modality specific architectural assumptions than comparable methods. Early summation or concatenation of the embedding representations of the different data modalities is a common design principle. Here, to allow for potentially mixed-frequency data and different sequence lengths across modalities, we instead follow a so-called multi-stream design principle using cross attention. This is similar to in, e.g., Lu et al. (2019) and Tsai et al. (2019), but extend this line of research from vision and language modeling to time series and economics.

A related literature uses LLM architectures to explicitly build and train foundational models for time series analysis (Das et al., 2023; Woo et al., 2024; Jin et al., 2023). However, for predicting macroeconomic time series the recent study by Carriero et al. (2024) suggest that these types of FMs do no better than existing econometric models. Similarly, studies such as Gambacorta et al. (2024) and Dell (2024) demonstrate that specific fine-tuning of, e.g., BERT, delivers performance on par, or better, than LLMs in a range of (economic) domain specific classification tasks, e.g., classifying the sentiment of monetary policy speeches. This motivates a domain specific approach also for narrating the business cycle, which we pursue here in a multimodal setting.

Finally, we speak to a large literature in NLP studying text summarization (Zhang et al., 2024). Within this tradition exploring how multimodal input, and output, can improve summarization quality is a growing area of interest extending traditional text summarization to include other forms of data such as images, tables, and audio. Although the benchmark *Narrater* is not a summarization model per se, we demonstrate how its multimodal design features in combination with RL facilitate narrative attribution and summarization.

3 The Narrater

The *Narrater* is built using standard tools from the deep learning literature and in particular recent advances in terms of modeling sequences using Transformer-based architectures (Vaswani et al., 2017).

Figure 1 shows a high-level overview of the proposed model architecture. Two encoders - text and time series - map each modality to embeddings of a shared hidden dimension. The time-series decoder integrates these embeddings and autoregressively produces both the model output and the next-period decoder input. The text encoder uses a BERT language model. BERT-like models, introduced by Devlin et al. (2018), are pre-trained

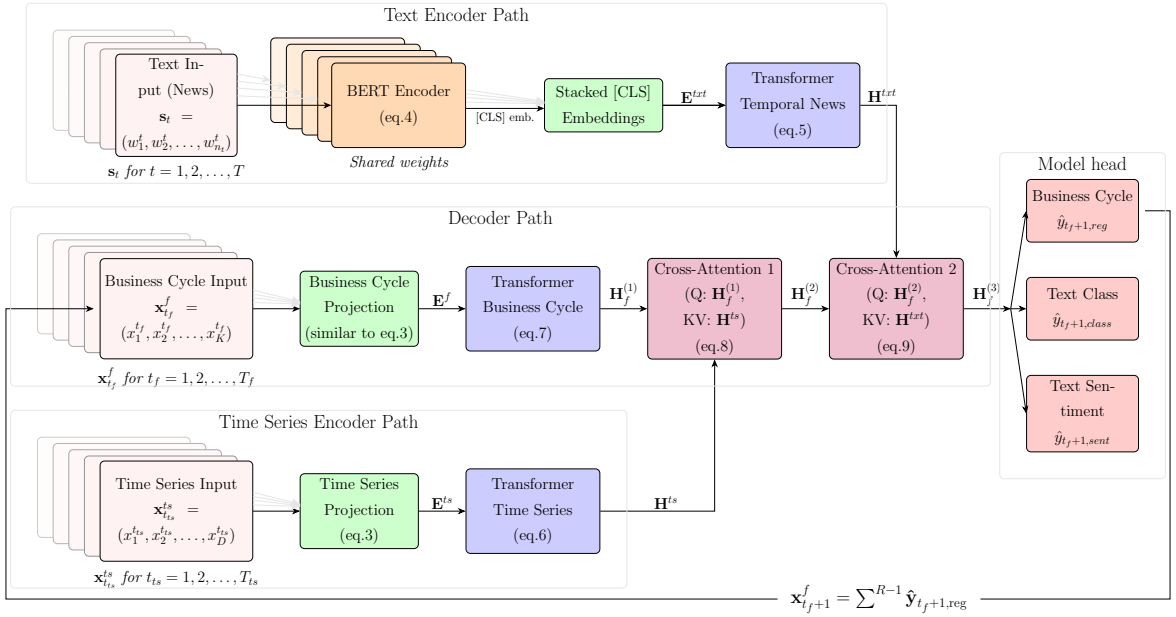


Figure 1. The *Narrater* model architecture.

on large-scale corpora and can be fine-tuned for a broad range of downstream tasks with minimal architectural modification, often achieving state-of-the-art or near–state-of-the-art results. From the BERT block we extract the sequence embeddings which are then forwarded to an independent Transformer layer to capture the assumption that text and time series are driven by common shocks and to some extent share dynamics - so text in period t may predict texts in later periods. For the same reason, independent Transformer layers are included in the time series encoder and decoder, and cross-attention blocks fuse the modalities into shared contextualized embeddings.

In sum, these choices aim to produce a network that processes text and time-series data simultaneously and autoregressively outputs both an estimate of the cycle and a narrative decomposition of the textual flow into its sentiment and class. Below we describe each part of the model in greater detail, with Section 3.5 highlighting particular modeling choices and benchmark specifications. Readers unfamiliar with Transformers and BERT might want to consult Appendix C before proceeding.

3.1 The text and time series encoders

Let a collection of T text sequences be denoted by $\mathbf{s}_t = (w_1^{(t)}, w_2^{(t)}, \ldots, w_{n_t}^{(t)})$ for $t = 1, \ldots, T$, where $w_i^{(t)}$ represents the i-th token of the t-th sequence and n_t is the length of that sequence. Then write the BERT architecture as a function that performs padding, masking, inserts special tokens, and transforms each input sequence $\mathbf{s}_t$ into a contextualized representation:

$$\mathbf{e}_t^{txt} = f_{\theta_1}^{BERT}(\mathbf{s}_t) \in \mathbb{R}^{N \times d}, \tag{1}$$

where d is the embedding dimension and θ_1 encompass all the embedding matrices (token and position) and weights of the Transformer encoder layers. Here we let $\mathbf{e}_t^{txt}$ represent the sequence embedding associated with the $[CLS]$ token. This is a common way of summarizing sequence content when using this type of architecture and serves as an efficient dimension reduction step in our application.[2] Next, let $\mathbf{E}^{txt} \in \mathbb{R}^{T\times d}$ define the $\mathbf{e}_t^{txt}$ vectors stacked for all $t \in \{1,\dots,T\}$. In this way, the same BERT structure and parameters (θ_1) are reused to model multiple sequences, ensuring that each sequence is encoded into a high-level representation space that is consistent across all T sequences.

Now, to allow for temporal relationships in the embedded text sequences we feed $\mathbf{E}^{txt}$ through a separate Transformer block:

$$\mathbf{H}^{txt} = f_{\theta_2}^{TF}(\mathbf{E}^{txt}) \in \mathbb{R}^{T\times d}, \tag{2}$$

where the $f_{\cdot}^{TF}$ function is a standard Transformer structure. Similarly, let $\mathbf{x}_{t_{ts}}^{ts} \in \mathbb{R}^D$ denote a D-dimensional vector of macroeconomic variables at time $t_{ts} \in \{1,\dots,T_{ts}\}$. Each vector is then first projected into a d-dimensional embedding space via:

$$\mathbf{e}_{t_{ts}}^{ts} = \mathbf{W}_{ts}\mathbf{x}_{t_{ts}}^{ts} + \mathbf{b}_{ts}, \tag{3}$$

where $\mathbf{W}_{ts} \in \mathbb{R}^{d\times D}$ and $\mathbf{b}_{ts} \in \mathbb{R}^d$ are learned parameters. The resulting embeddings are stacked into $\mathbf{E}^{ts} \in \mathbb{R}^{T_{ts}\times d}$ and serve as the input sequence to a Transformer, with positional encodings added in the usual manner:

$$\mathbf{H}^{ts} = f_{\theta_2}^{TF}(\mathbf{E}^{ts}) \in \mathbb{R}^{T_{ts}\times d}. \tag{4}$$

Accordingly, equations (1), (2), and (4) encode the input $\mathbf{s}_t$ for $t = 1,\dots,T$ and $\mathbf{x}_{t_{ts}}^{ts}$ for $t_{ts} = 1,\dots,T_{ts}$ into the embedding matrices $\mathbf{H}^{txt}$ and $\mathbf{H}^{ts}$.

3.2 The decoder

To decode this information and produce a narrative for the business cycle we build on the sequence-to-sequence type of architecture used for language translation in Vaswani et al. (2017), but extend and adjust this structure to process text and time series data.

For this purpose, let $\mathbf{x}_{t_f}^{f} \in \mathbb{R}^K$ be a time series measure(s) for the business cycle at time t_f, with $K \ll D$ in general and $K = 1$ here. Next, these K-dimensional observations are transformed into an embedding of dimension d following the same projections as in (3) for $t_f = 1,\dots,T_f$, and further processed as:

$$\mathbf{H}_{(1)}^{f} = f_{\theta_3}^{TF}(\mathbf{E}^{f}) \in \mathbb{R}^{T_f\times d}. \tag{5}$$

[2] One could alternatively have carried forward embeddings for all the tokens in the sequence. However, even for moderate time series lengths this would have had severe memory implications for subsequent Transformer-layers and is thus not something we have experimented with.

To accommodate the multimodal information in $\mathbf{H}^{txt}$ and $\mathbf{H}^{ts}$ we then apply two subsequent cross-attention layers. These are similar to regular self-attention (see (20) in Appendix C.1), but with the difference that the queries are defined as $\mathbf{Q}^f_{(\cdot)} = \mathbf{H}^f_{(\cdot)}\mathbf{W}_{Q^f_{(\cdot)}}$, and the keys and values as $\mathbf{K}^{txt} = \mathbf{H}^{txt}\mathbf{W}_{K^{txt}}$ and $\mathbf{V}^{txt} = \mathbf{H}^{txt}\mathbf{W}_{V^{txt}}$, or $\mathbf{K}^{ts} = \mathbf{H}^{ts}\mathbf{W}_{K^{ts}}$ and $\mathbf{V}^{ts} = \mathbf{H}^{ts}\mathbf{W}_{V^{ts}}$. In particular, we first construct:

$$\mathbf{H}^f_{(2)} = f^{CA}_{\theta_4}(\mathbf{H}^f_{(1)}, \mathbf{H}^{ts}) \in \mathbb{R}^{T_f \times d}, \tag{6}$$

and then:

$$\mathbf{H}^f_{(3)} = f^{CA}_{\theta_5}(\mathbf{H}^f_{(2)}, \mathbf{H}^{txt}) \in \mathbb{R}^{T_f \times d}, \tag{7}$$

where $f^{CA}_{\theta_\cdot}$ denotes the cross-attention mechanism and both functions are combined with a residual connection and layer normalization as in the standard attention layer.

Intuitively, since the queries for the cross-attention mechanism are derived from one sequence (the decoder representation), while the keys and values come from another sequence (the encoder representations), the cross-attention layers allow one sequence to attend to another sequence's latent representations, enabling the attending sequence to select and integrate the most relevant features from the other sequence. Figure 3 and the associated discussion in Section 3.5 provide a concrete example of this mechanism and how it is particular useful in settings with potentially mixed-frequency data or sequences of different length.

3.3 The model-heads

$\mathbf{H}^f_{(3)}$ contains the decoded and multimodal information provided by the original inputs $\mathbf{s}_t$, $\mathbf{x}^{ts}_{t_{ts}}$, and $\mathbf{x}^f_{t_f}$. We use $\mathbf{H}^f_{(3)}$ in combination with three separate model heads for inferring the state of the business cycle, and for classifying the type and sentiment of the text sequences. For a particular time index, the regression output is given by:

$$\hat{\mathbf{y}}_{t_f+1,reg} = \mathbf{W}_r\mathbf{h}^f_{(3)} + \mathbf{b}_r \quad \in \mathbb{R}^R, \tag{8}$$

where $\mathbf{h}^f_{(3)}$ is the transpose of the t_f-th row vector in $\mathbf{H}^f_{(3)}$ and the elements of $\hat{\mathbf{y}}_{t_f+1,reg}$ are discussed in subsequent sections.

Likewise, for text classification and sentiment estimation:

$$\hat{\mathbf{y}}_{t_f+1,class} = \text{softmax}(\mathbf{W}_c\mathbf{h}^f_{(3)} + \mathbf{b}_c) \quad \in \mathbb{R}^C, \tag{9}$$

and:

$$\hat{y}_{t_f+1,sent} = \tanh(\mathbf{W}_s\mathbf{h}^f_{(3)} + b_s), \tag{10}$$

where $\hat{y}_{t_f+1,sent}$ is a scalar and tanh is the hyperbolic tangent function, ensuring that the sentiment is bounded between -1 and 1. Thus, our model generates both continuous predictions ($\hat{\mathbf{y}}_{t_f+1,reg}$ and $\hat{y}_{t_f+1,sent}$) and a discrete probability distribution ($\hat{\mathbf{y}}_{t_f+1,class}$) from the same hidden representation $\mathbf{h}^f_{(3)}$.

3.4 Generative modeling

The original Transformer-based encoder-decoder architecture introduced by Vaswani et al. (2017) was generative in nature: given an input sequence (e.g., in English), the decoder produced the output sequence (e.g., in French) one token at a time, conditioning each prediction on the previously generated tokens. Decoder-only models follow a similar autoregressive mechanism.

The *Narrater* inherits this generative structure. Given encoder representations $\mathbf{H}^{\text{txt}}$ and $\mathbf{H}^{\text{ts}}$, along with an initial value $\mathbf{x}_1^f$, the model autoregressively generates $\mathbf{x}_2^f$, which then serves as input for the next time step. This autoregressive dynamic is not explicitly imposed by equation (8), but arises from our modeling assumption: $\hat{\mathbf{y}}_{t_f+1,\text{reg}} = \left(\hat{y}^{(1)}_{t_f+1,\text{reg}}, \hat{y}^{(2)}_{t_f+1,\text{reg}}, \ldots, \hat{y}^{(R)}_{t_f+1,\text{reg}}\right)$, where, as discussed further below, $\mathbf{x}^f_{t_f+1}$ is constructed as a sum over selected components of $\hat{\mathbf{y}}_{t_f+1,\text{reg}}$.

This design supports multi-step forecasting without requiring separate models for different prediction horizons. However, the effective forecasting window is bounded by the decoder's maximum sequence length, which defines its memory capacity.

3.5 Comments and benchmark specifications

Several model features and hyperparameter choices warrant discussion. First, the choices of R (number of structural components) and C (number of text classes) define the model's structural lens. In our benchmark (Section 4.1) we set $R = 3$ (demand, supply, noise), with demand and supply comprising the latent business cycle factor. This assumes the business cycle is driven by fundamentals while observables include measurement noise. Correspondingly, $C = 3$, classifying text into the same three categories. These choices are not inherent requirements, and the model can easily accommodate alternative specifications - as exemplified in Section 6.4.

Second, as noted in Appendix C.2, BERT comes in various sizes. We use $\text{BERT}_{\text{TINY}}$ (Turc et al., 2019), a compact BERT variant that trades off model size and computation for a slight accuracy reduction, enabling more efficient deployment while maintaining strong performance on many NLP tasks. In particular, our $\text{BERT}_{\text{TINY}}$ has 2 Transformer encoder layers, each with 4 attention heads and a 128-dimensional hidden size (about 4 million parameters).

Third, sequence length limitations in Transformer architectures impact the model's ability to capture long-term dependencies. For $\text{BERT}_{\text{TINY}}$, the maximum sequence length is $N = 512$, while for our model we set $T_{ts} = T_f = T = 40$, equivalent to 10 years of quarterly data. While longer contexts can be used, they incur higher computational costs. Similarly, the embedding dimension d, which could in principle vary across query, key, and

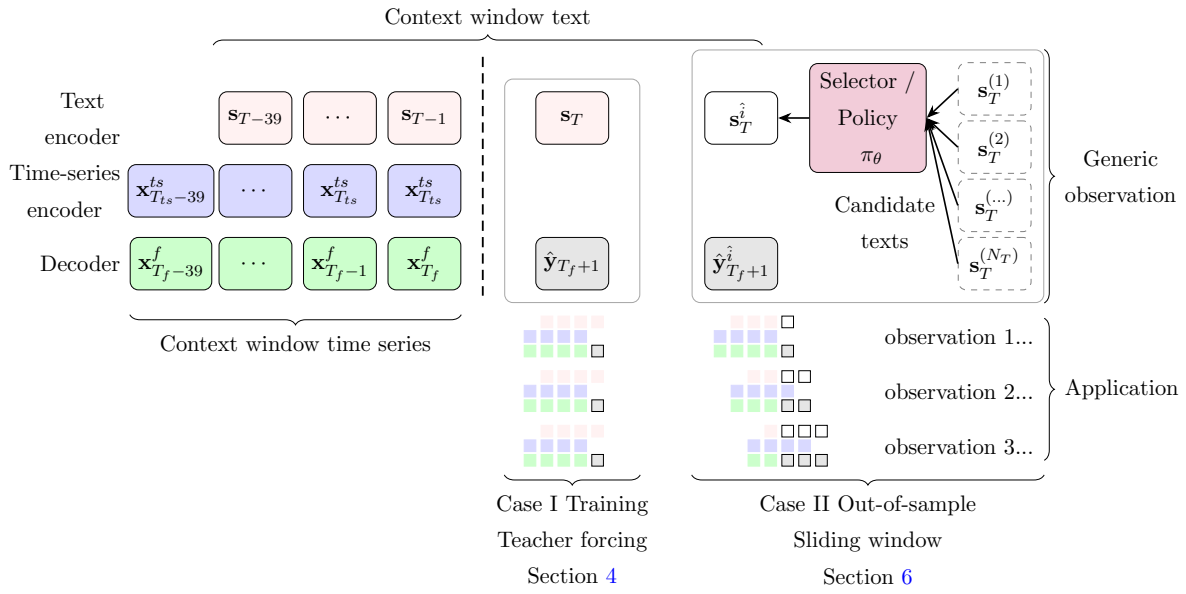


Figure 2. The *Narrater* information structure.

value matrices - is fixed at $d = 128$, consistent with $\text{BERT}_{\text{TINY}}$, using four attention heads. Increasing d could enhance representational capacity but at a significant computational expense. For this reason, we do not explore larger values, although we do report results for smaller d values.

Fourth, we use different time indices for different inputs to emphasize the model's capacity for handling mixed-frequency and asynchronous data via (cross-)attention and the padding and masking operations conventionally used in Transformer-based networks. Indeed, a key motivation for our multi-stream encoder-decoder design using cross-attention is its inherent suitability for handling mixed-frequency data, which is a common data property in macroeconomic contexts (Giannone et al., 2008; Mariano and Murasawa, 2003; Aruoba et al., 2009). In contrast, methods relying on concatenation or simple interactions often require synchronized sequences. While the benchmark specification assumes quarterly data throughout, an assumption relaxed in Section 6.4, we model the typical one-quarter lag in time series availability relative to text. Thus, $t_{ts} = t_f$ and $t = t_{ts} + 1$, reflecting the real-world delay in macroeconomic data releases. Figure 2 illustrates the assumed information structure used for training, and later, for out-of-sample applications and news summation.

Fifth, for the time series decoder and encoder input dimensions, we adopt $K = 1$ and $D = 8$, where D is determined by the structure of the training data (Section 4). Given $K \ll D$, our encoder-decoder model shares similarities with traditional autoencoders in machine learning (Goodfellow et al., 2016) and factor models in macroeconomics (Stock and Watson, 1989; Forni et al., 2000; Giannone et al., 2008), all of which reduce input to lower-dimensional latent representations. Unlike standard autoencoders, however, our model provides the decoder with the latent business cycle factor $\mathbf{x}^f_{t_f}$, making the learning

process more supervised. This structure is closer in spirit to adversarial autoencoders (Makhzani et al., 2015), which align the latent code distribution with a prior to ensure meaningful generative outputs.

Finally, attention in the Transformer and cross-attention layers can be either causal or bidirectional. We set all attention layers in (2), (4), (5), and (6) to causal, but allow bidirectional attention in (7). This design lets the model perform smoothing - akin to forward-backward filtering in econometrics - by incorporating future information into current state estimates.

Figure 3 illustrates the cross-attention mechanism and shows estimated attention scores (i.e., the softmax outputs from (20)) for one attention head in (2) and two in (7), based on a validation example. In Figure 3a, the scores reflect a clear autoregressive structure: text at time t attends only to prior periods. In contrast, the attention heads in (7) capture richer dynamics: Figure 3c emphasizes contemporaneous links between $\mathbf{H}^{f}_{(2)}$ and $\mathbf{H}^{txt}$, while Figure 3b captures both leading and lagging effects.

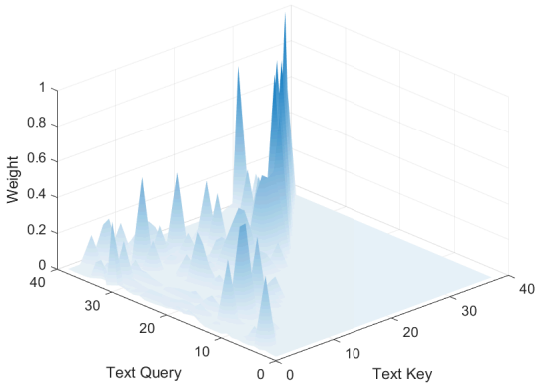
4 Training

Multimodal training data are common in many domains, but largely unavailable in macroeconomics. To address this gap, we generate synthetic training data via simulation, a standard practice in deep learning when real-world data is limited. In our case, simulation serves a dual purpose: it provides training data and facilitates structural interpretation by giving the user control on the assumed data-generating process. In other words, training on simulated data lets us instill a specific structural inductive bias into the *Narrater*, allowing it to act as a supervised narrative filter.[3] To avoid over-fitting and circular reasoning, simply assuming the conclusion we aim to test, we clearly split the observable data used as basis for the simulations into a training and validation sample and a sample containing genuinely unseen non-synthetic data.

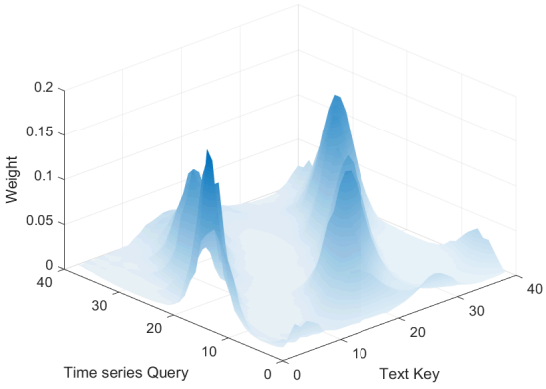
Below we summarize the key aspects of the data simulation process and the empirical data sources used to guide it. As the *Narrater* in principle is agnostic to the specific assumptions underpinning the simulations, we provide full methodological details in Appendix B. The subsequent sections outline the estimation procedure, including hyperparameter settings and the loss function.

[3]Inductive bias is the set of a-priori assumptions built into a learning procedure that steer the learner toward some hypotheses rather than others when generalizing from finite data, and is utilized in multi-task learning by introducing, e.g., auxiliary tasks (Caruana, 1997; Ruder, 2017). This stands in contrast to the term bias in the statistical-estimation sense because this is a numerical property of an estimator, but is closer in spirit to the use of prior information in Bayesian statistics because such information favors some hypotheses over others.

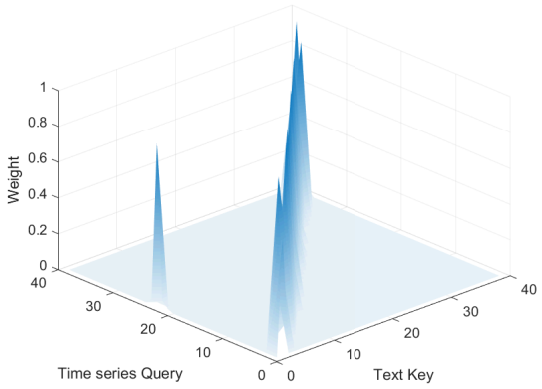
(a) Text to text
Attention head 1



(b) Text to time series
Attention head 1



(c) Text to time series
Attention head 2



(d) Manual audit performance

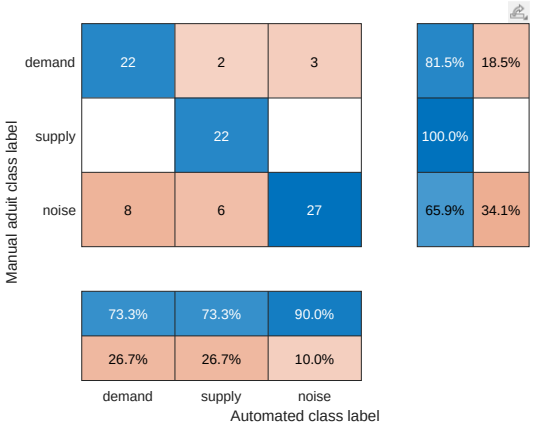


Figure 3. Figures 3a - 3c report attention scores (equation (20)), for one of the attention heads in (2) and two of the attention heads in (7) when processing one observation in the validation set. Figure 3d reports the manual audit's confusion matrix.

4.1 Simulation

To simulate economic time series data, we employ a Dynamic Factor Model (DFM). We acknowledge that the true data-generating process of the economy is undoubtedly more complex and non-linear than any DFM specification. However, the DFM is a well-established and effective framework commonly used in empirical business cycle analysis (Stock and Watson, 2016). DFMs can also be used for structural inference, and the log-linearized solutions of Dynamic Stochastic General Equilibrium (DSGE) models can be expressed as constrained DFMs.

Specifically, we assume that observable macroeconomic variables are driven by two latent factors, an aggregate business cycle factor and an inflation factor, along with idiosyncratic noise. These factors evolve in response to exogenous demand and supply

shocks. A demand shock drives prices and output in the same direction, while a supply shock drives them in opposite directions. The factors themselves are identified using the unit identification scheme discussed in, e.g., Bai and Wang (2014), letting in particular GDP growth load with one on the aggregate business cycle factor.

The DFM is estimated using the time series described in Section 4.3. After estimating the parameters, artificial datasets are simulated and split into training and validation sets. Each observation includes simulated macroeconomic variables $\mathbf{x}_{t_{ts}}^{ts}$, the latent factor $\mathbf{x}_{t_f}^{f}$, and its historical shock decomposition $\hat{\mathbf{y}}_{t_f+1,reg}$, for $t_{ts} = t_f = 1, \ldots, T_{ts} + 1$.

To link the simulated times series to text, a stratified sampling strategy inspired by Dell (2024) is applied. This aims to define the semantic territory of the category of interest in vector space without injecting unintended tone or framing. For this purpose we first define a set of benchmark key-word-based text sequences representing demand and supply narratives, and good or bad sentiment. In our application we think about demand narratives as associated with general expectations and demand, whereas supply narratives are written about in the context of productivity, efficiency and innovation. Good or bad sentiment is simply defined using positive and negative word sequences. Next, we obtain the embedding representation of these benchmark sequences and real-world text sequences (from the corpus described in Section 4.3) are labeled using their embedding representations and (cosine) similarity to these benchmarks. Noise is defined as texts that do not fit either the demand or supply benchmarks well, using a 90th percentile threshold as explained in Appendix B.

The class and sentiment of these texts are assumed to be influenced by the same shocks driving the DFM, consistent with studies portraying the media as "information intermediaries" (see, e.g., Nimark and Pitschner, 2019; Chahrour et al., 2021). However, we do not assume a deterministic mapping between shocks and texts, acknowledging the possibility of incomplete or inaccurate reporting. Instead, to construct $\mathbf{s}_t$, $y_{t_f,class}$, and $y_{t_f,sent}$ for training the text class is first sampled from a categorical distribution, where the class probabilities depend on the relative magnitude of the elements in $m_{t_f}^{(r)} = |y_{t_f,reg}^{(r)}|$ at each time point: $y_{t_f,class} \equiv c_{t_f} \sim \text{Categorical}\big(\boldsymbol{\pi}_{t_f}\big)$ *with* $\boldsymbol{\pi}_{t_f} = \big(m_{t_f}^{(c)}\big) / \sum_{r \in R} \big(m_{t_f}^{(r)}\big) \quad c \in R$. Let $\mathcal{B}_c$ be the set of labeled sequences for class c_t with known sentiments $\{s(b) : b \in \mathcal{B}_c\}$. Then, the class-consistent article whose sentiment is closest (in absolute deviation) to the target is chosen: $b_{t_f}^{\star} := \arg\min_{b \in \mathcal{B}_c} \big| s(b) - s_{t_f}^{\star} \big|$, *with* $s_{t_f}^{\star} = tanh(y_{t_f,reg}^{(c_{t_f})})$, and the text-based input at time t is simply the selected sequence $\mathbf{s}_t = \psi(b_{t_f+1}^{\star})$, where ψ maps the chosen article to the representation used by the model, with sentiment $y_{t_f,sent} = s(b_{t_f}^{\star})$.

Crucially, the associations learned by the *Narrater* will be shaped by the assumptions embedded in the chosen data-generation process, and alternative assumptions are discussed below. Still, while any of these assumptions may or may not reflect real-world

dynamics, overly simplistic or unrealistic structures are likely to result in poor model performance when applied to real data or evaluated through manual audits. To facilitate such evaluations, the DFM is estimated using data from 1986 to 2010, withholding the post-2010 period for out-of-sample evaluation. Similarly, text data from the post-2010 period is excluded from training to allow for independent testing on data never seen during training.

4.2 A manual audit and robustness

The simulation strategy described above is fast and simple and facilitates generating a large training data set in an automated manner. To assess whether the automated labeling procedure aligns well with human classification, we perform a manual audit. From the generated training data we randomly select 30 articles of each category and manually label them as demand, supply, or noise.

Figure 3d summarizes the results from the audit. The precision is above 70% for both demand and supply, and 90% for noise. In terms of sensitivity the performance is on average slightly better, with a perfect score for the supply category. On average the automated labeling procedure obtains an accuracy and F1-score of 79%.

For more concrete examples, Table 1 reports the first sentences of five articles and their manually and automatically assigned class. As illustrated, the labeling task is not always trivial, and different auditors might reach different conclusions.

In Appendix B.2 we perform a robustness experiment contrasting the embedding-based labeling procedure used here with a simpler Boolean key-word-based approach. This alternative labeling procedure does, however, only obtain an accuracy of 46% on the manual audit, suggesting that the embedding-based labeling procedure is reasonable.

To keep the analysis transparent and results easy to interpret we focus on the simulation baseline described above. Still, in Appendix B.3 we explore the effects of small perturbations to the text-time-series mapping. The results suggest that training on data generated under different assumptions - or directly on pooled data - could potentially further enhance model performance and robustness. We leave such explorations for future research.

4.3 Data

Consistent with a large literature on business cycle analysis, we utilize a broad set of macroeconomic time series to infer the latent business cycle. In the benchmark case the dataset includes Norwegian quarterly National Account Statistics: real GDP (GDP), real investments (I), and the unemployment rate (U). We also incorporate quarterly measures

Table 1. First sentences of five articles used in the automatically classified training set and the associated manual audit. The news texts are translated from Norwegian to English using GPT-4.5.

Manual	Automated	Text
Noise	Noise	*"A frequently repeated accusation against the government parties is that they have broken their election promises to give Norway a better school system. The accusation would have been correct if the Conservatives, the Liberal Party, and the Christian Democrats had promised during last year's election campaign..."*
Supply	Supply	*"The Norwegian postal monopoly is falling, and new international players are ready to claim their share of the market. Fierce competition, similar to that in the telecommunications market, is expected. The fall of the postal monopoly will lead to lower prices for consumers, a variety of new services, many new small companies that won't make money, and eventually postal bankruptcies..."*
Demand	Demand	*"The National Association of Business Economists (NABE) yesterday released a survey of 47 leading analysts, expressing that the American economy is facing a sharp downturn in the first quarter. Unemployment is expected to rise to around nine percent this year, and this recession is likely the worst..."*
Noise	Demand	*"Denmark should not expect special agreements on sensitive political issues in the Maastricht Treaty. "The EC cannot grant Denmark exemptions within defense cooperation and the economic and monetary union as long as we demand that future member states must adhere to the Maastricht Treaty,"..."*
Noise	Supply	*"By Friday, all production had already stopped at the small and medium-sized businesses along Greåkerveien, which runs parallel to the Glomma River just outside Sarpsborg. From the morning, there was hectic moving activity — saving whatever could be saved. By the afternoon, the business owners could do nothing but watch as the river rose minute by minute..."*

of consumer price inflation (CPI), house price inflation (HPI), the interest rate spread between short and long maturities (SPREAD), and household credit volumes (CREDIT). These variables, individually or jointly, have been shown to possess strong predictive power for business cycle dynamics (Estrella and Hardouvelis, 1991; Stock and Watson, 2002; Schularick and Taylor, 2012). Given the significance of oil for the Norwegian economy (Bjørnland and Thorsrud, 2016), we also include the oil price (OIL).

Data are transformed to emphasize cyclical components. Specifically, GDP, I, and CREDIT are log-transformed and differenced over eight quarters, following Hamilton (2018). For CPI, HPI, and OIL, we use the four-quarter log difference to approximate annual inflation rates.

The main text data source is a comprehensive corpus of articles from Dagens Næringsliv (DN), Norway's largest and most widely read business newspaper, and the fourth largest overall. The articles are sourced from Retriever's "Atekst" archive, covering all DN pub-

lications from the late 1980s through mid 2023.[4] The resulting dataset contains over 500,000 articles. For each, we retain the timestamp, title, and full text, with the article body serving as input to the model.

4.4 Estimation

The model is trained using Stochastic Gradient Descent with the Adam optimizer. We employ a batch size of 16 and an initial learning rate of 0.0005, which is halved every second epoch to facilitate convergence.

All model parameters, θ_1 through θ_5, along with weight and bias matrices $\mathbf{W}_j$ and $\mathbf{b}_j$ for $j \in \{ts, r, c, s\}$, are jointly estimated. The $\text{BERT}_{\text{TINY}}$ encoder (θ_1) is initialized with pre-trained weights (Turc et al., 2019), while the remaining parameters are initialized using the Glorot scheme (Glorot and Bengio, 2010). To prevent excessive updates to the pre-trained encoder, its learning rate is set to one-fifth of the rate used for the other model components.

The benchmark model outputs three predictions, regression, sentiment, and classification targets, and is trained to minimize a weighted composite loss function:

$$L_{\text{total}} = \omega_r \, L_{\text{r}}\big(\hat{\mathbf{y}}_{reg}, \mathbf{y}_{reg}\big) + \omega_s \, L_{\text{s}}\big(\hat{\mathbf{y}}_{sent}, \mathbf{y}_{sent}\big) + \omega_c \, L_{\text{c}}\big(\hat{\mathbf{y}}_{class}, \mathbf{y}_{class}\big), \tag{11}$$

where predictions are evaluated for $t_f = 2, \ldots, T_f + 1$. Mean squared error (MSE) loss is used for both the regression and sentiment tasks:

$$L_{\text{r}}\big(\hat{\mathbf{y}}_{reg}, \mathbf{y}_{reg}\big) = \|\hat{\mathbf{y}}_{reg} - \mathbf{y}_{reg}\|_2^2, \quad L_{\text{s}}\big(\hat{\mathbf{y}}_{sent}, \mathbf{y}_{sent}\big) = \|\hat{\mathbf{y}}_{sent} - \mathbf{y}_{sent}\|_2^2,$$

and categorical cross-entropy is used for classification:

$$L_{\text{c}}\big(\hat{\mathbf{y}}_{\text{class}}, \mathbf{y}_{\text{class}}\big) = -\sum_{c=1}^{C} y_{\text{class}}^{(c)} \log\big(\hat{y}_{\text{class}}^{(c)}\big).$$

Loss weights are set to $\omega_r = 0.5$ and $\omega_s = \omega_c = 0.25$, reflecting equal prioritization of the regression and text classification tasks. These weights can be tuned as hyperparameters to optimize predictive performance.

As described in Section 3.5, the model autoregressively constructs output sequences. However, during training, we use teacher forcing: ground-truth values ($\mathbf{y}_{t_f+1,\text{reg}}$) are supplied as inputs rather than the model's own predictions ($\hat{\mathbf{y}}_{t_f+1,\text{reg}}$). Similarly, to facilitate efficient training, input data $\mathbf{x}_{t_{ts}}^{ts}$ is standardized to zero mean and unit variance within each training observation.

[4] For certain recent years with incomplete data in the archive, missing articles were manually retrieved from DN's digital print edition.

We allocate 90% of the dataset for training and reserve 10% for validation. Validation is performed every 200 iterations. To prevent overfitting, early stopping is triggered after five validation rounds without improvement. Additional regularization is applied through L2 penalties on the model head parameters and dropout. Dropout mitigates overfitting by randomly setting a fraction of hidden units to zero during training (Srivastava et al., 2014), and is applied throughout the attention and feed-forward sublayers, promoting reliance on diverse patterns rather than specific features. The composite loss function in (11), combined with hard parameter sharing via (7), is an example of so-called multi-task learning and provides another form for (implicit) regularization (Ruder, 2017). In machine learning this approach has a long history because it typically "improves generalization by leveraging the domain-specific information contained in the training signals of related tasks" (Caruana, 1997).

Model training is performed on two NVIDIA RTX 4090 GPUs with 24 GB of memory each. Training typically completes in approximately 3 hours, terminating after about 10 epochs due to stagnation in validation performance.

5 Validation results and diagnostics

In this section we use the data validation set to assess overall model performance, conduct permutation tests, component ablations, and hyperparameter sensitivity analyses.

5.1 Model accuracy

On average, the model achieves nearly 100% classification accuracy across the validation sequences and root mean squared errors (RMSE) of approximately 0.02 for sentiment predictions and 0.18 for the business cycle predictions. To contextualize these results, we compare them to three alternative approaches: a Naive Bayes classifier for text classification, a regularized Lasso regression for sentiment prediction, and three state-of-the-art LLMs with multimodal prompts.

Naive Bayes is a simple but often competitive baseline in multi-class classification tasks. Prior studies have found that deep learning models such as BERT typically outperform Naive Bayes by 5–20% (Minaee et al., 2021). For the sentiment task, we estimate a Lasso regression model using 5-fold cross-validation for hyperparameter tuning. For both models the document-term matrix is used for training. In terms of using LLMs directly for time series modeling, Gruver et al. (2023) was among the first to document that LLMs can surprisingly zero-shot extrapolate time series at a level comparable to or exceeding the performance of purpose-built time series models trained on the downstream tasks, while Requeima et al. (2024) demonstrate that LLMs can process numerical data

Table 2. Panel A reports classification and sentiment performance of the *Narrater* relative to a Naive Bayes (NB) classifier and the Lasso sentiment regression, as well as relative performance measures on all tasks compared to three different LLMs. A score below 1 indicates the *Narrater* performs better. Panel B reports the performance of the benchmark model relative to several alternative model specification; without a text encoder (*NoTxt*); without a time series encoder (*NoTS*); for two different embedding dimensions (*TxtComp* and *TsExp*); and estimating the model without simultaneously fine-tuning the language encoder (*NoFt*).

Panel A	NB / Lasso		LLM (zero-shot)			*Panel B*	Ablation		Hyperparameter		Fine tuning
			Gemini	o3	R1		NoTxt	NoTS	TxtComp	TsExp	NoFt
Class (accuracy)	0.95		0.42	0.34	0.32		0.36	1.00	1.00	1.00	0.94
Sent. (RMSE)		0.29	0.05	0.04	0.07		0.08	0.70	0.63	0.63	0.16
BusC. (RMSE)			0.55	0.68	0.00		0.83	0.96	1.00	0.97	0.92

and make probabilistic predictions at arbitrary locations, guided by natural language text describing a user's prior knowledge. Inspired by these developments we query Google's Gemini, OpenAI's o3, and DeepSeek's R1 with a multimodal prompt (Appendix E), asking the LLM for the next observation prediction as well as the class and sentiment of the text.

Panel A of Table 2 reports the model's performance relative to these baselines. While the *Narrater*'s near-perfect classification accuracy suggests the task may not be particularly difficult, the model consistently outperforms Naive Bayes by approximately 5 percentage points. In the sentiment regression task, the improvement is more pronounced, and the proposed model reduces the RMSE by nearly 70% relative to the Lasso baseline. The three LLMs demonstrate substantially lower performance on the text classification and sentiment scoring tasks, with classification accuracy approaching random chance and sentiment RMSEs an order of magnitude larger than the proposed model's, suggesting that a more refined prompt engineering strategy should be adopted. In particular, with the used prompt, the LLMs do not seem to be able to efficiently utilize the multimodal information. In terms of predicting the business cycle the relative performance of the o3 model is not bad, but still substantially worse than the *Narrater*. One reason for the somewhat weak LLM performance is that they occasionally make extreme predictions, an observation also made by Carriero et al. (2024). Another reason is likely that these LLMs are foundational models not trained specifically to utilize structural multimodal information to address the problem at hand, and studies such as Gambacorta et al. (2024) and Dell (2024) also demonstrate that specific fine-tuning of, e.g., BERT, delivers performance on par, or better, than LLMs in a range of (economic) domain specific classification tasks.

5.2 Permutation tests

The model is trained on data with an assumed multimodal structure. To learn about the weights the model assigns to the different data modalities we perform permutation tests.[5] In the tradition of Breiman (2001) and Fisher et al. (2019), an input feature is determined as "important" if shuffling its values increases the model error because in this case the model relied on the feature for the prediction. Here we assess the relative importance of text versus time series input by randomly shuffling the respective encoder input data and then comparing the change in predictive accuracy relative to the original case.

Figure 4 presents results for the last elements in the predicted sequences from 10 repetitions of the permutation test. Results for the benchmark *Narrater* are reported in the first row in each graph, while the remaining rows report robustness. For the classification task, performance drops sharply when the text input is permuted, but not when the time series input is permuted. The absence of a permutation effect on text classification in our baseline reflects that the textual input is sufficiently informative on its own, making additional cues from the time series redundant in this setting. This should not, however, be read as a limitation of the multimodal design itself. In Section 6.4, for example, we show that expanding the task complexity (increasing the number of classes) under the benchmark simulation scheme produces a stronger role for the time series, and thus a clear multimodal interaction.[6] In contrast, both modalities contribute to sentiment and growth predictions. For growth, shuffling either modality roughly doubles the prediction error, indicating joint dependence. For sentiment, however, performance deteriorates far more when the text input is randomized, highlighting its dominant role in that task.

5.3 Ablation experiments

To more directly assess the contribution of each model component, we conduct ablation experiments by removing either the text encoder ($NoTxt$) or the time series encoder ($NoTS$) from the full model. In both cases, the modified models are re-estimated to account for potential reallocation of weights across remaining components.

[5] Rather than applying post-hoc feature attribution methods such as SHAP or LIME, we use permutation tests to evaluate modality-level importance. This approach aligns more closely with our structural design and provides an intuitive assessment of the model's dependence on each modality. Moreover, for Transformer-based architectures it is still an open research question how to apply feature attribution methods.

[6] In such cases, borderline class assignments, noise, and greater overlap between textual features make it harder for the model to rely on text alone, increasing the value of complementary information from the time series. Similar patterns are observed for the additional robustness experiments reported in Figure 4. In the interest of brevity, these are discussed in Appendix B.3.

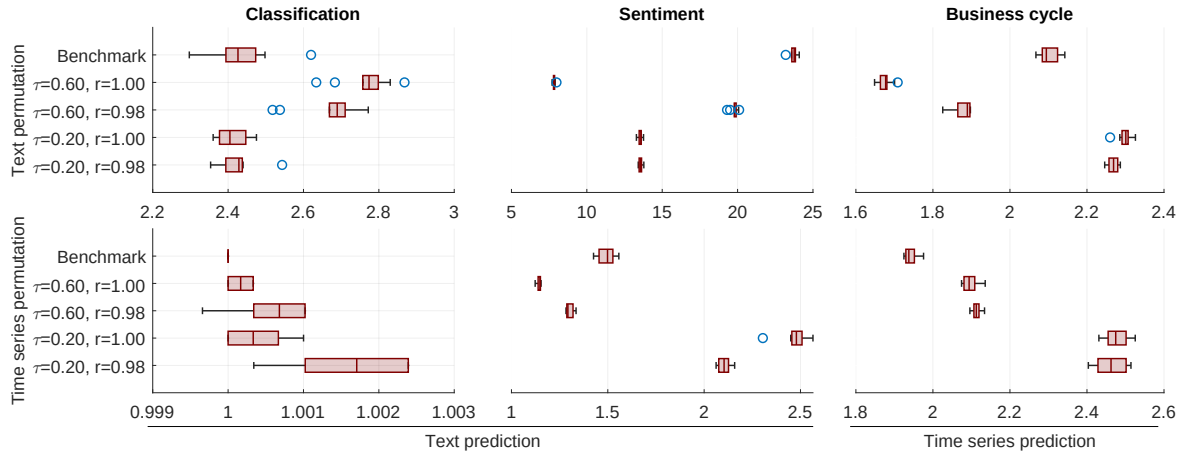


Figure 4. Traditional box plots of relative performance differences from permutation tests. A value above one indicates that the model performs worse when the particular data input is randomized.

Panel B of Table 2 summarizes the results. Removing the text encoder leads to a substantial decline in predictive performance across all outputs, especially for sentiment, where accuracy drops markedly. While the time series encoder is less critical for classification, it still plays an important role: sentiment prediction performance declines by approximately 30% in the *NoTS* model compared to the benchmark. These findings closely align with the permutation test results in Figure 4, and demonstrate that the benchmark model effectively integrates both data modalities to improve prediction.

5.4 Hyperparameter sensitivity

We also evaluate two reduced-size variants of the benchmark model in which the embedding dimensions of the time series encoder and decoder are reduced by a factor of eight, significantly lowering the number of trainable parameters. To align embedding dimensions in the cross-attention layers, we consider two alternatives: in the first (*TxtComp*), text encoder embeddings are compressed to match the smaller time series embeddings; in the second (*TsExp*), the time series embeddings are expanded to match the original text dimensions. As shown in Table 2, both configurations perform slightly worse than the benchmark, with negligible differences between them, suggesting that the original embedding sizes capture meaningful structure.

The final column of Table 2 compares the benchmark to a model in which the BERT language encoder is not fine-tuned, i.e., the pre-trained weights θ_1 in (1) remain fixed during training. This constraint leads to a substantial drop in sentiment prediction performance and a 6–8% decline in accuracy for the classification and growth tasks. These results underscore the importance of task-specific fine-tuning of the language encoder.[7]

[7]The $\text{BERT}_{\text{TINY}}$ tokenizer is based on an English-language corpus. We experimented with retraining the encoder using a Norwegian tokenizer and domain-specific corpus, but found that performance declined on

6 Reinforcement learning for out-of-sample applications and news summation

To evaluate the model on non-synthetic data, we conduct a quasi–real-time out-of-sample experiment using DN news and Norwegian macroeconomic time series (Section 4.3). We initialize the model with news texts, macro data, and business cycle values for the sample 1986Q2–1996Q1, then iteratively roll it forward from 1996Q2 to 2023Q2 - noting that data after 2010 were not seen during training or validation.[8]

Each new quarter contains thousands of news articles, but only one can be mapped into the model state as the representative story. Thus, the real-time application becomes a sequential decision problem: once an article is chosen, it enters the state history and can influence all subsequent predictions. This motivates an RL framing, wherein the *Narrater* environment updates states and rewards and an agent scores articles according to a policy, as illustrated in Figure 2.

To make estimation of the policy function a tractable problem in a high-dimensional setting, while ensuring predictive accuracy, interpretable narrative attribution, robustness and efficient deployment, we build on three strands of work: (i) entropy-regularized control, which motivates the use of soft value functions and Boltzmann policies (Ziebart, 2010; Todorov, 2007; Kappen, 2005; Haarnoja et al., 2018); (ii) Kullback–Leibler (KL) based policy optimization, which provides stable projection methods for updating policies toward value-weighted targets while remaining close to a baseline distribution (Peters et al., 2010; Schulman et al., 2015; Abdolmaleki et al., 2018); and (iii) classical rollout paradigms from stochastic control, which justify usage of short-horizon lookahead to approximate long-horizon value functions (Bertsekas and Tsitsiklis, 1996; Bertsekas, 2011, 2019).

Appendix D.1 briefly outlines the mapping between this framework and the classical Bellman recursions. Below we describe our approach in greater detail. A simple special case is greedy selection, which requires no additional estimation or learning.

a range of language understanding tasks. While perhaps surprising, we suspect that this negative result likely is driven by the (limited) size and quality of the text corpus used for re-training. We also tested the larger, pre-trained Norwegian $\text{BERT}_{\text{BASE}}$ model (Kummervold et al., 2021), without re-estimating the model weights. In line with results reported in Table 2, the Norwegian $\text{BERT}_{\text{BASE}}$ model performed worse than our benchmark. While such models may offer benefits for more complex tasks, they appear sub-optimal for our application.

[8]The experiment is labeled *quasi-real-time* because we do not use real-time vintages of data but still assume an information structure that to a large degree mimics the information structure available in a real-world application.

6.1 Reinforcement learning approach

To outline the problem formulation, we start from the maintained assumption that news is available for the current quarter whereas the macroeconomic data is not, i.e., $t_{ts} = t_f$ and $t = t_f + 1$. Further, for a generic quarterly time segment, let i denote the i-th news article in the last quarter T in the segment and N_T the total number of articles in this quarter. The state at time T is then $s_T = \left(\mathbf{s}_T,\ \mathbf{x}^{ts}_{T_{ts}},\ \mathbf{x}^{f}_{T_f}\right)$, and the action set $\mathcal{A}_T = \{a_{T,1}, \ldots, a_{T,N_T}\}$ consists of all candidate news articles in quarter T. Choosing $a_{T,i}$ corresponds to selecting article i for narrative attribution as well as model predictions $\hat{\mathbf{y}}^{(i)}_{T_f+1,reg}$, $\hat{\mathbf{y}}^{(i)}_{T,class}$, and $\hat{y}^{(i)}_{T,sent}$, which in turn feed into the next state s_{T+1}.

In general, any reward function can be specified. Here, to emphasize accuracy, allow for narrative priors, and favor fundamentals and persistence, we evaluate candidates along four dimensions:

$$\begin{aligned} r_T(s_T, a) = -\mathrm{RMSE}\Big(G\hat{D}P^{(i)}_{1:T_f}, GDP_{1:T_f}\Big) + \lambda_N \,\mathrm{NPRIOR}(a, \tilde{a}_T) \\ + \lambda_F \,\mathrm{FUND}(a) + \lambda_P \,\mathrm{PER}(a, \mathcal{H}_{T-1}), \end{aligned} \tag{12}$$

where $G\hat{D}P^{(i)}_{1:T_f}$ is approximated as the sum of the elements in $\hat{\mathbf{y}}^{(i)}_{1:T_f,reg}$, the subscripts associated with a are dropped for notation simplicity, and the mathematical definition of each term is described in Appendix D. In short, accuracy is captured via $\mathrm{RMSE}(G\hat{D}P^{(i)}_{1:T_f}, GDP_{1:T_f})$; $\mathrm{NPRIOR}(a, \tilde{a}_T)$ measures similarity between the candidate and a reference narrative $\tilde{a}_T$. The usage of reference texts is similar in spirit to the narrative identification approach taken in Antolín-Díaz and Rubio-Ramírez (2018), and helps keep the model aligned with any narrative prior the model user might have for particular time periods; $\mathrm{FUND}(a)$ scores each candidate article by how likely it is fundamental news (demand or supply) relative to noise and intends to capture the idea that important events are fundamental; $\mathrm{PER}(a, \mathcal{H}_{T-1})$ measures persistence and rewards overlap with previously chosen narratives, or in other words, provides higher rewards for narratives that prevail. Finally, $\lambda_N, \lambda_F, \lambda_P$ are hyperparameters governing trade-offs.

Classical optimal control would optimize a policy to maximize cumulative reward, but with thousands of candidate actions per period ($N_T \gg 2000$) this is computationally infeasible, and the resulting policies tend to be brittle with poor exploration. We therefore adopt an entropy-regularized approximation with short rollouts of horizon H and a parameterized softmax policy (Ziebart, 2010; Todorov, 2007; Kappen, 2005; Haarnoja et al., 2018).

The rollout-based target distribution weights actions by truncated expected returns:

$$q_T(a) \ \propto\ \mathbb{E}\left[\sum_{h=0}^{H-1} \gamma^h r_{T+h}(s_{T+h}, a_{T+h}) \,\middle|\, a_T = a\right], \tag{13}$$

with discount factor γ. Setting $H = 1$ recovers a pure contextual bandit formulation, matching the setup used, for example, in news recommendation (Li et al., 2010), while modest H captures near-term dependencies without exploding complexity.

The policy class is a multinomial logistic regression over features $\phi(s_T, a)$ extracted from the multimodal representation in (7):

$$\pi_\theta(a \mid s_T) = \frac{\exp\{\beta\, \phi(s_T, a)^\top \theta\}}{\sum_{a' \in \mathcal{A}_T} \exp\{\beta\, \phi(s_T, a')^\top \theta\}}, \tag{14}$$

where β is the inverse temperature. This amortizes rollout information into parameters θ, so evaluations at deployment do not require new rollouts.

Training projects the policy onto the rollout targets using a KL-regularized objective:

$$\mathcal{L}(\theta) = \sum_{t_s=1}^{T_s} \Big[\mathrm{KL}(q_{t_s} \,\|\, \pi_\theta(\cdot \mid s_{t_s})) + \lambda_{\mathrm{KL}}\, \mathrm{KL}(\pi_\theta(\cdot \mid s_{t_s}) \,\|\, \pi_0(\cdot \mid s_{t_s})) - \lambda_{\mathrm{H}}\, \mathcal{H}(\pi_\theta(\cdot \mid s_{t_s})) \Big], \tag{15}$$

where $t_s = 1, \ldots, T_s$ are the (quarterly) decision points. Further, π_0 is a baseline prior (here $\propto \exp(-\mathrm{RMSE})$), entropy $\mathcal{H}(\cdot)$ promotes exploration, and $\lambda_{\mathrm{KL}}, \lambda_{\mathrm{H}}$ control conservatism and smoothing. This follows the logic of KL-regularized or trust-region updates (Peters et al., 2010; Schulman et al., 2015; Abdolmaleki et al., 2018), ensuring stable and interpretable policy improvement even in high-dimensional spaces.

While this general policy learning scheme provides a principled foundation for narrative attribution, in practice it can be useful to employ a simpler, faster rule, more robust to potential out-of-sample distributional shifts. Specifically, if we restrict attention to the immediate predictive reward and collapse q_T onto the single best candidate, the policy reduces to a greedy bandit:

$$\hat{a}_{T,i} := \arg\min_{a \in \mathcal{A}_T} \ \mathrm{RMSE}\Big(\hat{GDP}^{(i)}_{1:T_f}, GDP_{1:T_f} \Big). \tag{16}$$

As such, equation (16) can be seen as a degenerate case of the general RL framework without any rollouts and reward based solely on prediction accuracy.

6.2 Application details

In the real-time experiments below, we implement both the greedy baseline of (16) and the RL policy, trained via contextual bandit rollouts and KL-regularized projections. We denote these as the *Narrater-greedy* and *Narrater-RL*, respectively. The RL approach permits integrating additional dimensions of narrative quality - such as coherence with the narrative prior - into a single decision rule. It also balances exploration and exploitation through the softmax temperature and entropy regularization, and delivers probabilistic policy weights that quantify the decisiveness of each choice. The greedy rule cannot

accommodate these trade-offs and provides no natural measure of confidence. It does, however, serve as a fast and transparent benchmark.

To explore the effects of different hyperparameter choices, we consider three versions of *Narrater-RL*; the default specification with rewards as in (12); a version emphasizing only RMSE rewards ($\lambda_F = \lambda_N = \lambda_P = 0$); and finally a version with only RMSE and NPRIOR rewards ($\lambda_F = \lambda_P = 0$). At each decision point the *Narrater* delivers an updated "smoothed" history of its output history - analogous to standard forward-backward filtering algorithms in traditional time series analysis. RMSE rewards are computed using this sequence, ensuring consistency with the information structure encountered in real-world applications.

In total 110 quarterly decision points, or time segments, are processed. For the initial segment, covering the sample 1986Q2 to 1996Q1, the history of $\mathbf{s}_{t-1}$, $\mathbf{x}^{ts}_{t_{ts}}$, and $\mathbf{x}^{f}_{t_f}$ for $t_f = 1, \ldots, T_f$ is constructed by iterating the model forward starting from a random draw of $\mathbf{s}_1$ and setting $\mathbf{x}^{f}_{1}$ equal to the eight quarter growth in GDP. Implementing extractive summarization and narrative attribution using (16) requires no training. For learning θ in (14) only data for decision points prior to 2010 are used. The selected narrative, and thus the extractive summary, maximizes $\pi_\theta(a \mid s_t)$, with the optimal selection index denoted $\hat{i}$.

The reference texts used in the reward function act as narrative priors: they need not reappear verbatim out of sample, but during training they bias the policy towards articles that align semantically with their content. Thus, the learned policy is effectively anchored in a particular narrative direction. Out of sample this anchoring ensures that new but related articles are more likely to be selected, thereby preserving continuity of the narrative attribution. Although each decision point in the training sample in principle could contain a narrative reference article $\tilde{a}_T$, we only specify three related to the collapse of Long-Term Capital Management in 1998Q3, the 9/11 terror attacks in 2001Q3, and the bankruptcy of Bear Stearns in 2008Q1.

Appendix D.2 provides additional training details and hyperparameter settings.

6.3 Application results

Capturing business cycle fluctuations and narrative attribution are the primary objectives of the *Narrater*. Predictive accuracy of observables, however, provides a more objective evaluation of model performance. For this reason we start by evaluating growth predictions and then in subsequent sections discuss the model's narrative output.

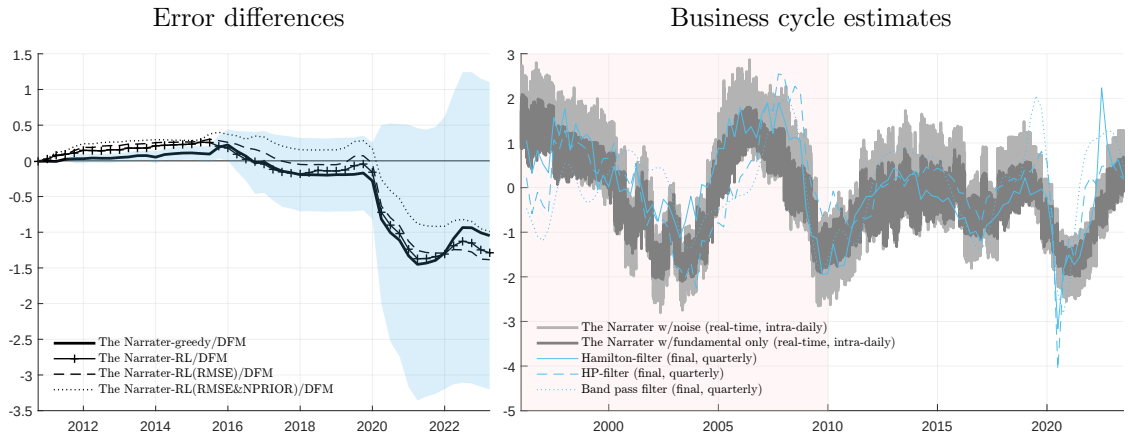


Figure 5. Figure 5a reports the cumulative squared prediction error difference between the *Narrater* and the DFM benchmark. A falling value implies a relative *Narrater* improvement. The shaded areas are 95% confidence intervals of differences in predictive performance, computed following Diebold and Mariano (1995) with HAC corrected standard errors. Figure 5b reports the real-time estimates of $\hat{\mathbf{y}}^{(i)}_{T_f+1,reg}$ implied by all incoming news and aggregated into either the sum of all its components or only the demand and supply components. The white areas are time periods with data not seen during training and validation.

6.3.1 Growth predictions

To evaluate the model's growth predictions we use the sum of the elements in $\hat{\mathbf{y}}^{(\hat{i})}_{t_f+1,reg}$ as a forecast of $\Delta_8 GDP_{T_f+1}$, the observed eight-quarter GDP growth, and compare it against a Dynamic Factor Model (DFM) benchmark. The DFM follows the specification outlined in Section 4.1 and produces recursive one-step-ahead forecasts via the Kalman Filter. To avoid information leakage, we evaluate all models over the 2010–2023 period, which is strictly out-of-sample for the *Narrater*.

Figure 5a shows the cumulative difference in squared prediction errors between the *Narrater* versions and the DFM benchmark. For the sample as a whole, the differences are not statistically significant. However, the *Narrater* models tend to outperform the DFM during the early Covid-19 period, suggesting the model's capacity to integrate timely textual information during periods of heightened volatility. Furthermore, using a fully RL-based implementation typically yields a small improvement in average forecasting performance, even with a somewhat outdated policy function. Overall, the *Narrater-RL* variant trained with RMSE-only rewards performs best, although the differences are modest.

To examine how predictive accuracy evolves within a quarter as new information becomes available, we focus on the *Narrater-greedy* model and replace the predictions based on the $\hat{i}$th article with those obtained as the simple running average of all the i articles within the quarter. Focusing on daily time steps, Figure A.2a (Appendix A) illustrates this dynamic and echoes findings in the nowcasting literature (Giannone et al., 2008):

Compared to the DFM, the *Narrater* shows a relative performance gain of approximately 20% at the start of the quarter, which steadily improves, stabilizing around 27% relative improvement after one month (30 days) of news data has been processed.[9]

6.3.2 The business cycle and economic sentiment

Figure 5b presents intra-daily business cycle estimates implied by processing all articles in the application sample. The correlations between the business cycle estimates provided by the *Narrater-greedy* and *Narrater-RL* models are high and above 0.95 (Table A.1 in Appendix A). For clarity we thus use the estimates obtained from rolling the *Narrater-greedy* model forward at each decision point in the sample. Specifically, we display predictions aggregated from the components of $\hat{\mathbf{y}}^{(i)}_{T_f+1,\text{reg}}$, either as the full sum or restricted to the demand and supply components for each article instance in the sample. For comparison, we also include quarterly business cycle estimates from three standard filtering methods: the eight-quarter GDP difference (Hamilton, 2018), the HP filter with $\lambda = 40000$ (Hodrick and Prescott, 1997), and a band-pass filter targeting cycles between 1.5 and 8 years (Baxter and King, 1999).[10]

The model's cyclical estimates are well aligned with the inductive bias instilled into the model via the training data, and as Figure 5b illustrates, also well aligned with estimates obtained from the other methods. The usage of the daily news input, however, facilitates more frequent updates using the Transformer-based model than the other filtering methods considered here. It is also clearly visible how adding noise to the demand and supply components of $\hat{\mathbf{y}}^{(i)}_{t_f+1,reg}$ substantially increases the variation in the predictions.

Turning to the quarterly time frequency Figures A.1a and A.1b (Appendix A), compare the model's filtered (real-time) and smoothed (retrospective) business cycle estimates. The squares show $\hat{y}^{(\hat{i})}_{T_f+1,\text{reg}}$ as predicted at the end of each quarterly segment, while the solid lines reflect the smoothed estimates $\hat{\mathbf{y}}^{(\hat{i})}_{1:T_f+1,\text{reg}}$. Although the filtered estimates are revised slightly as new data becomes available, these adjustments tend to be modest, suggesting robustness in the model's real-time inference.

[9]The performance scores in this section are a function of training data assumptions, model structure, and the adequacy of the data sources used for prediction. To highlight the role of the latter we perform the predictive experiment using a generic text sequence for all time periods. I.e., we replace the DN news corpus with the sequence; "This is not a real sentence. This is just noise and should not contribute towards lowering the prediction error.", reflecting prior confidence in the DN corpus. The results reported in Figure A.2b (Appendix A) support this prior and show that the DN-based news helps improve predictive accuracy with a roughly 25% relative improvement.

[10]The HP filter parameter $\lambda = 40000$ is standard for Norwegian GDP applications. As Norway lacks an official business cycle dating authority (unlike the U.S. NBER), the evaluation of cyclical estimates remains inherently subjective.

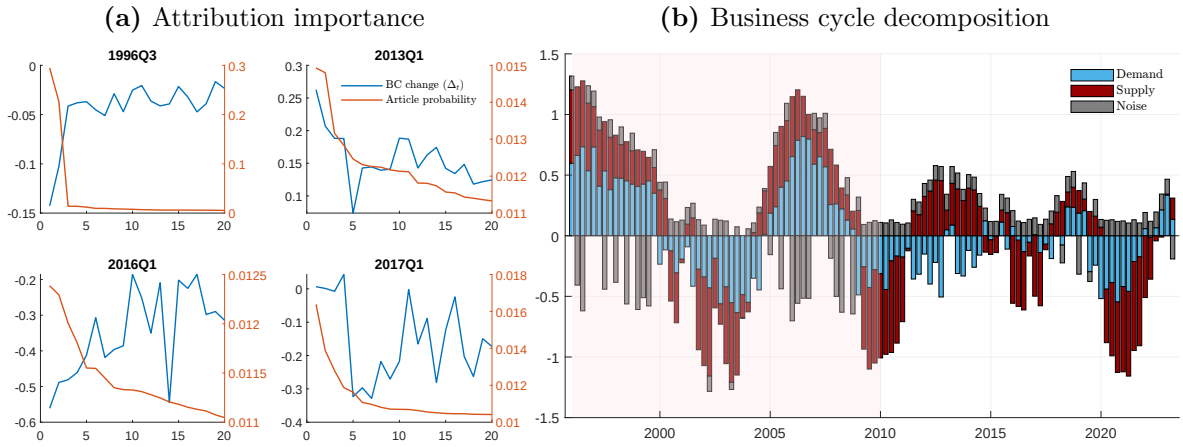


Figure 6. Figure 6a reports, for four randomly chosen dates, the probability score for the most likely articles and their implied business cycle estimate. Figure 6b reports the real-time estimate of the vector $\hat{\mathbf{y}}^{(\hat{i})}_{T_f+1,reg}$. The white areas are time periods with data not seen during training and validation.

A large literature documents a strong link between aggregate sentiment and the business cycle (Blanchard, 1993; Barsky and Sims, 2012; Shapiro et al., 2022). Figure A.3a (Appendix A) shows that *Narrater-greedy* sentiment, averaged across articles and normalized, correlates with the estimated cycle at 0.83 on a quarterly basis. At the article level, sentiment is noisier, has a negative bias, and is more weakly correlated, as illustrated in Figure A.3b (Appendix A). When aggregated to daily frequency, the correlation remains above 0.6, supporting the model's assumption that sentiment and macroeconomic conditions are (partly) jointly driven by common shocks.

6.3.3 Narrative attribution

Having established that the model produces plausible estimates of the business cycle and reasonable growth predictions, we now turn to its more structural outputs.

Figure 6a highlights how narrative attribution shapes business cycle estimates. It reports the implied change in the cycle at multiple dates for the 20 articles with the highest selection probabilities using the *Narrater-RL* framework. Selecting a different narrative can materially shift short-term dynamics, underscoring the value of a systematic and principled attribution scheme. As we document below, however, the RL-based and greedy versions considered here produce reasonably similar structural interpretations.

The vector $\hat{\mathbf{y}}^{(\hat{i})}_{T_f+1,\text{reg}}$ delivers a real-time decomposition of the business cycle into demand, supply, and noise. Figure 6b plots demeaned contributions over time. Noticeable business cycle events, such as: the pre–Great Financial Crisis boom is demand-led; the subsequent downturn reflects negative supply shocks; and the Covid-19 recession combines both, with a modest late-sample demand rebound. Noise is present throughout but wanes in recent years. The figure reports output from the *Narrater-greedy* model. The

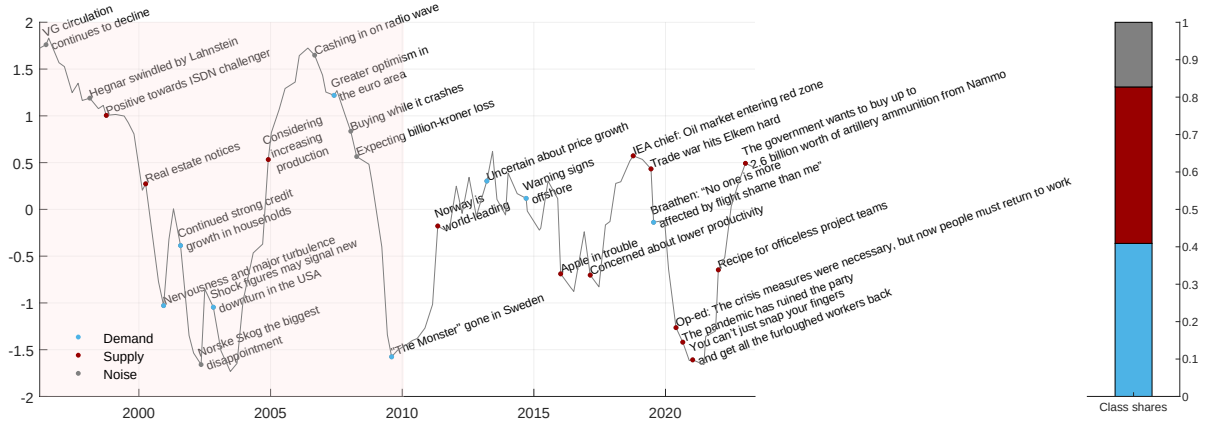


Figure 7. Real-time estimate of the vector $\hat{y}^{(\hat{i})}_{T_f+1,reg}$ together with the headline and class of article $\hat{a}$ in a given quarter. To not clutter the graph extractive news summaries are not illustrated for each quarter. The white areas are time periods with data not seen during training and validation. The news texts are translated from Norwegian to English using GPT-4.5. The stacked bar reports the class share of $\hat{a}$ articles across the whole sample.

default *Narrater-RL* model yields a similar decomposition; component correlations with the greedy version are 0.93 (demand), 0.93 (supply), and 0.59 (noise).

Although these decompositions cannot be objectively verified, we compare them to the structural decomposition generated by the DFM used to simulate training data. Figure A.4 (Appendix A) shows that 70% (75%) and 82% (88%) of the *Narrater-greedy*'s (*Narrater-RL*) demand and supply decompositions, respectively, fall within the admissible bounds of the DFM's when this model is estimated using the whole sample. Considering that *Narrater*'s outputs are real-time, unlike the ex-post DFM estimates, this degree of overlap is encouraging.[11]

Figure 7 uses the *Narrater-greedy* model output, and combines quarterly business cycle estimates with extractive summaries of news articles, using the headline and classification of article $\hat{i}$ for each quarter. To maintain clarity, summaries are shown selectively, while Table 3 provides a list of representative first sentences for interpretability. Additional sentiment and class outputs for $\hat{i}$ articles are shown in Figure A.6 (Appendix A).

Given that each quarter includes over 2000 articles, the quality of these summaries is notable, and suggests meaningful scope for model-driven narrative attribution. For example, headlines such as "Hegnar beaten by Lahnstein", "The Monster gone in Sweden",

[11] Figure A.5 (Appendix A) displays the fraction of news articles classified as demand or supply, aggregated to quarterly frequency, and using the *Narrater-greedy* model version. Three observations stand out. First, only 15–20% of articles are classified as demand- or supply-related, implying most news is categorized as noise. Second, the temporal distribution of classified news aligns with known events, e.g., increased demand coverage in the early 2000s and around the Great Financial Crisis, and increased supply coverage during the early sample period, post-crisis years, and the Covid-19 episode. Third, demand-related coverage shows more variation over time, possibly due to an upward drift in classification rates.

Table 3. The headline, first sentences, date, class, and sentiment of some of the stories identified via the extractive news summary. The news texts are translated from Norwegian to English using GPT-4.5.

Date	Class	Sent.	Headline and text
1998Q1	Noise	-0.53	*Hegnar swindled by Lahnstein: "Newspapers with profits exceeding two million kroner lose press support. The Ministry of Culture has enacted a profit cap that means Trygve Hegnar's Finansavisen and..."*
2000Q4	Demand	-0.53	*Nervousness and large volatility: "It has been quite a turbulent week for American and European stock investors. The nervous atmosphere in the stock markets was underscored by the significant price movements...."*
2004Q4	Supply	0.13	*Considering increasing production: "Saudi Arabia may increase the country's production capacity to 12.5 million barrels per day to ensure that the oil market is adequately supplied..."*
2009Q3	Demand	-0.61	*"The Monster" is missing in Sweden: "The economic downturn is hitting Sweden hard, but the country is recovering faster than the government initially feared..."*
2012Q3	Demand	-0.65	*Interest rates: "A series of disappointing confidence indicators from European businesses sparked new concern about the euro crisis..."*
2017Q1	Supply	-0.55	*Concerned about falling productivity: "Most of us have gotten less to get by on. But one of the most worrying things is that productivity growth in the industry is falling, warns researcher Ådne Cappelen..."*
2019Q3	Demand	-0.62	*Braathen: "Nobody is more affected by flight shame than I am": "Per G. Braathen (58) must lay off one third of the employees at the Swedish airline BRA because passenger numbers are falling short. He believes that the resulting shame will spread to Norway..."*
2020Q2	Supply	-0.51	*Op-ed: The crisis measures were necessary, but now people must return to work: "Typically enough, there is now, in the start-up phase, criticism of the emergency crisis measures taken during the shutdown phase. Even so, other measures are needed now. Several have criticized the crisis measures for contributing to companies choosing to remain closed..."*

and "Concerned about lower productivity" plausibly reflect noise, demand, and supply stories, respectively. Moreover, the narrative structure aligns well with known macroeconomic developments, for instance, Covid-19-related stories during the 2020 downturn, recession-themed articles during global slowdowns, and oil-related news during the energy crisis in 2014 and 2015.

Evaluated across the whole sample, roughly 40% of the summaries are allocated to the demand and supply classes (Figure 7). To further illustrate how the model output can be used to gauge the narrative strength of this type of attribution we compute two additional metrics for each quarter t; the *RMSEMargin* and *ClassSharpness*. The *RMSEMargin*

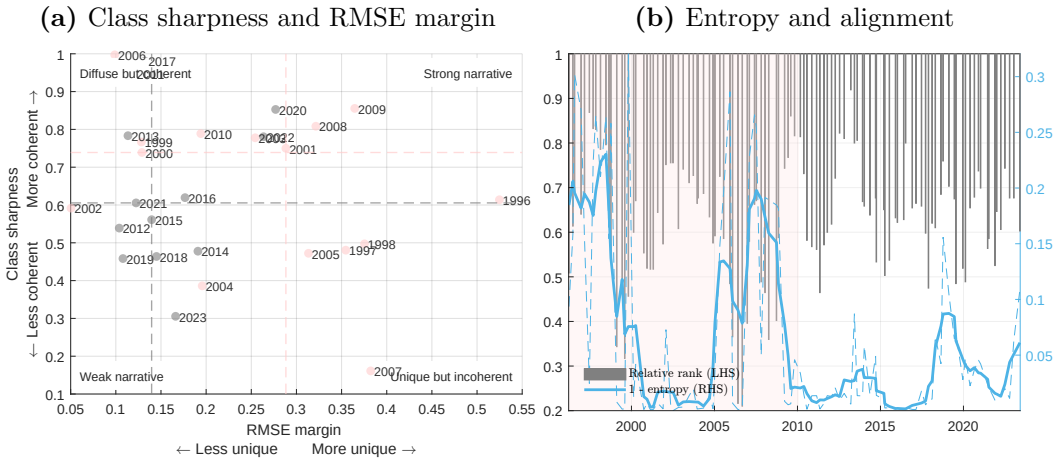


Figure 8. Figure 8a reports a scatter plot of the *RMSEMargin* and *ClassSharpness* measures. Quadrants, illustrated using red and gray broken lines, are defined using the median for the in- and out-of-sample periods, respectively. Values are aggregated to yearly averages for visual clarity. See the text for further details. The black bars in Figure 8b report the relative rank, defined as $1 - \frac{r_{t_s}-1}{N_{t_s}-1}$, between articles selected by the *Narrater-greedy* and *Narrater-RL* routines, where r_{t_s} is the rank of the *Narrater-greedy* selection in the probability distribution of the default *Narrater-RL* routine. The blue lines report $1-\hat{\mathcal{H}}_{t_s}$, where the broken line is the raw estimate and the solid line is smoothed with a 4-period moving average for visual clarity. The white areas are time periods with data not seen during training.

is defined as the relative performance gap between the best-performing article and the average performance of the next $K = 100$ best articles - all in terms of RMSE loss. Similarly, we define *ClassSharpness* for quarter t as the proportion of top-K articles whose predicted class matches that of the chosen article.

Figure 8a illustrates how periods with a high *ClassSharpness* and *RMSEMargin* typically are associated with notable global events and business cycle fluctuations. For instance, both the global financial crisis (2009) and the Covid-19 outbreak (2020) display relatively large *RMSEMargins* alongside high class coherence. In contrast, years such as 2002 and 2012 combine low *RMSEMargins* with weaker class coherence, suggesting more random and thus less focused narratives.

The bar plot in Figure 8b shows that the selected articles from the *Narrater-greedy* summation routine also tend to be among the articles with the highest probability score when applying the default *Narrater-RL* policy function. Indeed, the median of the greedy selections is above the 75th percentile of the top performing articles using the default *Narrater-RL* model, and the two approaches selects the same article class in 65% of the cases. Table A.1 (Appendix A) shows that this statistic increases if a more RMSE oriented reward function is used. Thus, narrative attribution is affected by user preferences, but still fairly consistent across the different hyperparameter specifications considered here.

An advantage with the *Narrater-RL*-based approach is that it naturally delivers prob-

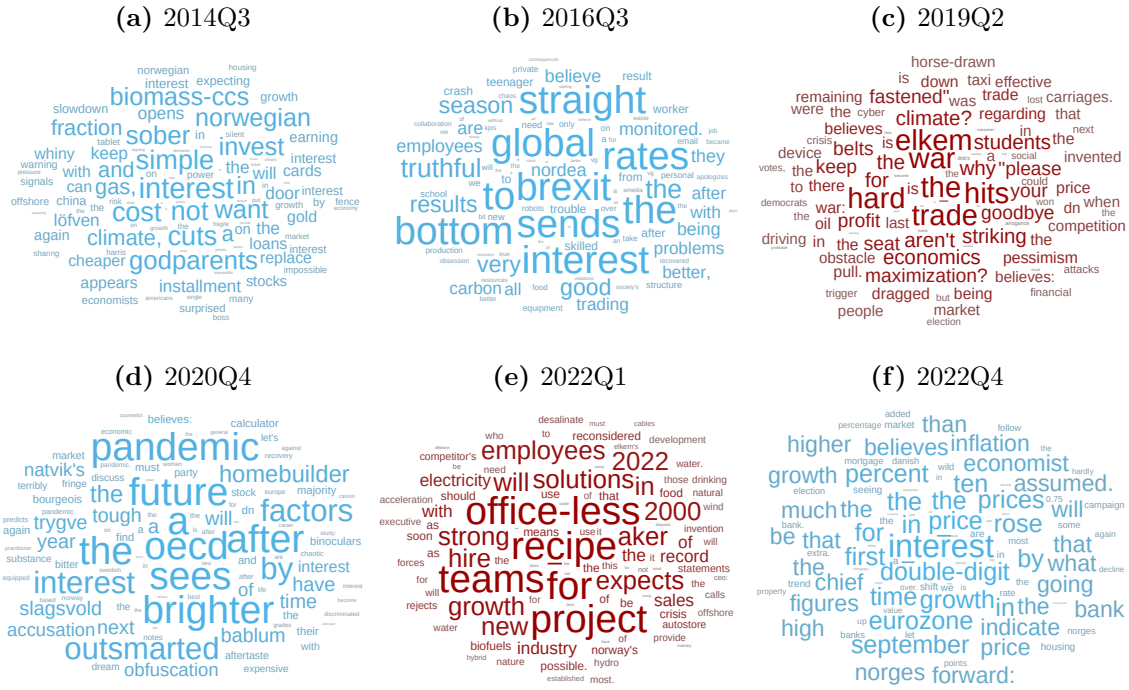


Figure 9. Figures 9a–9f report traditional word clouds derived from words in news headlines weighted by the article's selection probability using the default *Narrater-RL*.

abilistic policy weights that quantify the decisiveness of each choice. The blue line in Figure 8b reports $1 - \hat{\mathcal{H}}_{t_s}$, where $\hat{\mathcal{H}}_{t_s} = H_{t_s}/log(|\mathcal{A}_{t_s}|)$, as a measure of attribution decisiveness across time. There is relative high entropy in the early 2000s, following the Great Financial Crisis (GCF) and the oil-price slump in 2014, and also in the period before and after Covid-19.

Another favorable property of the probabilistic approach is that it also enables period-level "abstractive" summaries. Figures 9a–9f provide an illustration where headline words are weighted by an article's selection probability and rendered as traditional word clouds. We focus on six dates/periods aligned with canonical business-cycle episodes. Because entropy varies over time, the sharpness of narrative attribution likewise fluctuates, as evident in the figure, but the relevance of the narrative attribution is striking: The model automatically surfaces the mid-2014 oil-price slump, the Brexit referendum, the China-U.S. trade war, Covid-19 and remote work, and the inflation surge towards the end of the sample.

6.4 Alternative model configurations

To demonstrate the flexibility of the framework, and how it is easily adaptable and capable of addressing diverse user needs, we evaluate two alternative configurations of the model: the *Narrater-topic* and the *Narrater-mixed*.

In *Narrater-topic* we keep the benchmark setup but expand the number of text classes to $C = 9$. Both demand and supply are divided into four topical subclasses - *policy/regulation*, *labor market*, *prices*, and *trade/energy*. These subclasses are less "structural" than the originals but enrich the narrative content and show how the model can also serve as a supervised topic model.

In *Narrater-mixed* we highlight the architecture's mixed-frequency properties and focus on extractive summarization. Here all text sequences $\mathbf{s}_t$ are intra-quarterly, but only one is informative for predicting the business cycle, while $\mathbf{x}_{t_{ts}}^{ts}$ is observed monthly. Because text and time series are non-overlapping, this model predicts only the latent business cycle index rather than reconstructing the full history, but enables efficient filtering of high-frequency text via cross-attention, with article selection replacing the benchmark classification task. As such, the RL approach adopted for news summation becomes redundant for this model specification. Appendix B.5 provides further details. Allowing simultaneous classification, selection, and overlapping time windows is possible but would require substantially more memory.

Training and validation data are generated with the simulation scheme in Section 4.1, with minor modifications described in Appendix B.4.

Table A.2 (Appendix A) reports the different model configurations' performance on the validation data. Of main interest here is the classification task. As discussed earlier, the benchmark model achieves close to perfect accuracy. Although the alternative model configurations perform different classification (*Narrater-topic*) or text selection (*Narrater-mixed*) tasks, their accuracy is also close to one, exemplifying how these model configurations can efficiently identify user-defined news topics or be used to automatically perform accurate extractive news summaries.

For regression tasks, both alternatives perform worse than the benchmark in predicting sentiment, and the *Narrater-mixed* also underperforms on the business cycle. This decline is not due to poor handling of monthly time series - Figure A.7 (Appendix A) shows that cross-attention assigns weight across the full monthly sequence - but to the removal of structural decomposition (supply, demand, noise) in the output which previously provided implicit regularization via multi-task learning (Caruana, 1997; Ruder, 2017). The last column in Table A.2 substantiates this argument by showing how predictive performance drops when the benchmark *Narrater* model is estimated without the classification and sentiment regression heads.

Permutation tests (Figure A.8, Appendix A) mirror those in Section 5.2. Increasing classification complexity heightens the importance of both data modalities, whereas in *Narrater-mixed* permuting time series does not affect sentiment predictions - a direct result of attaching the sentiment head to the encoder rather than the decoder (see Appendix

B.5).

Finally, to illustrate the *Narrater-topic* output more explicitly, Figure A.9 (Appendix A) reports the fraction of articles classified as one of the four topics across time when the model is used to process the non-synthetic data: We do, for example, observe an elevated focus on politics and regulations following the introduction of inflation targeting and the EU enlargement in the early 2000s, a larger fraction of labor market oriented news during turbulent times such as the financial crisis and Covid-19 period, and a substantial increase in articles about prices and inflation towards the latter part of the sample.

7 Conclusion

Human perception and decision-making are inherently multimodal; traditional empirical models in economics, however, are largely unimodal. In macroeconomics, for example, modern time-series tools deliver high-quality business-cycle estimates and structural decompositions of its fluctuations, but typically lack a narrative interpretation that textual data might help provide. Against this background, we explore how recent advances in deep learning and NLP - specifically, encoder–decoder Transformer architectures - can be used to construct a multimodal supervised filter for narrative attribution.

While inherently domain agnostic, we call our proposed model the *Narrater* and demonstrate how it jointly processes (news) text and (macroeconomic) time series via contextualized embedding representations. At the core of this framework is the Transformer-based attention mechanism, which integrates modalities with potential different frequencies and reporting lags while preserving temporal context. This makes the architecture well suited to macro settings with asynchronous and high-dimensional information. In the process, we show how the model can be used together with Reinforcement Learning techniques to produce summaries of the high-frequency and high-dimensional flow of textual information encountered in real-word contexts.

Empirically, on Norwegian data, the model yields structural decompositions and narratives that align with well-known historical episodes; it also attains quasi real-time predictive performance on par with, or modestly better than, competitive benchmarks. Permutation tests, ablations, and hyperparameter experiments show that both data modalities matter, larger embeddings improve fit within resource limits, fine-tuning the language encoder is critical, and multi-task learning improves model performance.

Our contribution is foremost methodological: a general multimodal supervised filtering framework for measurement and narrative attribution. We do not claim model-free identification of causal relationships from multimodal data. Rather, we attribute narratives consistent with a maintained structural view. This is both a strength and limitation.

Limitations include reliance on labels and dynamics derived from underlying structural constraints. However, this is also the model's strength because it makes the framework general and easily adaptable to different user needs and structural views.

Going forward we envision numerous improvements and extensions based on this type of model architecture, which also can be useful in other settings where text (or other unstructured streams) co-evolves with numeric time series.

References

Abdolmaleki, A., J. T. Springenberg, D. Tateo, T. Degris, N. Heess, and M. A. Riedmiller (2018). Maximum a posteriori policy optimisation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Angeletos, G.-M., F. Collard, and H. Dellas (2020). Business-Cycle Anatomy. *American Economic Review 110*(10), 3030–70.

Antolín-Díaz, J. and J. F. Rubio-Ramírez (2018). Narrative sign restrictions for SVARs. *American Economic Review 108*(10), 2802–29.

Arias, J. E., J. F. Rubio-Ramirez, and D. F. Waggoner (2018). Inference Based on Structural Vector Autoregressions Identified With Sign and Zero Restrictions: Theory and Applications. *Econometrica 86*(2), 685–720.

Aruoba, S. B., F. X. Diebold, and C. Scotti (2009). Real-Time Measurement of Business Conditions. *Journal of Business & Economic Statistics 27*(4), 417–427.

Bai, J. and P. Wang (2014). Identification theory for high dimensional static and dynamic factor models. *Journal of Econometrics 178*(2), 794–804.

Baker, S. R., N. Bloom, and S. J. Davis (2016). Measuring economic policy uncertainty. *The Quarterly Journal of Economics 131*(4), 1593–1636.

Bańbura, M., D. Giannone, and L. Reichlin (2010). Large Bayesian vector auto regressions. *Journal of applied Econometrics 25*(1), 71–92.

Banbura, M., D. Giannone, and L. Reichlin (2011). *Nowcasting*. The Oxford Handbook of Economic Forecasting (Oxford Handbooks in Economics). New York: Oxford University Press.

Barsky, R. B. and E. R. Sims (2012). Information, Animal Spirits, and the Meaning of Innovations in Consumer Confidence. *American Economic Review 102*(4), 1343–77.

Baxter, M. and R. G. King (1999). Measuring business cycles: Approximate band-pass filters for economic time series. *Review of Economics and Statistics 81*(4), 575–593.

Bernanke, B., J. Boivin, and P. S. Eliasz (2005). Measuring the effects of monetary policy: A Factor-augmented Vector Autoregressive (FAVAR) approach. *The Quarterly Journal of Economics 120*(1), 387–422.

Bertsekas, D. P. (2011). Rollout algorithms for stochastic control: A survey. *Annals of Operations Research 193*(1), 3–66.

Bertsekas, D. P. (2019). *Reinforcement Learning and Optimal Control*. Belmont, MA: Athena Scientific.

Bertsekas, D. P. and J. N. Tsitsiklis (1996). *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific.

Bjørnland, H. C. and L. A. Thorsrud (2016). Boom or gloom? Examining the Dutch disease in two-speed economies. *The Economic Journal 126*(598), 2219–2256.

Blanchard, O. (1993). Consumption and the recession of 1990-1991. *The American Economic Review 83*(2), 270–274.

Breiman, L. (2001). Random forests. *Machine learning 45*, 5–32.

Burns, A. F. and W. C. Mitchell (1946). *Measuring Business Cycles*. Number 46-1 in NBER Books. National Bureau of Economic Research, Inc.

Burri, M. and D. Kaufmann (2020). A daily fever curve for the Swiss economy. *Swiss Journal of Economics and Statistics 156*, 1–11.

Bybee, L., B. Kelly, A. Manela, and D. Xiu (2024). Business news and business cycles. *The Journal of Finance 79*(5), 3105–3147.

Carriero, A., D. Pettenuzzo, and S. Shekhar (2024). Macroeconomic forecasting with large language models. *arXiv preprint arXiv:2407.00890*.

Caruana, R. (1997). Multitask learning. *Machine learning 28*(1), 41–75.

Chahrour, R., K. Nimark, and S. Pitschner (2021). Sectoral Media Focus and Aggregate Fluctuations. *American Economic Review 111*(12), 3872–3922.

Cochrane, J. H. (1994). Shocks. *Carnegie-Rochester Conference Series on Public Policy 41*, 295–364.

Das, A., W. Kong, R. Sen, and Y. Zhou (2023). A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*.

Dell, M. (2024). Deep learning for economists. Technical report, National Bureau of Economic Research.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Diebold, F. X. and R. S. Mariano (1995). Comparing predictive accuracy. *Journal of Business & Economic Statistics 13*(3), 253–63.

Estrella, A. and G. A. Hardouvelis (1991). The term structure as a predictor of real economic activity. *The Journal of Finance 46*(2), 555–576.

Evans, M. D. D. (2005). Where Are We Now? Real-Time Estimates of the Macroeconomy. *International Journal of Central Banking 1*(2), 127–175.

Fisher, A., C. Rudin, and F. Dominici (2019). All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research 20*(177), 1–81.

Forni, M., M. Hallin, M. Lippi, and L. Reichlin (2000). The generalized dynamic-factor model: Identification and estimation. *Review of Economics and statistics 82*(4), 540–554.

Friedman, M. and A. J. Schwartz (1963). *A Monetary History of the United States, 1867-1960*. Princeton University Press.

Gambacorta, L., B. Kwon, T. Park, P. Patelli, and S. Zhu (2024). Cb-lms: language models for central banking. Technical report, Bank for International Settlements.

Giannone, D., L. Reichlin, and D. Small (2008). Nowcasting: The real-time informational content of macroeconomic data. *Journal of Monetary Economics 55*(4), 665–676.

Glorot, X. and Y. Bengio (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press.

Gorodnichenko, Y., T. Pham, and O. Talavera (2023). The voice of monetary policy. *American Economic Review 113*(2), 548–84.

Gruver, N., M. Finzi, S. Qiu, and A. G. Wilson (2023). Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems 36*, 19622–19635.

Haarnoja, T., A. Zhou, P. Abbeel, and S. Levine (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 1861–1870.

Hamilton, J. D. (2018). Why you should never use the Hodrick-Prescott filter. *Review of Economics and Statistics 100*(5), 831–843.

Hansen, S. and M. McMahon (2016). Shocking language: Understanding the macroeconomic effects of central bank communication. *Journal of International Economics 99*(S1), 114–133.

Hansen, S., M. McMahon, and A. Prat (2018). Transparency and deliberation within the fomc: A computational linguistics approach. *The Quarterly Journal of Economics 133*(2), 801–870.

Hodrick, R. J. and E. C. Prescott (1997). Postwar U.S. Business Cycles: An Empirical Investigation. *Journal of Money, Credit and Banking 29*(1), 1–16.

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

Kappen, H. J. (2005). Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment* (11), P11011.

Kummervold, P. E., J. De la Rosa, F. Wetjen, and S. A. Brygfjeld (2021). Operationalizing a National Digital Library: The Case for a Norwegian Transformer Model. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, Reykjavik, Iceland, pp. 20–29. Linköping University Electronic Press, Sweden.

Larsen, V. H. and L. A. Thorsrud (2019). The value of news for economic developments. *Journal of Econometrics 210*(1), 203 – 218.

Larsen, V. H., L. A. Thorsrud, and J. Zhulanova (2021). News-driven inflation expectations and information rigidities. *Journal of Monetary Economics 117*, 507–520.

Li, L., W. Chu, J. Langford, and R. E. Schapire (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, pp. 661–670. ACM.

Lim, B., S. Ö. Arık, N. Loeff, and T. Pfister (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting 37*(4), 1748–1764.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81.

Liu, Z., D. Huang, K. Huang, Z. Li, and J. Zhao (2021). Finbert: A pre-trained financial language representation model for financial text mining. In *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, pp. 4513–4519.

Lu, J., D. Batra, D. Parikh, and S. Lee (2019). Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems 32*.

Makhzani, A., J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey (2015). Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

Marcellino, M., M. Porqueddu, and F. Venditti (2016). Short-Term GDP Forecasting With a Mixed-Frequency Dynamic Factor Model With Stochastic Volatility. *Journal of Business & Economic Statistics 34*(1), 118–127.

Mariano, R. S. and Y. Murasawa (2003). A new coincident index of business cycles based on monthly and quarterly series. *Journal of Applied Econometrics 18*(4), 427–443.

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Minaee, S., N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao (2021). Deep learning–based text classification: a comprehensive review. *ACM computing surveys (CSUR) 54*(3), 1–40.

Nikolenko, S. I. (2021). *Synthetic data for deep learning*, Volume 174. Springer.

Nimark, K. P. and S. Pitschner (2019). News media and delegated information choice. *Journal of Economic Theory 181*, 160 – 196.

Pennington, J., R. Socher, and C. Manning (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1532–1543. Association for Computational Linguistics.

Peters, J., K. Mülling, and Y. Altün (2010). Relative entropy policy search. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI 10)*, pp. 1607–1612.

Ramey, V. A. (2011). Identifying government spending shocks: It's all in the timing. *The Quarterly Journal of Economics 126*(1), 1–50.

Requeima, J., J. Bronskill, D. Choi, R. Turner, and D. K. Duvenaud (2024). Llm processes: Numerical predictive distributions conditioned on natural language. *Advances in Neural Information Processing Systems 37*, 109609–109671.

Romer, C. D. and D. H. Romer (1989). Does monetary policy matter? A new test in the spirit of Friedman and Schwartz. *NBER macroeconomics annual 4*, 121–170.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Schularick, M. and A. M. Taylor (2012). Credit booms gone bust: monetary policy, leverage cycles, and financial crises, 1870–2008. *American Economic Review 102*(2), 1029–1061.

Schulman, J., S. Levine, P. Abbeel, M. Jordan, and P. Moritz (2015). Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 1889–1897.

Shapiro, A. H., M. Sudhof, and D. J. Wilson (2022). Measuring news sentiment. *Journal of Econometrics 228*(2), 221–243.

Shiller, R. J. (2017). Narrative economics. *American Economic Review 107*(4), 967–1004.

Sims, C. A. (1980). Macroeconomics and reality. *Econometrica 48*(1), 1–48.

Smets, F. and R. Wouters (2007). Shocks and frictions in US business cycles: A Bayesian DSGE approach. *American economic review 97*(3), 586–606.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research 15*(1), 1929–1958.

Stock, J. and M. Watson (2012). Disentangling the channels of the 2007-2009 recession. *Brookings Papers on Economic Activity Spring 2012*, 81–135.

Stock, J. H. and M. W. Watson (1989). New indexes of coincident and leading economic indicators. In O. J. Blanchard and F. Stanley (Eds.), *NBER Macroeconomics Annual*, NBER Chapters, pp. 351–394. Cambridge, MA: The MIT Press.

Stock, J. H. and M. W. Watson (2002). Macroeconomic forecasting using diffusion indexes. *Journal of Business & Economic Statistics 20*(2), 147–62.

Stock, J. H. and M. W. Watson (2016). Dynamic factor models, factor-augmented vector autoregressions, and structural vector autoregressions in macroeconomics. In J. B. Taylor and H. Uhlig (Eds.), *Handbook of Macroeconomics*, Volume 2, pp. 415–525. Elsevier.

Thorsrud, L. A. (2018). Words are the new numbers: A newsy coincident index of the business cycle. *Journal of Business & Economic Statistics*, 1–17.

Todorov, E. (2007). Linearly-solvable markov decision problems. *Advances in Neural Information Processing Systems (NeurIPS)*, 1369–1376.

Tsai, Y.-H. H., S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov (2019). Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the conference. Association for computational linguistics. Meeting*, Volume 2019, pp. 6558.

Turc, I., M.-W. Chang, K. Lee, and K. Toutanova (2019). Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin (2017). Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems*.

Veldkamp, L. L. (2011). *Information choice in macroeconomics and finance*. Princeton University Press.

Wen, Q., T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun (2022). Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*.

Woo, G., C. Liu, A. Kumar, C. Xiong, S. Savarese, and D. Sahoo (2024). Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592*.

Xu, P., X. Zhu, and D. A. Clifton (2023). Multimodal learning with transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence 45*(10), 12113–12132.

Zhang, H., P. S. Yu, and J. Zhang (2024). A systematic survey of text summarization: From statistical methods to large language models. *ACM Computing Surveys*.

Ziebart, B. D. (2010). *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. Ph. D. thesis, Carnegie Mellon University.

Appendices for online publication

Appendix A Additional results

Table A.1. Correspondence and correlation between *Narrater-greedy* and different implementations of *Narrater-RL*; default, only RMSE rewards ($\lambda_F = \lambda_N = \lambda_P = 0$), and only RMSE and NPRIOR rewards (($\lambda_F = \lambda_P = 0$).

	Default	*Only RMSE rewards*	*Only RMSE and NPRIOR rewards*
Class/selection	0.65	0.72	0.70
Business cycle	0.97	0.95	0.95

Table A.2. Alternative model specifications and validation data performance. The sentiment (Sent.) and business cycle (BusC.) tasks are similar across models while the classification tasks are different.

	Narrater	*Narrater-topic*	*Narrater-mixed*	*Narrater-noMTL*
Class/selection (accuracy)	0.99	0.99	0.99	
Sent. (RMSE)	0.02	0.03	0.03	
BusC. (RMSE)	0.18	0.18	0.19	0.19

(a) Business cycle quarterly and ex-post **(b)** Zoomed-in

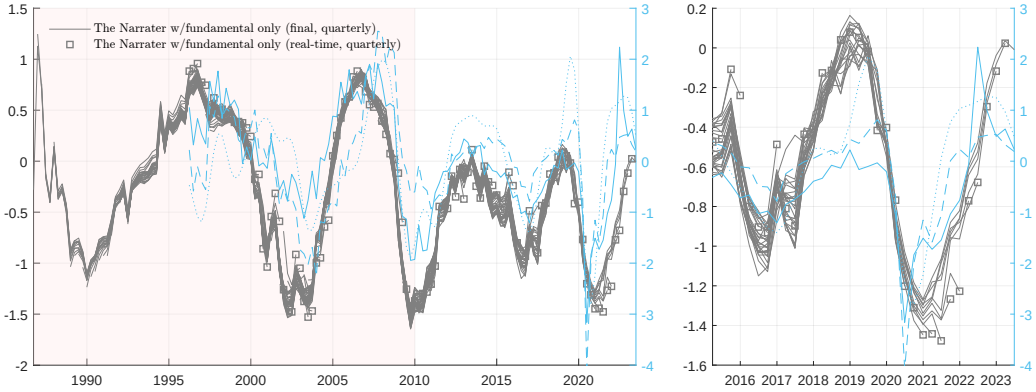


Figure A.1. The graphs report the real-time (filtered) estimates $\hat{y}^{(\hat{i})}_{T_f+1,reg}$ together with the smoothed versions $\hat{y}^{(\hat{i})}_{1:T_f+1,reg}$. The white areas are time periods with data not seen during training and validation.

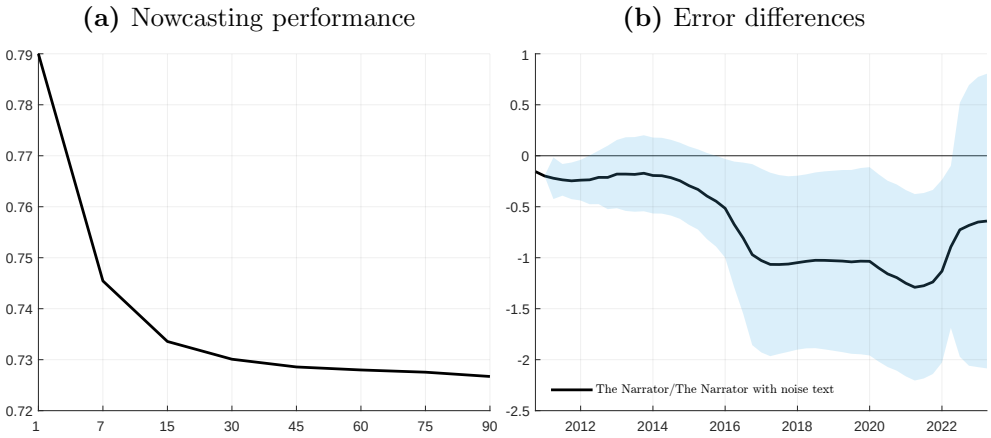


Figure A.2. Figure A.2a reports the predictive performance of the *Narrater* relative to the DFM, when the former predictions are constructed as averages of 1 - 90 daily (x-axis) predictions. Figure A.2b reports the cumulative squared prediction error difference between the *Narrater* and a version of the model using a generic and recurrent text sequence. A falling value implies a relative *Narrater* improvement. The shaded areas are 95% confidence intervals of differences in predictive performance, computed following Diebold and Mariano (1995) with HAC corrected standard errors.

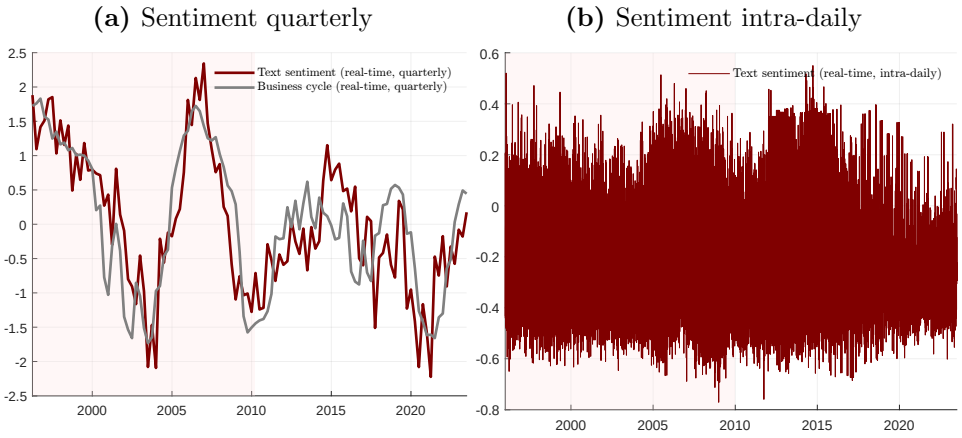


Figure A.3. Figure A.3a reports the normalized quarterly mean of the *Narrater*'s real-time sentiment predictions. Figure A.3b reports the quarterly real-real time estimates of $\hat{\mathbf{y}}^{(i)}_{T_f+1,sent}$ implied by all incoming news. The white areas are time periods with data not seen during training and validation.

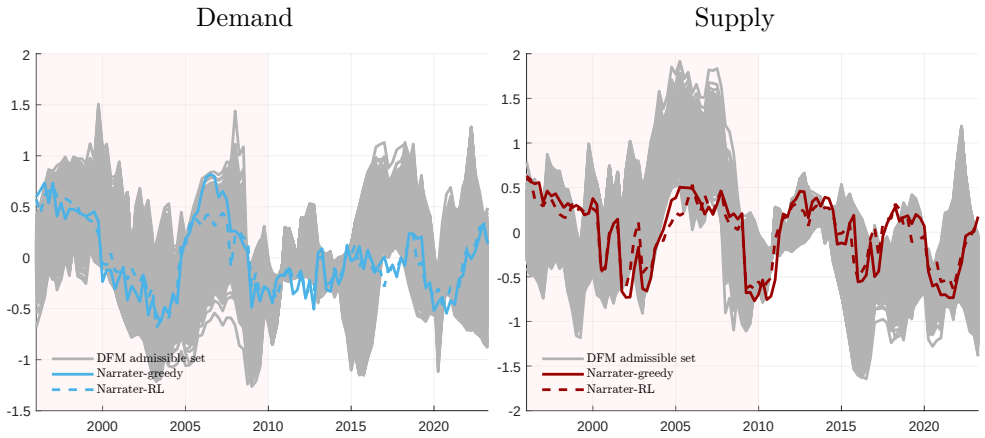


Figure A.4. Real-time (predictive) demand and supply decomposition produced by the *Narrater* together with the similar objects produced by the DFM, described in Appendix B. For the DFM these are ex-post estimates (because they are computed with data for the whole sample available). Since the DFM is identified using sign restrictions, we report the set of historical shock decompositions associated with the impulse response functions fulfilling the sign restrictions, i.e., the admissible set. The white areas are time periods with data not seen during training and validation of the *Narrater*.

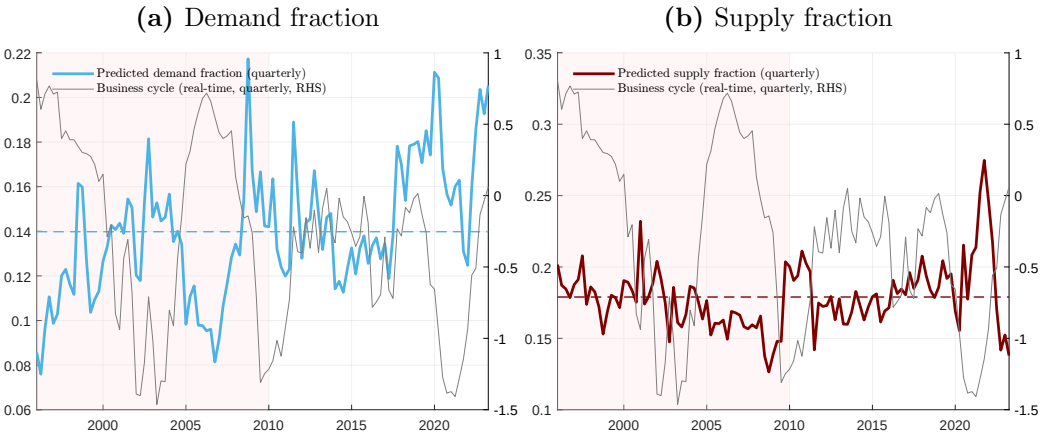


Figure A.5. Fraction of news classified as either demand or supply using $\hat{\mathbf{y}}^{(i)}_{T_f+1,class}$ and each quarterly time segment. The white areas are time periods with data not seen during training and validation.

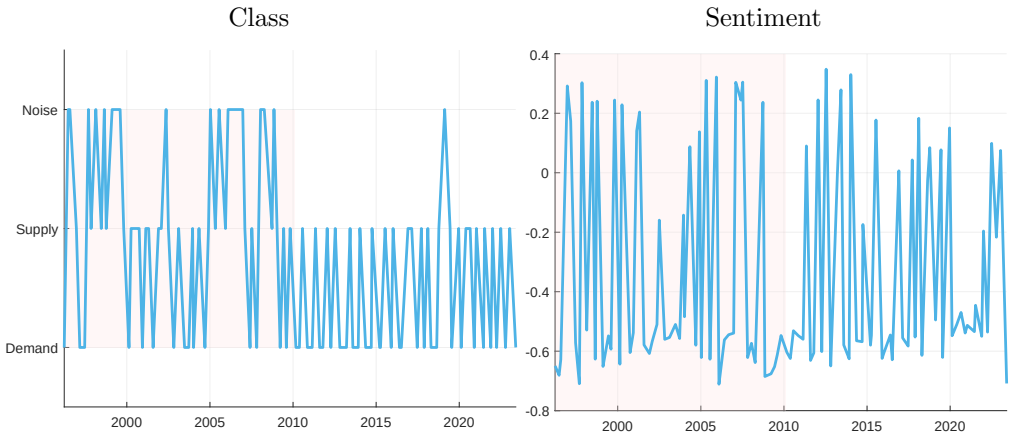


Figure A.6. Quarterly real-time estimates of the class and sentiment of article $\hat{i}$. The white areas are time periods with data not seen during training and validation.

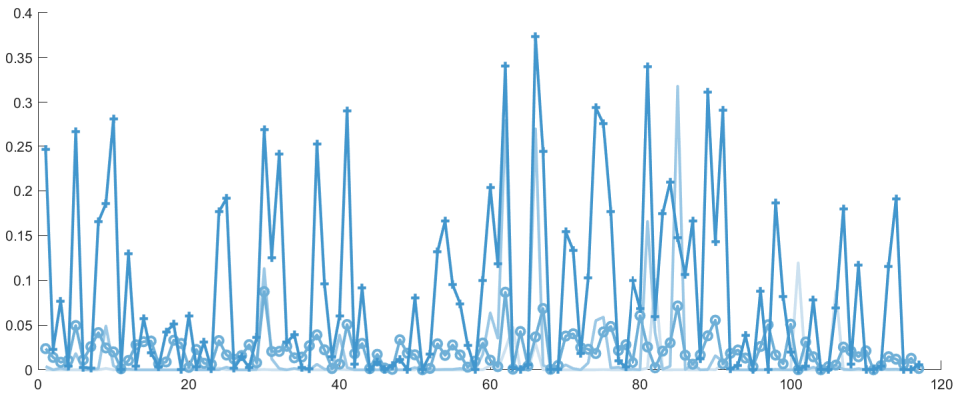


Figure A.7. Attention scores from equation (6) and the *Narrater-mixed* model for period $T_f + 1$ and one particular validation set observation. The four lines reflect the different attention heads, while the x-axis reflect the sequence length of the monthly data.

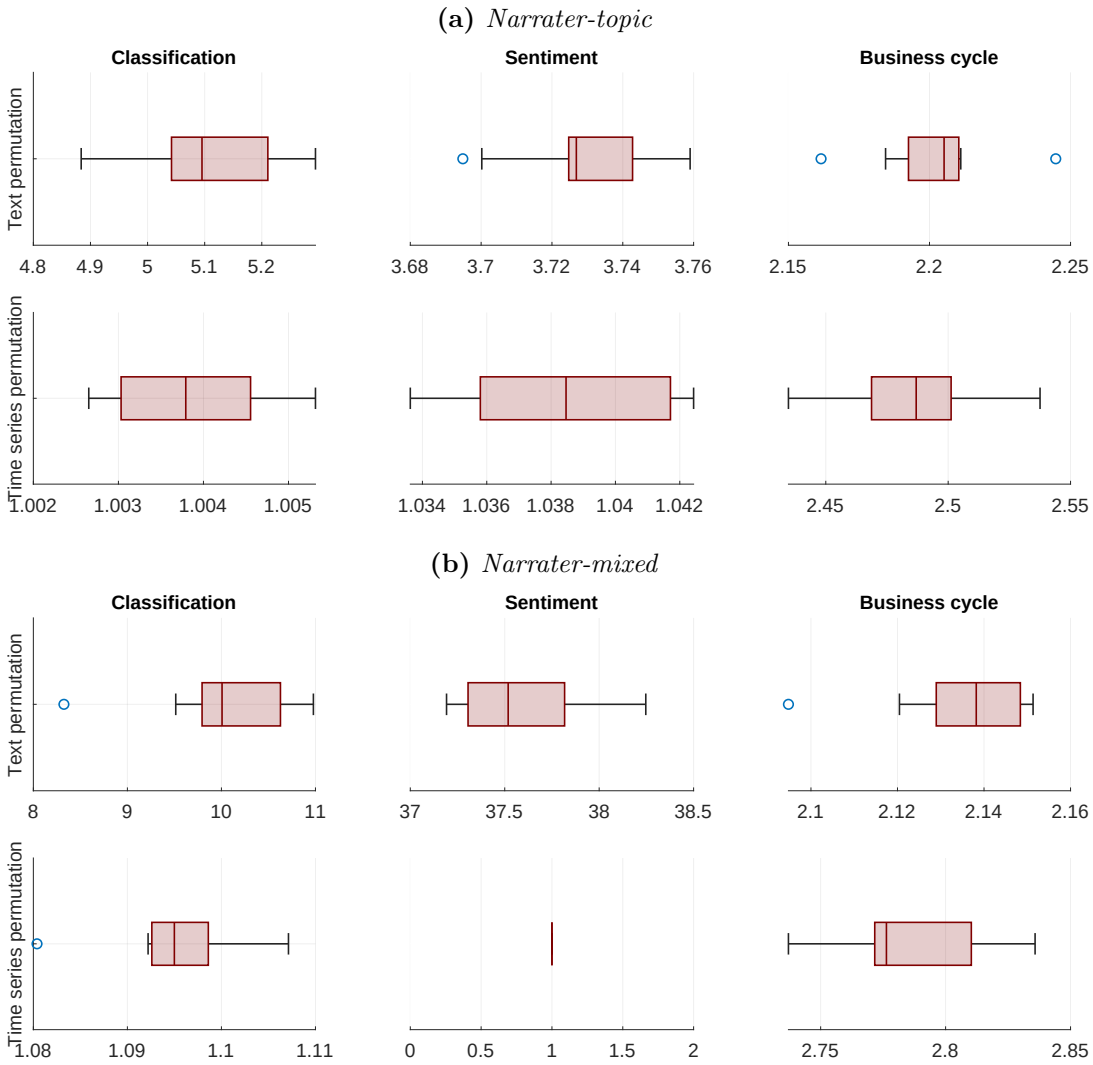


Figure A.8. Traditional box plots of relative performance difference from permutation tests. A value above one indicates that the model performs worse when the particular data input is randomized.

Figure A.9. Fraction of news articles, aggregated to quarterly frequency, classified by the *Narrator-topic* model to be about policy/regulation, the labor market, prices, or trade/energy. The white areas are time periods with data not seen during training and validation.

Appendix B Generating training data

As described in Section 4.1, synthetic training data is generated by simulating time series data from an estimated Dynamic Factor Model and mapping text to time series using the implied historical shock decomposition. Prior to this, the class and sentiment of a large set of text data used for training and validation is automatically recorded using embeddings. Below we provide the details of each step.

B.1 Business cycle simulations

To simulate synthetic macroeconomic data we assume a state space system where the dynamics of the observable data are described by a set of latent state variables evolving according to an autoregressive process. In particular, let $\mathbf{x}_{t_{ts}}^{ts}$ be a $D \times 1$ vector of observed

time series at time t_{ts} and $\mathbf{f}_{t_{ts}}$ a $r \times 1$ vector of latent factors, such that:

$$\begin{aligned} \mathbf{x}_{t_{ts}}^{ts} &= \mathbf{\Lambda}\mathbf{f}_{t_{ts}} + \mathbf{u}_{t_{ts}}, \quad \mathbf{u}_{t_{ts}} \sim \mathcal{N}(0, \mathbf{R}), \\ \mathbf{f}_{t_{ts}} &= \mathbf{\Phi}(L)\mathbf{f}_{t_{ts}-1} + \mathbf{H}\varepsilon_{t_{ts}}, \quad \varepsilon_{t_{ts}} \sim \mathcal{N}(0, \mathbf{Q}), \end{aligned} \tag{17}$$

where $\mathbf{\Lambda}$ is a $D \times r$ matrix of factor loadings, $\mathbf{u}_{t_{ts}}$ is a $D \times 1$ vector of idiosyncratic errors (or noise), $\mathbf{\Phi}(L)$ is a lag polynomial matrix of autoregressive coefficients. $\mathbf{R}$ and $\mathbf{Q}$ are the variance-covariance matrices of the error terms. We assume a diagonal structure for $\mathbf{R}$ but not for $\mathbf{Q}$. Moreover, we write the transition equation in (17) using the companion form, and let q denote the number of latent factors such that $q \leq r$ and the number of lags of the factors is p (with $p \times q = r$). Thus, $\varepsilon_{t_{ts}}$ is a $q \times 1$ vector of innovations (or shocks) to the factors and $\mathbf{H}$ is a $r \times q$ selection matrix.

For our particular experiment, we focus on two fundamental shocks ($q = 2$), namely aggregate demand and supply shocks, and identify the latent factors as an aggregate business cycle index and an underlying inflation measure. The factors are identified following the unit identification scheme described in, e.g., Bai and Wang (2014), implying that the upper left $q \times q$ block of $\mathbf{\Lambda}$ is the identify matrix. The relationship between the reduced form errors and the structural shocks is given by $\varepsilon_{t_{ts}} = \mathbf{A}\mathbf{e}_{t_{ts}}$. Because $(\mathbf{A}\mathbf{e}_{t_{ts}})(\mathbf{A}\mathbf{e}_{t_{ts}})^\top = \mathbf{Q}$, identification is achieved from $E(\mathbf{e}_{t_{ts}}\mathbf{e}_{t_{ts}}^\top) = \mathbf{I}$ and by restricting the elements of $\mathbf{A}$ using sign restrictions following Arias et al. (2018). Accordingly, $\mathbf{A}$ is restricted such that the following relationship holds:

$$\begin{bmatrix} \varepsilon_{1,t_{ts}} \\ \varepsilon_{2,t_{ts}} \end{bmatrix} = \begin{bmatrix} + & + \\ + & - \end{bmatrix} \begin{bmatrix} e_{t_{ts}}^d \\ e_{t_{ts}}^s \end{bmatrix} \tag{18}$$

where a positive demand innovation moves prices and the business cycle in the same direction while a positive supply shock moves these factors in the opposite direction.

To simulate training data from the model we first estimate it using the data described in Section 4.3 and covering the sample 1986 to 2010. We allow for $p = 8$ lags and apply a two-step estimation procedure, similar to in Bernanke et al. (2005): Common factors are first estimated using Principal Components Analysis (PCA) and identified as described above. Next, these factors are included in the VAR to estimate the time series dynamics using OLS. Although sign restrictions only give set identification we focus on the structural parameters associated with the median draw when sampling admissible impulse responses following (18).

In total we simulate 60000 time series segments from the model. The length of each segment is determined by the memory constraint of the *Narrater*, while we always remove the first five years of simulated data to avoid dependence on starting values. Simulating data directly from (17) produces synthetic data for the observable macroeconomic indicators ($\mathbf{x}_{t_{ts}}^{ts}$) and the latent factors, including the business cycle, where $\mathbf{x}_{t_f}^{f} = \mathbf{f}_{1,t_{ts}}$. To

Table B.1. The class and sentiment sequences and key-words used to generate benchmark embeddings and Boolean search, respectively. For the key-word lists only the most relevant terms are reported. See the text for details.

	Text sequence	Key-words
Demand	*"demand, sentiment, expectations"*	*shopping, income, demand, consumption, wage growth, tourism, interest rate hike, wage settlement, transfers, consumer loan,...*
Supply	*"productivity, efficiency, innovation, technology"*	*technology, labor, strike, sanctions, regulation, storm, sick leave, leave of absence, recruitment, overcapacity,...*
Positive	*"positive, strong, good"*	
Negative	*"negative, weak, bad"*	

also produce synthetic data for the associated historical shock decomposition we use:

$$\mathbf{f}_{t_{ts}} = \sum_{j=0}^{t} \Psi_j \mathbf{A} \mathbf{e}_{t_{ts}-j},$$

where Ψ_j are the impulse response coefficient matrices (from the moving average (MA) representation of the VAR), and consider the effect of each shock individually on the implied decomposition of $\mathbf{f}_{1,t_{ts}}$. To also capture the noise component we use the element in $\mathbf{u}_{t_{ts}}$ associated with GDP growth.

B.2 Creating labeled text data

To create labeled text data that can be mapped to the simulated time series dynamics we proceed in three steps. First we define a set of key-word sequences for the demand and supply categories and good and bad sentiment. These are listed in Table B.1. Next, we use the OpenAI API, and the `text-embedding-3-small` model, to produce embedding representations of each of these benchmark text sequences. The intuition for this approach is that the embedding representations capture shared context, and thus captures broader aspects and abstractions than the key-words themselves. At the same time, using key-word-based sequences instead of, e.g., prototype sentences, reduces the risk of introducing unintended sentiment, framing, or policy biases in the categorizations. Finally, we randomly sample text sequences from the DN news corpus described in Section 4.3, but only consider news reported within the sample 1986 to 2010, and compute each sampled article's similarity with the benchmark embeddings.

More formally, let $\mathcal{X}$ denote the set of sampled news texts from the DN corpus (restricted to 1986–2010), and let $\phi : \mathcal{T} \to \mathbb{R}^d$ be the embedding map induced by the

`text-embedding-3-small` model. Likewise, let $\mathcal{B}^{\text{cls}} = \{b_c\}_{c=1}^{C-1}$ be the benchmark keyword sequences used for categorization (e.g., demand and supply), and let $\mathcal{B}^{\text{sent}} = \{b^+, b^-\}$ be the two sentiment benchmarks used for scoring. Then, for $x \in \mathcal{X}$ and $c \in \{1, \ldots, C-1\}$ we define class-wise and sentiment similarity as:

$$s_c(x) \;=\; \cos\big(\phi(x), \phi(b_c)\big) \qquad s^+(x) \;=\; \cos\big(\phi(x), \phi(b^+)\big), \qquad s^-(x) \;=\; \cos\big(\phi(x), \phi(b^-)\big),$$

where, for any $u, v \in \mathbb{R}^d$, $cos(u, v)$ is cosine similarity. Then, to motivate distinct article classes we set a class-specific threshold:

$$\tau_c \;=\; Q_{0.9}(\{\, s_c(x) : x \in \mathcal{X} \,\}), \qquad c = 1, \ldots, C-1,$$

where $Q_{0.9}(\cdot)$ denote the empirical 90th percentile, and assign an article x to a distinct class if its top score clears that class's threshold:

$$y(x)_{class} \;=\; \begin{cases} c^\star(x), & \text{if } s_{c^\star(x)}(x) \;\geq\; \tau_{c^\star(x)} \text{ and } c^\star(x) \text{ is unique,} \\ \text{noise}, & \text{otherwise,} \end{cases}$$

with $c^\star(x) := \arg\max_c s_c(x)$. Because τ_k is the 90th percentile for each class, the majority of sampled texts are labeled as noise.[12] We have experimented with using different thresholds, finding that if we do not define a relatively high threshold value, labeled articles will be difficult to discriminate. Finally, independently of the class assignment, we compute:

$$sd(x) \;=\; s^+(x) - s^-(x), \qquad y(x)_{sent} \;=\; \tanh\big(sd(x)\big) \;\in (-1, 1),$$

and use $y(x)_{sent}$ as a measure of the article's sentiment.

The procedure for creating labeled text data is simple and fast, and allows labeling a large set of articles automatically. Arguably, the procedure could be improved and refined in a number of directions. As an alternative we have also explored creating labeled text data using a Boolean search-based approach. First, two domain-specific dictionaries of Norwegian terms were manually compiled. One dictionary was intended to capture *aggregate demand* news and the other *aggregate supply* news. The final demand dictionary contains 72 unique tokens, while the supply dictionary contains 129. The 10 most frequent words in each list are displayed in the last column of Table B.1. The full lists are available upon request.

Second, each article was lower-cased, stripped of punctuation, and tokenised. We then counted the number of hits from the demand and supply dictionaries for every article, producing two variables `demand_count` and `supply_count`. An article is labelled as **demand**

[12] To avoid poor performance for the minority classes we use a weighted loss function when training the model. In particular, a class is assigned a weight inversely proportional to its frequency in the training data. These weights are then used to scale the loss contribution of each training sample so that misclassifying a minority class costs more than misclassifying a majority class.

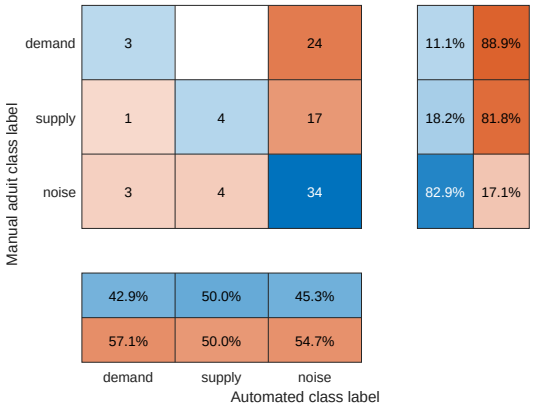


Figure B.1. Manual audit's confusion matrix when a Boolean search-based approach has been used to create labeled text data.

when `demand_count`>0 and `supply_count`$=0$, and as **supply** when `supply_count`>0 and `demand_count`$=0$. Articles that contain keywords from *both* lists or from *neither* list are classified as noise. To avoid extremely long feature articles with many tangential topics, we discard texts longer than 1 000 words before applying the rule. The Boolean approach therefore focuses on relatively short, single-topic news items.

Figure B.1 reports the results from the same type of manual audit as described in Section 4.2, but now applied to the text data labeled using the Boolean search-based approach. The overall accuracy is only 0.46, compared to 0.79 for the embedding-based approach. Similarly, the results presented in Section 6 of the main paper are positive, suggesting that the embedding-based labeling procedure is reasonable. We thus leave it for future research to investigate potentially more adequate methods for automatically creating labeled text data.

B.3 Simulation robustness

To examine how training data assumptions shape the model's use of multimodal information, we have simulated alternative datasets that explicitly manipulate the sharpness and noise in the text–time-series mapping. Instead of linearly linking the probability of drawing each text class to the size of the historical shock decompositions, we apply a temperature-controlled softmax with $\tau = 0.2$ or $\tau = 0.6$. In addition, for each τ we introduce noise by randomly breaking the link between the true class and the chosen text, governed by $r \sim \text{Bernoulli}(p_{\text{rel}})$ with $p_{\text{rel}} = 0.98$ or 1.00.

The additional rows in Figure 4 (Section 5.2) report permutation tests for models trained on these alternative data. A clear pattern emerges: the sharper and more deterministic the mapping between shock decompositions and texts, the more predictive performance deteriorates when the time-series input is permuted. Similarly, injecting noise into the textual mapping shifts the model's reliance toward the time series in the

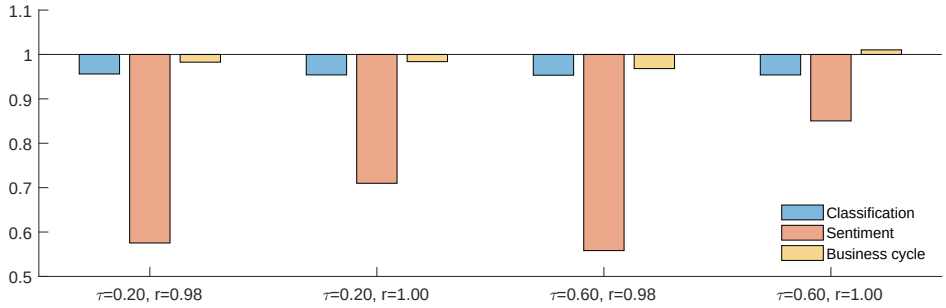


Figure B.2. Performance scores relative to the benchmark *Narrater* model. All model versions are evaluated on the same validation data. A value below one indicates that the alternative model is better.

text-prediction tasks.

Drawing on the multi-head attention mechanism in the Transformer architecture, an interesting avenue for future research is to train a single model directly on data generated under varying assumptions. Such a setup would allow different attention heads to specialize in distinct regimes, potentially yielding a final model that is robust across a broader range of data-generating processes. Speaking to this argument, Figure B.2 reports relative predictive scores when all models analyzed in Figure 4 are evaluated on a test data set containing random draws from all five simulation experiments (including the benchmark described in Section 4.1). Interestingly, under these assumptions, the benchmark model is worse than all of the alternatives (except in one case), and the best model would be the one trained on data using the $\tau = 0.6$ and $p_{\text{rel}} = 0.98$ assumption.

B.4 Extensions

To create labeled text data for the *Narrater-topic* experiments conducted in Section 6.4 we follow the same type of embedding-based labeling procedure as above. We first define text sequences for each of the topic categories of interest. These are listed in Table B.2. Then, for each sampled text sequence we again use the "text-embedding-3-small" model to obtain its embedding representation and compute the cosine similarity with the four different topic benchmark embdeddings. Finally, we attach a new class label for each text depending on the index of the maximum cosine similarity score and the original classifications. I.e., we obtain four subclasses for both "demand" and "supply".

For the *Narrater-mixed* experiment we construct a mapping between the text and time series data so that the sentiment of the true text is as close to $tanh(y_{T_f+1,reg})$ as possible, where $y_{T_f+1,reg}$ is the business cycle index. Then, to construct a set of false articles for each training and validation observation we simply select the $C-1$ articles whos sentiment differs the most with the true sentiment. This clearly simplifies the classification problem. For practitioners wanting to apply this model a potentially more realistic approach for creating the set of false articles could be to randomly select articles from the same real

Table B.2. The table reports the text sequences used to generate the topic-based embeddings. See the text for details. Note that terms or translated from Norwegian to English, and that Norwegian frequently combines two or more words to create a single new word with a specific meaning. E.g. "monetary policy" is one single word in Norwegian.

	Text sequence
Policy/regulation	*"interest rate, central bank, budget, tax, regulation, monetary policy, fiscal policy"*
Labor market	*"unemployment, employment, jobs, permits, wage, labor market"*
Prices	*"inflation, consumer price index, cost of living, price level, price development"*
Trade/energy	*"export, import, trade, trade balance, oil price, energy, war, geopolitics"*

word time index as the true article.

Monthly time series data are sourced to reflect the quarterly time series when possible. We use the monthly frequency of U, CPI, SPREAD, and OIL, while the reminding variables are "replaced" by (log) yearly changes in monthly industrial production, retail sales, and money growth, and the level of OECD's business tendency survey for Norway. Then to simulate monthly $\mathbf{x}_{t_{ts}}^{ts}$ time series data we first use PCA and the monthly variables to compute one common factor, factor loadings, and the variance of the idiosyncratic noise. Because the monthly variables have somewhat different sample lengths, we use data covering the period 1993 to 2010 to derive these objects. Next we simply assume that the mean of the monthly factors are equal to the quarterly business cycle index from the the original training data, and use the estimated factor loadings, the variance of the idiosyncratic noise, and the normal distribution, to simulate monthly observations that are consistent with the dynamics of the original quarterly business cycle index.

B.5 Alternative model specifications

To keep the computational burden and memory requirements for the *Narrater-mixed* reasonable we only consider $T = C = 10$, and since the model does not give any structural decomposition of the business cycle in this configuration $R = 1$. Moreover, since we do not assume any potential temporal relationship between the C articles, equation (2) is removed and the model head in (10) is attached to the model's text encoder. Thus, the sentiment of each of the C articles is predicted, but only one article is relevant for the business cycle at time $T_f + 1$. To facilitate this type of selection we further let the text classification head be a direct function of the time-series-text-embedding cross-attention scores.

Letting $\mathbf{q}_2^f$ denote the $1 \times d$ decoder query vector for time period $T_f + 1$, the scores

are:

$$\mathbf{a} = \text{softmax}\left(\frac{\mathbf{q}_2^f \mathbf{K}^{txt^\top}}{\sqrt{d}}\right) \in \mathbb{R}^{1\times T}, \tag{19}$$

and $\mathbf{a}$ replaces $\mathbf{h}^f_{(3)}$ in (9). This prevents the network from mixing the information in the two data modalities before identifying the relevant text. In contrast, using the aggregated attention output from (20) as input for the text selection head would imply that the model looses the individual identity of the texts before trying to select the correct one. As before $T_f = 40$ quarters, but $T_{ts} = 120$ monthly observations, and we assume that $t_{ts} = t_f + 1$ such that (realistically) monthly observations are available for the current quarter.

Appendix C Transformers and BERT

In the proposed architecture we combine a BERT language encoder with a Transformer-based encoder-decoder structure for processing time series data. Below we shortly describe the general Transformer block in a neural network and the BERT structure.

C.1 The Transformer

A Transformer block, sometimes called a Transformer encoder layer, is a key computational unit originally introduced in the Transformer architecture (Vaswani et al., 2017), which has since become foundational in a wide range of sequence modeling tasks.

At its core, a Transformer block processes an input sequence representation through a self-attention mechanism, followed by a position-wise feed-forward network (FFN). The input to the block is first normalized using a layer normalization operation. Let the input embedding sequence be represented as a matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, where n is the sequence length and d is the feature dimension. After applying layer normalization, a so-called multi-head attention module computes a weighted sum of values across positions, enabling each element to attend to other relevant elements in the sequence. By allowing for multiple heads, the model can focus on information from different representation subspaces at different positions, capturing a richer variety of dependencies compared to single-head attention. This is achieved by projecting the input $\mathbf{X}$ into three matrices: queries $\mathbf{Q} = \mathbf{X}\mathbf{W}_Q$, keys $\mathbf{K} = \mathbf{X}\mathbf{W}_K$, and values $\mathbf{V} = \mathbf{X}\mathbf{W}_V$, where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d_k}$ are learnable parameters, and $d_k = d/h$ for h attention heads. The scaled dot-product attention is then computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \tag{20}$$

For multi-head attention, this operation is performed h times in parallel and the results are concatenated to form a d-dimensional output. A residual connection is added to the

layer input, and the output is again normalized.

In many sequence modeling settings, attention is required to be causal, which in this setting implies that each element in the sequence can only attend to its current and preceding positions, not future ones. Causality is typically enforced by applying a masking operation to the attention matrix, ensuring that $\mathbf{QK}^\top$ entries associated with future positions are set to $-\infty$ before the softmax is applied. This restricts the flow of information and prevents "peeking" into the future, a property critical for tasks such as language modeling or time series prediction. In contrast, non-causal (or bidirectional) attention imposes no such constraints, allowing each element in the sequence to attend to both past and future elements in the sequence, which can be advantageous for tasks like machine translation or masked language modeling where full contextual information is available.

Following the attention layer, a FFN is applied to each position independently. The FFN typically consists of two linear transformations with a non-linear activation function, such as ReLU, in between:

$$\mathrm{FFN}(\mathbf{Z}) = \max(0, \mathbf{Z}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \tag{21}$$

where $\mathbf{Z} \in \mathbb{R}^{n \times d}$ is the output of the multi-head attention module, and $\mathbf{W}_1 \in \mathbb{R}^{d \times d_f}$, $\mathbf{W}_2 \in \mathbb{R}^{d_f \times d}$, $\mathbf{b}_1 \in \mathbb{R}^{d_f}$, and $\mathbf{b}_2 \in \mathbb{R}^d$ are learnable parameters. Here d_f is often chosen to be larger than d to increase representational capacity. As with the attention layer, the FFN output is combined with a residual connection and layer normalization. Thus, each Transformer block systematically refines the hidden representations through attention-based context aggregation and non-linear transformations.

C.2 BERT

BERT is a widely-used Transformer-based model for natural language understanding tasks. Introduced by Devlin et al. (2018), BERT leverages a deep stack of Transformer encoders to produce rich, contextualized embeddings of input tokens. Unlike previous popular models that relied on a unidirectional context, or did not use an attention mechanism at all (Mikolov et al., 2013; Pennington et al., 2014), BERT employs bidirectional attention, allowing each token to attend to all other tokens in both directions. This property is achieved through a masked language modeling pre-training objective, where a certain percentage of tokens are masked and the model learns to predict these masked tokens from their surrounding context.

One of the primary strengths of BERT-like models is that they are provided pre-trained on large-scale text corpus, but can be fine-tuned for a broad range of downstream tasks

with minimal architectural modification, often achieving state-of-the-art or near state-of-the-art results.[13] This transfer learning capability stems from the generality and richness of the pre-trained token representations, allowing developers and researchers to focus on task-specific data without the need for extensive task-specific model architectures. As a result, BERT has seen widespread adoption and has significantly influenced the direction and methodologies of NLP research.

BERT is provided in several model sizes to accommodate different resource constraints and performance targets. For example, $\text{BERT}_{\text{BASE}}$ consists of 12 Transformer encoder layers, each with 12 attention heads and a hidden size of 768 dimensions (totaling roughly 110 million parameters). $\text{BERT}_{\text{LARGE}}$, in contrast, uses 24 layers, 16 attention heads, and a hidden size of 1024, leading to about 340 million parameters. These models can be further adapted or distilled into smaller variants, enabling more efficient deployment on resource-limited devices.

Here we utilize $\text{BERT}_{\text{TINY}}$ (Turc et al., 2019), which present a family of more compact BERT models that trade off model size and computation for slightly reduced accuracy, allowing for more efficient deployments while still maintaining strong performance on many NLP tasks. In particular, the $\text{BERT}_{\text{TINY}}$ version we use consists of only 2 Transformer encoder layers, each with 4 attention heads and a hidden size of 128 dimensions (totaling roughly 4 million parameters).

In terms of processing a text sequence, $\text{BERT}_{\text{TINY}}$ first uses the same WordPiece tokenizer as the original BERT model.[14] Let a given input sequence consist of n tokens: $(w_1, w_2, \ldots, w_n)$, where each token w_i is drawn from a fixed vocabulary V. Then BERT first maps these tokens into a sequence of token embeddings $\mathbf{E} \in \mathbb{R}^{n \times d}$, where d is the hidden dimension and $\mathbf{E}$ is a learnable parameter matrix. In addition to token embeddings, BERT uses learnable position embeddings $\mathbf{P} \in \mathbb{R}^{n \times d}$ to encode the positional information of each token. Thus, each token w_i is represented as:

$$\mathbf{x}_i = \mathbf{e}_i + \mathbf{p}_i,$$

where $\mathbf{e}_i$ and $\mathbf{p}_i$ are the i^{th} row vectors of $\mathbf{E}$ and $\mathbf{P}$, respectively. Stacking these vectors for all $i \in \{1, \ldots, n\}$ results in the input embedding matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, which serves as the input to the first Transformer encoder layer of BERT.

To further prepare the input, and facilitate batch-processing, each input sequence of

[13] BERT was originally trained on a corpus which included the English Wikipedia (approximately 2.5 billion words) and the BookCorpus (approximately 800 million words).

[14] This tokenizer splits input text into subword units based on statistical properties derived from large-scale text corpora. By applying the same WordPiece vocabulary and tokenization strategy, $\text{BERT}_{\text{TINY}}$ maintains compatibility with BERT's input format and leverages the same robust subword segmentation to effectively handle a broad variety of words, including rare and out-of-vocabulary terms.

length n is typically padded to a fixed length $N \geq n$ by appending special padding tokens. A corresponding binary attention mask is then defined and ensures that the model only attends to non-padding tokens. In $\text{BERT}_{\text{TINY}}$ $N = 512$ is the maximum sequence length, and sequences that are longer than this will automatically be truncated. In addition to padding and masking, BERT also incorporates special tokens into the input sequences. For a single sequence or a pair of sentences, BERT prepends a $[CLS]$ token at the start of the sequence and inserts a $[SEP]$ token after each sentence. The token $[CLS]$ provides a convenient vector representation for the entire input sequence, and each $[SEP]$ helps the model distinguish between sentence boundaries. For our purpose, the former $[CLS]$ token is important because it facilitates sequence classification.

C.3 Transformers for time series

Originally developed for natural language processing, the Transformer architecture has increasingly been applied to time series modeling due to its ability to capture long-range dependencies without relying on recurrence. Indeed, by leveraging self-attention mechanisms, Transformer-based models have demonstrated strong performance across a variety of time series tasks (Wen et al., 2022).

Key challenges in applying Transformers to time series include encoding temporal structure and handling multivariate data. For the former, a common approach, adopted here, is to treat time steps as discrete tokens and apply either fixed or learned positional embeddings, analogous to token position encoding in NLP models. In particular, let $\mathbf{x}_{t_{ts}}^{ts} \in \mathbb{R}^D$ denote a D-dimensional vector of macroeconomic variables at time $t_{ts} \in \{1, \ldots, T_{ts}\}$. Each vector is then projected into a d-dimensional embedding space via:

$$\mathbf{e}_{t_{ts}}^{ts} = \mathbf{W}_{ts}\mathbf{x}_{t_{ts}}^{ts} + \mathbf{b}_{ts}, \tag{22}$$

where $\mathbf{W}_{ts} \in \mathbb{R}^{d \times D}$ and $\mathbf{b}_{ts} \in \mathbb{R}^d$ are learned parameters. The resulting embeddings are stacked into $\mathbf{E}^{ts} \in \mathbb{R}^{T_{ts} \times d}$ and serve as the input sequence to the Transformer, with positional encodings added in the usual manner.[15]

[15] To improve temporal awareness, additional features, such as day of the week, month, or other seasonal indicators, can be incorporated into the embeddings, either through addition or concatenation (Lim et al., 2021). In settings with long sequences, "patching" techniques, inspired by computer vision, can also be employed. Here, consecutive time steps are grouped into segments (patches), reducing the sequence length seen by the model while preserving intra-patch dependencies. We leave exploring the adequacy of these model designs for macroeconomic modeling for future research.

Appendix D Reinforcement Learning for Narrative Attribution

D.1 Connection to optimal control

Formally, the out-of-sample narrative attribution problem can be cast as a finite-horizon Markov decision process with state s_t, action $a_t \in \mathcal{A}_t$, and reward $r_t(s_t, a)$ given by (12). Full RL with optimal control would optimize long-horizon returns, but with $N_T \gg 2000$ this is infeasible. We therefore adopt an approximation with short rollouts of horizon H, inspired by classical work in stochastic control (Bertsekas and Tsitsiklis, 1996; Bertsekas, 2011, 2019). A basic (hard) rollout estimator for the state–action value is:

$$\widehat{Q}_H(s_T, a) \;:=\; \mathbb{E}\left[\sum_{h=0}^{H-1} \gamma^h \, r_{T+h}(s_{T+h}, a_{T+h}) \;\middle|\; a_T = a\right], \tag{23}$$

with discount factor γ.

Standard optimal control relies on the *hard* Bellman recursion:

$$Q^\star(s, a) \;=\; r(s, a) + \gamma \, \mathbb{E}_{s'}\left[\max_{a'} Q^\star(s', a')\right].$$

However, in high-dimensional settings the max operator easily induce brittle policies and poor exploration. Entropy-regularized or "soft" control instead augments reward with an entropy or KL-to-prior bonus (Todorov, 2007; Kappen, 2005; Ziebart, 2010; Haarnoja et al., 2018), yielding:

$$Q^\star(s, a) \;=\; r(s, a) + \gamma \, \mathbb{E}_{s'}[V^\star(s')], \tag{24}$$

$$V^\star(s) \;=\; \tau \, \log \sum_{a'} \exp\left\{\tfrac{1}{\tau} \, Q^\star(s, a')\right\}, \tag{25}$$

with temperature $\tau > 0$. The induced soft-optimal policy is:

$$\pi^\star(a \mid s) \;=\; \frac{\exp\left\{\frac{1}{\tau} Q^\star(s, a)\right\}}{\sum_{a'} \exp\left\{\frac{1}{\tau} Q^\star(s, a')\right\}}, \tag{26}$$

which recovers the hard Bellman policy as $\tau \downarrow 0$.

To align rollout targets with (26), we map truncated returns to probabilities via a Boltzmann transform:

$$q_T(a) \;\propto\; \exp\left\{\tfrac{1}{\tau_r} \, \widehat{Q}_H(s_T, a)\right\}. \tag{27}$$

Equation (27) is the finite-horizon, rollout-based analogue of (26), replacing the exact $Q^\star$ with the H-step estimator (23).

Training then projects the policy onto these rollout targets using the KL-regularized objective (15), in line with trust-region methods (Peters et al., 2010; Schulman et al.,

Table D.1. Hyperparameters and baseline values used in training.

Hyperparameter	Baseline value	Hyperparameter	Baseline value
Learning rate (η)	5×10^{-3}	Discount factor (γ)	0.97
Softmax inv. temp. (β)	1.6	Rollout horizon (H)	3
Target temp. (τ_r)	0.8	Entropy weight (λ_H)	0.01
FUND weight (λ_F)	0.8	KL weight (λ_{KL})	0.1
NPRIOR weight (λ_N)	0.1	Steps of training	250
PER weight (λ_P)	0.1	Refresh interval	5 iterations
Continuation strategy	deterministic (max)	Eval. metric	expected reward

2015; Abdolmaleki et al., 2018). With q_{t_s} of the form (27), the optimizer of (15) is driven toward the soft-optimal policy (26), with π_0 acting as a stabilizing prior and λ_{H} controlling entropy regularization.

Since $\widehat{Q}_H$ is truncated, noisy, and expensive to recompute for every (s, a) and period T, we use a parameterized softmax policy $\pi_\theta(a \mid s) = \mathrm{softmax}_a\{\beta\, \phi(s,a)^\top \theta\}$ as a compact approximation class. This projects noisy rollout targets onto a smooth policy, generalizes across states via features, and amortizes rollout costs so that deployment does not require recomputing $\widehat{Q}_H$. If the class is sufficiently rich, the projection recovers (26) exactly.

D.2 Application details and practical considerations

The training algorithm is outlined in Algorithm 1. We adopt a two-timescale training scheme common in RL: policy parameters are updated every iteration, while the environment is refreshed and new state caches are generated only every few iterations. This setup, often referred to as using multiple epochs per batch or semi-on-policy training, reduces computational cost while still providing stable learning. Similar strategies are standard in policy gradient and fitted policy iteration methods.

Table D.1 lists the parameters used for estimation. The temperature settings are tuned to avoid large jumps in the learning curves while the weights in the objective function (15) allow for a small degree of smoothing and conservatism. The weights in the reward function (12) reflect user specific preferences, and do in our benchmark application reflect a high weight on fundamental news and relatively little weight on the narrative prior and persistence.

We implement policy learning on slates of candidate articles rather than on the full set of available texts. This ease the computational burden: each period typically contains thousands of articles, making it intractable to evaluate every candidate in each policy update. By restricting attention to a smaller, representative slate - obtained through

Algorithm 1 Reinforcement Learning for Narrative Attribution

Require: Dataset $\{(\mathcal{A}_{t_s}, \mathbf{X}_{t_s}^{ts})\}_{t_s=1}^{T_s}$ and for $t_s = 1$ initial states of $\left(\mathbf{s}_{1:T-1},\ \mathbf{x}_{1:T_f-1}^{f}\right)$

Initialize policy parameters (θ).
for iter $= 1, 2, \ldots$ **do**
  if iter **mod** $5 = 0$ **then**
    Build a deterministic prefix trajectory $\{s_{t_s}\}_{t_s=1}^{T_s}$, with $s_{t_s} = \left(\mathbf{s}_T,\ \mathbf{x}_{T_{ts}}^{ts},\ \mathbf{x}_{T_f}^{f}\right)$, by rolling forward with $\arg\max_a \pi_\theta(a \mid s_{t_s})$.
    for $t_s = 1$ **to** T_s **do**
      Compute approximate action values by H-step rollouts:

$$\hat{Q}_{t_s}(a) \leftarrow \textsc{Rollout}(s_{t_s}, a, \gamma, H;\ \pi_\theta).$$

      Form target distribution:

$$q_{t_s}(a) \propto \exp\{\hat{Q}_{t_s}(a)/\tau_r\}.$$

    end for
  end if
  Update θ by one step of (stochastic) gradient descent on:

$$\mathcal{L}(\theta) = \sum_{t_s=1}^{T_s} \mathrm{KL}(q_{t_s} \parallel \pi_\theta(\cdot \mid s_{t_s})) + \lambda_{\mathrm{KL}}\, \mathrm{KL}(\pi_\theta(\cdot \mid s_{t_s}) \parallel \pi_0(\cdot \mid s_{t_s})) + \lambda_{\mathrm{H}}\, \mathcal{H}(\pi_\theta(\cdot \mid s_{t_s}))\,.$$

  Convergence check: stop when $\mathcal{L}$ or validation score stabilizes.
end for
return trained policy $\pi_\theta(a \mid s_{t_s}) \propto \exp\{\beta(\phi(s_{t_s}, a)^\top \theta)\}$.

cheap pre-ranking heuristics using the *Narrater-greedy* approach - we can train efficiently while still exposing the policy to diverse alternatives. Here we consider 100 articles in each time period. 10% of these are among the best candidates according to the pre-ranking heuristics and the rest are randomly selected from the full set of candidates. Learning convergence is monitored by evaluating the policy function using expected rewards on fixed slates - keeping the best performing parameter set for later application.

In the reward function accuracy is defined as:

$$\mathrm{RMSE}\Big(G\hat{D}P_{1:T_f}^{(i)}, GDP_{1:T_f}\Big) = \sqrt{\frac{1}{T_f}\sum_{t_f=1}^{T_f}(G\hat{D}P_{1:T_f}^{(i)} - GDP_{1:T_f})^2},$$

where $G\hat{D}P_{1:T_f}^{(i)}$ is the implied growth prediction when using article $a_{T,i}$ as input to the

filter in period T. The narrative prior is measured using the ROUGE-1 score (Lin, 2004):

$$\text{NPRIOR}(a, \tilde{a}_T) = \frac{2 \cdot \sum_{w \in V} \min\left(\text{Count}_a(w), \text{Count}_{\bar{a}}(w)\right)}{|a| + |\bar{a}|},$$

where $|a|$ and $|\bar{a}|$ denote the total number of unigrams in the candidate and reference article, respectively. We further define the degree to which the candidate article is fundamental as:

$$\text{FUND}(a) = \sigma((p_{fund}(a) - p_{noise}(a))/0.5),$$

where $p_{fund} = \sum_{c \in (demand, supply)} p(c|a)$ and $\sigma()$ is the sigmoid function. Thus, $\text{FUND}(a)$ favors articles with higher probability scores for the demand and supply classes. Finally, we use cosine similarity to measure:

$$\text{PER}(a, \mathcal{H}_{T-1}) = \frac{1}{J} \sum_{j=1}^{J} \cos\left(\bar{e}(a), \bar{h}_j\right)$$

where $\bar{e}(a)$ is the sequence embedding from (1) in period T for decision point t_s, and $\bar{h}_j$ is the associated embedding for the $t_s - j$ periods earlier with $J = 4$. To further avoid getting stuck in repeating noise, we gate the $\text{PER}(a, \mathcal{H}_{T-1})$ measure by interacting it with $\text{FUND}(a)$ before computing the aggregate reward.

Appendix E LLM prompt

```
I would like your help in forecasting the quarterly business cycle (F) for Norway alongside a more narrative account of this
↪ prediction. I will provide you with historical data consisting of 38 quarterly values of structural supply,
↪ demand, and noise series in a CSV file titled timeSeriesF_{number}.csv. These values are ordered chronologically from
↪ earliest to latest, with the first and latest values representing time index 2 and 39, respectively. For each quarter, the
↪ sum of the supply and demand series correspond to the business cycle indicator. Your task is to predict the business cycle
↪ indicator (F) for the next quarter (the 40th time index).

Additionally, I will supply you with data on 8 macroeconomic variables, consisting of 39 quarterly values, in a separate CSV
↪ file, timeSeriesX_{number}.csv. These variables collectively form the basis for computing the business cycle indicator (F)
↪ for time index 1 to 39.

You will also be provided with 39 recent news stories as text data, formatted as a list of strings in a CSV file,
↪ text_{num}.csv. Each news story is associated with one quarter, ranging from time index 2 to 40, and can assist you in
↪ inferring the historical correlation patterns as well as making your business cycle prediction for time index 40. The
↪ class of each text is either **supply**, **demand**, or **noise**, and there is statistical mapping between the
↪ structural three-part decomposition of the values in timeSeriesF_{number}.csv and the classes and sentiment of the texts.
↪ To help you understand this mapping, the true class and sentiment for each of the texts for time index 2 to 39 is provided
↪ in the file classSentiment_{number}.csv. These values are ordered chronologically from earliest to latest, with the 39th
↪ value representing the most recent quarter for which we have data. The classes are identified using numbers, with
↪ demand=1, supply=2, and noise=3.

Alongside predicting the business cycle indicator (F), please also provide:
1. An economic sentiment prediction (S) for the 40th quarter, expressed as a value between -1 and 1, reflecting the overall
↪ economic sentiment which in this context is highly correlated with the business cycle indicator.
2. A classification (C) of the primary driving force behind the economy for the 40th quarter, categorized as one of the
↪ following:
* **Supply**: Economy driven primarily by supply-side factors, defined here as residing in the semantic territory of
↪ "technology, innovation, productivity".
* **Demand**: Economy driven primarily by demand-side factors, defined here as residing in the semantic territory of "demand,
↪ preferences, sentiment".
* **Noise**: No specific supply or demand shocks dominate the economic conditions this quarter.

Your predictions should be informed by the relationship implicit in the historical data among macroeconomic variables, news
↪ stories, and the business cycle indicator.
```

Please provide your predictions structured clearly in the following JSON format:

```json
{
        "F": <predicted business cycle indicator value>,
        "S": <predicted economic sentiment value between -1 and 1>,
        "C": "Supply" | "Demand" | "Noise"
}
```