Nonlinear Factor Dynamics Forecasting: A Two-Stage Autoencoder–LSTM Framework

January 22, 2026

Abstract

We study forecasting in high-dimensional time series with nonlinear factor dynamics. When the evolution of latent factors is nonlinear, traditional direct forecasting methods suffer from structural misspecification, leading to systematic errors. Building on recent advances in deep learning, we propose **AE–LSTM**, a two-stage framework that first extracts latent factors with an autoencoder (AE) and then models their dynamics using a long short-term memory (LSTM) network. The autoencoder consistently recovers nonlinear structures, while the LSTM flexibly approximates dynamic transitions, allowing the system to capture both cross-sectional and temporal nonlinearities. Simulation studies and an application to FRED-MD real-time vintages show that AE–LSTM achieves systematic gains in predictive accuracy over standard linear and regularized benchmarks, underscoring the value of separating representation learning from dynamic forecasting.

Keywords: Factor models; Deep Learning; Nonlinear; Macroeconomic time series
JEL classification: C38; C53; E37

1 Introduction

Factor models offer a parsimonious framework for analyzing high-dimensional data by capturing co-movements across large panels through a small number of latent factors. In macroeconomics, they provide a powerful tool to distill the common sources of variation underlying numerous indicators, thereby facilitating the study of aggregate dynamics while addressing the curse of dimensionality. By decomposing observed series into systematic and idiosyncratic components, factor models enhance forecasting performance and yield structural insights into the drivers of economic fluctuations. Their ability to aggregate dispersed information across

sectors and regions makes them indispensable for business cycle analysis, real-time economic monitoring, and policy evaluation.

Building on this foundation, dynamic factor models extend the framework by explicitly incorporating temporal dynamics into the latent structure. This allows researchers to capture both cross-sectional co-movements and serial dependence, making DFMs a central tool for forecasting and real-time macroeconomic analysis. The seminal contributions of Forni, Hallin, Lippi, and Reichlin (2000, 2005) establish the generalized DFM, which recovers common components from large panels using frequency-domain methods and develops one-sided procedures for forecasting. Subsequent work focuses on determining the number of factors (Hallin and Liška, 2007; Kapetanios, 2010), improving real-time applications (Giannone, Reichlin, and Small, 2008; Barnett, Chauvet, and Leiva-Leon, 2016), and refining estimation through state-space approaches and Kalman filtering (Doz, Giannone, and Reichlin, 2011). Stock and Watson (2014) further demonstrate the utility of DFMs in identifying turning points in economic activity.

In empirical macroeconomic forecasting, DFMs are commonly estimated with a two-step procedure, reflecting the challenges of real-time data such as asynchronous releases, missing observations, and structural instability. Giannone, Reichlin, and Small (2008) highlight how two-step DFMs enable nowcasting from large and unbalanced panels; Doz, Giannone, and Reichlin (2011) formalize the approach with principal components and Kalman filtering; and Barnett, Chauvet, and Leiva-Leon (2016) show its robustness under structural breaks. This practical motivation for two-step estimation parallels our framework, where factor recovery and forecasting are separated to address misspecification in dynamic environments.

Linear factor models, while widely used, can be too restrictive in capturing macroeconomic dynamics that exhibit nonlinearities, threshold effects, or regime changes. This has motivated a line of research on nonlinear factor estimation and forecasting, including early evidence that neural networks can enhance real-time predictions (Swanson and White, 1997), statistical formulations of nonlinear factor analysis (Yalcin and Amemiya, 2001), targeted predictors for nonlinear forecasting (Bai and Ng, 2008), and recent advances using deep autoencoders (Shen and Xiu, 2024). Existing nonlinear approaches are mostly formulated as *one-step* methods, mapping high-dimensional predictors X_t directly to the target y_{t+1}. While effective in many settings, such approaches may face limitations when the latent factors themselves evolve nonlinearly, since potential misspecification can accumulate over horizons and is difficult to eliminate in multi-step forecasting.

Deep learning methods have witnessed substantial development since the early theoretical foundations on learning unknown mappings with multilayer feedforward networks (Gallant and White, 1992) and the introduction of long short-term memory (LSTM) networks for

handling sequential dependence (Hochreiter and Schmidhuber, 1997). Building on these advances, we propose a two-stage AE–LSTM framework tailored for macroeconomic forecasting. In the first stage, an autoencoder (AE) is employed to extract latent factors in a nonlinear and target-relevant manner, following the approach of Shen and Xiu (2024). In the second stage, a long short-term memory network models the temporal dynamics of the recovered factors, allowing the framework to capture both cross-sectional nonlinearities and nonlinear serial dependence. This separation of factor extraction and dynamic modeling not only mitigates potential misspecification inherent in one-step approaches, but also leverages the strength of recurrent architectures in forecasting. Applications in related domains, such as financial market prediction (Fischer and Krauss, 2018), further demonstrate the effectiveness of LSTMs in capturing complex temporal patterns, underscoring the relevance of our proposed two-step architecture.

Contributions: Summarize methodological and empirical contributions of the framework.

Structure of the Paper: Outline the subsequent sections.

2 AE–LSTM Framework

2.1 Nonlinear Dynamic Factor Model Setup

We consider a high-dimensional time series $\{X_t\}_{t=1}^T$, where each $X_t \in \mathbb{R}^N$ is driven by a low-dimensional vector of latent common factors $F_t \in \mathbb{R}^K$ with $K \ll N$. Relative to the linear approximate factor framework of Stock and Watson (2002b), we generalize both the cross-sectional mapping from factors to observables and the temporal law of motion of the factors to be governed by unknown nonlinear functions (Yalcin and Amemiya, 2001). This setting accommodates threshold effects, state dependence, and smooth transition behaviors that are often observed in macroeconomic aggregates.

Factor dynamics. The latent factors evolve according to a nonlinear vector autoregression:

$$F_{t+1} = G(F_t, F_{t-1}, \ldots, F_{t-p}) + u_{t+1}, \qquad t = p+1, \ldots, T-1, \tag{1}$$

where $G : \mathbb{R}^{K(p+1)} \to \mathbb{R}^K$ is an unknown Borel-measurable transition function. The innovation process $\{u_{t+1}\}$ satisfies

$$\mathbb{E}[u_{t+1} \mid \mathcal{F}_t] = 0,$$

where $\mathcal{F}_t = \sigma(F_s, X_s, y_s : s \leq t)$ denotes the information set at time t. This specification permits regime-dependent persistence, nonlinear propagation of shocks, and asymmetric responses across business cycle phases.

Measurement system. Each observable X_{it} is represented as the sum of a nonlinear common component and an idiosyncratic disturbance:

$$X_{it} = \varphi_i^\star(F_t) + U_{it}, \qquad i = 1, \ldots, N, \quad t = 1, \ldots, T, \tag{2}$$

or, in stacked vector form,

$$X_t = \varphi^\star(F_t) + U_t, \qquad \varphi^\star(\cdot) := \big(\varphi_1^\star(\cdot), \ldots, \varphi_N^\star(\cdot)\big)^\top.$$

Here $\{\varphi_i^\star\}$ are unknown nonlinear loading functions mapping the latent factors F_t into observables, while U_t is an idiosyncratic component with mean zero, bounded variance, and weak cross-sectional dependence.

This specification generalizes the linear factor structure by allowing flexible nonlinearities in the mapping from F_t to X_t. Such nonlinearities can capture a wide range of economically relevant patterns, including saturation effects, threshold-type responses, and asymmetric sensitivities across regimes. For instance, measures of real activity may display heightened responsiveness to financial stress during recessions, whereas labor market indicators may react more strongly to negative demand shocks than to positive ones of comparable magnitude.

Forecasting relation. The target variable depends on the next-period latent state via a nonlinear outcome mapping:

$$y_{t+1} = \psi^*(F_{t+1}) + \varepsilon_{t+1}, \qquad \mathbb{E}[\varepsilon_{t+1} \mid \mathcal{F}_t] = 0, \tag{3}$$

where $\psi^*(\cdot)$ is an unknown conditional expectation function and ε_{t+1} is the unpredictable forecast error. This formulation admits state-dependent predictability and nonlinear policy response functions, reflecting features such as asymmetric monetary interventions or nonlinear risk premia in financial markets.

To ensure statistical tractability, we assume that the transition map G, the loading functions $\{\phi_i^*\}$, and the forecasting function ψ^* belong to smooth function classes (e.g., Hölder). These restrictions guarantee that the functions can be consistently approximated by flexible nonlinear estimators. The innovations u_{t+1}, idiosyncratic shocks U_t, and forecast errors ε_{t+1} are treated as mean-zero disturbances orthogonal to the relevant information sets.

The specification nests the linear approximate factor model as a special case, while allowing for nonlinear factor loadings, nonlinear factor dynamics, and state-dependent forecasting relations.

2.2 Stage 1: Autoencoder for Common Component Recovery

We model the observed data $\{X_{it}\}$, for $i = 1, \ldots, N$ and $t = 1, \ldots, T$, following a nonlinear factor structure originally proposed by Yalcin and Amemiya (2001):

$$X_{it} = X_{it}^{\star} + U_{it} = \varphi_i^{\star}(F_t^{\star}) + U_{it}, \tag{4}$$

where the common component $X_{it}^{\star}$ is governed by a potentially nonlinear factor loading function $\varphi_i^{\star}(\cdot)$ of a K-dimensional latent factor vector $F_t^{\star}$. The idiosyncratic error term is denoted by U_{it}. We use vector notation $X_t, X_t^{\star}, \varphi^{\star}(F_t^{\star})$, and U_t to denote the $N \times 1$ stacked values of these variables at time t. Similarly, $X, X^{\star}, U$ represent their $N \times T$ matrix versions, and $F^{\star}$ denotes the $K \times T$ factor matrix.

Assumption 1 (Boundedness). There exists a constant $B > 0$ such that $F_t^{\star}$ is supported on a compact set $\mathcal{B} \subset [-B, B]^K$ almost surely for all t. Conditional on $F^{\star}$, we have

$$\mathrm{vec}(U) \overset{d}{=} \Sigma^{1/2}\mathrm{vec}(Z),$$

where $Z \in \mathbb{R}^{N \times T}$ contains independent, zero-mean sub-Gaussian entries with norm bounded by σ_z^2, and Σ is positive semi-definite with bounded spectral norm.

Assumption 2 (Smoothness). There exists $\beta \in \mathbb{N}_+$ such that $\varphi^{\star}(\cdot) \in (H^{\beta}(\mathcal{B}, B))^N$.

This setup ensures that the loading functions $\varphi_i^{\star}(\cdot)$ are sufficiently smooth to be approximated by deep neural networks, while the latent factors are bounded and the idiosyncratic errors are weakly dependent. These conditions parallel those in the linear factor literature but are essential for developing nonparametric theoretical guarantees for autoencoders.

2.2.1 Autoencoder Architecture

Given the nonlinear factor model in (4), our objective is to use an autoencoder (AE) to approximate the unknown loading functions $\varphi_i^{\star}(\cdot)$ and to recover the latent factors $F_t^{\star}$ up to an invertible transformation. We now provide a step-by-step description of the AE architecture, aligned with the notation of the DGP.

Neural network building blocks. Let $\sigma(x) = \max\{x, 0\}$ be the ReLU activation and $\sigma_v : \mathbb{R}^r \to \mathbb{R}^r$ its shifted version with shift vector $v = (v_1, \ldots, v_r)$. A fully-connected d-layer network with width vector $n = (n_0, \ldots, n_{d+1})$, n_0 input and n_{d+1} output, computes

$$f(x) = W_d \sigma_{v_d} W_{d-1} \sigma_{v_{d-1}} \cdots W_1 \sigma_{v_1} W_0 x, \tag{5}$$

with weight matrices W_ℓ and shifts v_ℓ. We denote by $\mathcal{F}_{n_0}^{n_{d+1}}(d, w)$ the class of such networks with depth d and maximum width w.

Step 1: Encoder. For each observation $X_t = (X_{1t}, \ldots, X_{Nt})^\top \in \mathbb{R}^N$, the encoder is a neural network

$$\rho : \mathbb{R}^N \to \mathbb{R}^{K_1}, \qquad \widehat{F}_t = \rho(X_t), \tag{6}$$

which compresses X_t into a bottleneck representation $\widehat{F}_t$. The dimension K_1 is a tuning parameter, serving as the estimated number of latent factors.

Step 2: Decoder. Each coordinate X_{it} is reconstructed from $\widehat{F}_t$ through a decoder $\phi_i(\cdot)$:

$$\phi_i : \mathbb{R}^{K_1} \to \mathbb{R}, \qquad \widehat{X}_{it} = \phi_i(\widehat{F}_t), \quad i = 1, \ldots, N. \tag{7}$$

Stacking all outputs gives

$$\widehat{X}_t = \big(\phi_1(\rho(X_t)), \ldots, \phi_N(\rho(X_t))\big)^\top \in \mathbb{R}^N.$$

This "disjoint-output" design uses N separate 1-output subnetworks rather than a single N-output decoder, reducing parameter complexity while preserving approximation power.

Step 3: AE function class. We define the function class of disjoint-output AEs as

$$\mathcal{F}_{\mathrm{AE}}^{K_1} := \Big\{ (\phi_1, \ldots, \phi_N) \circ \rho : \ \rho : \mathbb{R}^N \to \mathbb{R}^{K_1}, \ \phi_i : \mathbb{R}^{K_1} \to \mathbb{R}, \ i = 1, \ldots, N \Big\}. \tag{8}$$

Step 4: Training objective. The encoder and decoders are trained jointly by minimizing the reconstruction error:

$$(\widehat{\rho}, \widehat{\phi}_1, \ldots, \widehat{\phi}_N) =_{(\rho, \phi_1, \ldots, \phi_N) \in \mathcal{F}_{\mathrm{AE}}^{K_1}} \sum_{t=1}^{T} \sum_{i=1}^{N} \big(X_{it} - \phi_i(\rho(X_t))\big)^2. \tag{9}$$

This objective targets the common component $X_{it}^\star = \varphi_i^\star(F_t^\star)$ in (4), replacing the unknown $(F_t^\star, \varphi_i^\star)$ by $(\rho(X_t), \phi_i)$.

Step 5: Outputs. After estimation, the AE yields

$$\widehat{F}_t = \widehat{\rho}(X_t) \quad \text{(estimated factors)}, \qquad \widehat{X}_{it} = \widehat{\phi}_i(\widehat{F}_t) \quad \text{(reconstructed common component)}.$$

Thus, $\widehat{F}_t$ approximates $F_t^\star$ up to an invertible transformation, while $\widehat{\phi}_i(\cdot)$ approximates $\varphi_i^\star(\cdot)$.

2.2.2 Theoretical Results

Based on the autoencoder architecture described above and following the analysis of Shen and Xiu (2024), we can establish theoretical guarantees for how well the AE recovers the nonlinear factor structure in (4). Two key conclusions are summarized below.

Proposition 1 Suppose Assumptions 1–2 hold. Then, with high probability, the mean squared reconstruction error satisfies

$$\frac{1}{NT}\sum_{t=1}^{T}\sum_{i=1}^{N}\left(\widehat{X}_{it} - X_{it}^\star\right)^2 \;\lesssim\; T^{-\frac{2\beta}{2\beta+K}} \;+\; N^{-1}K_1 \;+\; \epsilon_N,$$

up to logarithmic factors, where ϵ_N is the approximation error of the encoder.

This result shows that the AE achieves the optimal nonparametric rate $T^{-2\beta/(2\beta+K)}$, as if the true factors $F_t^\star$ were observed. The additional terms reflect the cost of factor estimation and encoder approximation.

Proposition 2 Under the same conditions as Theorem 2.2.2, there exists a function $\mu : \mathbb{R}^{K_1} \to \mathbb{R}^K$ such that, with high probability,

$$\frac{1}{T}\sum_{t=1}^{T}\|\mu(\widehat{F}_t) - F_t^\star\|^2 \;\lesssim\; \epsilon_N K_1 \;+\; N\epsilon_N T^{-\frac{2\beta}{2\beta+K}} \;+\; N\epsilon_N^2,$$

up to logarithmic factors.

This result establishes that autoencoders consistently recover the latent factors, up to an invertible nonlinear transformation. Unlike linear factor models where the number of factors K is point-identified, in nonlinear settings the bottleneck dimension K_1 is treated as a hyperparameter.

What the AE delivers. The estimated encoder $\widehat{\rho}$ maps each observation X_t into a *low-dimensional representation*

$$\widehat{F}_t = \widehat{\rho}(X_t),$$

which serves as a proxy for the latent factors $F_t^\star$, identifiable up to a smooth invertible transformation. The associated decoder $\widehat{\phi} = (\widehat{\phi}_1, \ldots, \widehat{\phi}_N)$ reconstructs the *common component* as

$$\widehat{X}_t = \big(\widehat{\phi}_1(\widehat{F}_t), \ldots, \widehat{\phi}_N(\widehat{F}_t)\big)^\top,$$

thereby separating systematic variation from idiosyncratic noise.

Under the boundedness, smoothness, and pervasiveness conditions introduced earlier, and with appropriately growing network depth and width, the average reconstruction error of $\widehat{X}_t$ converges to zero as (N, T) increase. Moreover, the representations $\widehat{F}_t$ consistently recover the latent factors $F_t^\star$ up to a reparameterization.

From a practical perspective, these estimated factors $\widehat{F}_t$ constitute the inputs to the dynamic module developed in the next section, which models the transition $t \mapsto t+1$ and is ultimately used to forecast the target variable Y_{t+1}.

2.3 Mathematical Analysis of Direct Autoencoder Limitations

The autoencoder architecture described above provides a principled way to recover low-dimensional latent representations and reconstruct the common component of observables. A natural question is whether one can go further and directly forecast the target variable y_{t+1} from the panel X_t, by embedding the prediction step inside the autoencoder. This leads to the notion of a *direct autoencoder*, in which the encoder simultaneously produces factor proxies and prediction features, bypassing explicit modeling of factor dynamics.

TBD

This motivates separating factor recovery from dynamic modeling, leading to our proposed AE–LSTM framework.

To address the misspecification of direct approaches, we adopt a **two-step procedure**:

(i) **Factor extraction.** Apply an unsupervised autoencoder to recover latent representations,

$$\widehat{F}_t = \rho(X_t),$$

which serve as proxies for the true factors $F_t^\star$.

(ii) **Dynamic modeling.** Capture the temporal evolution $\widehat{F}_t \mapsto \widehat{F}_{t+1}$ using a recurrent neural network, specifically an LSTM, which has universal approximation properties for nonlinear state transitions.

(iii) **Forecasting the target.** Map the predicted factors into the target space,

$$\widehat{y}_{t+1} = g(\widehat{F}_{t+1}),$$

where $g(\cdot)$ is a flexible decoder trained to approximate $g^{\star}(\cdot)$.

This separation ensures consistent estimation of common factors while allowing a flexible treatment of nonlinear dynamics, thereby avoiding the structural misspecification inherent in one-step direct autoencoders.

2.4 Stage 2: LSTM for Temporal Dynamics

LSTM as a natural choice. We employ long short-term memory (LSTM) networks to model the nonlinear and persistent evolution of latent factors. Relative to linear autoregressions or vanilla recurrent networks, LSTMs are particularly well suited for economic time series:

(a) *Nonlinear transformations.* Gated, multiplicative interactions implement state-dependent mappings, enabling LSTMs to approximate complex transition laws featuring thresholds, asymmetries, and other nonlinearities (Hochreiter and Schmidhuber, 1997; Landi et al., 2021).

(b) *Long-term memory.* The cell state and forget gate provide an explicit memory mechanism that captures long-range dependence, crucial for business-cycle persistence, delayed policy transmission, and regime changes (Hochreiter and Schmidhuber, 1997; Gers et al., 2000).

(c) *Training stability.* Gating mitigates vanishing and exploding gradients that plague vanilla RNNs; in practice, gradient clipping and related techniques further stabilize optimization over long sequences (Bengio et al., 1994; Pascanu et al., 2013).

(d) *Expressive power.* Recurrent architectures are universal approximators of state-transition and sequence mappings; LSTMs inherit this expressivity while remaining trainable in practice (Hammer, 2000; Schäfer and Zimmermann, 2006).

From an applied economics perspective, these properties align with evidence that LSTMs deliver forecasting gains in macro-financial environments, including mixed-frequency settings (Fischer and Krauss, 2018; Kamolthip, 2021).

LSTM dynamics model. Let $\{\widehat{F}_t\}_{t=1}^T$, $\widehat{F}_t \in \mathbb{R}^K$, denote the latent factors estimated in Step 1 by the autoencoder. We aim to approximate the unknown transition mapping G in

$$F^\star_{t+1} \;=\; G(F^\star_t, F^\star_{t-1}, \ldots) \;+\; \eta_{t+1}, \tag{10}$$

using a recurrent neural network that takes recent factor histories as input and outputs the next-period factor. For notational convenience, define the observed input sequence as

$$z_t := \widehat{F}_t \in \mathbb{R}^K, \qquad \mathcal{Z}_t := (z_{t-L+1}, \ldots, z_t),$$

for a chosen lag length $L \geq 1$. The LSTM then parameterizes a mapping

$$\widehat{F}_{t+1} = \mathcal{G}_\theta(\mathcal{Z}_t),$$

where $\mathcal{G}_\theta$ denotes the nonlinear state-transition function implemented by the LSTM with trainable parameters θ. By construction, $\mathcal{G}_\theta$ serves as a flexible approximation of G, with the cell and hidden states of the LSTM capturing both short- and long-term dependencies in $\mathcal{Z}_t$.

LSTM recurrence Given an input sequence $\{z_t\}$ with $z_t \in \mathbb{R}^D$, a one-layer (unidirectional) LSTM with hidden size H maintains two internal states: a hidden state $h_t \in \mathbb{R}^H$ that serves as the output at time t, and a cell state $c_t \in \mathbb{R}^H$ that acts as the long-term memory. The update equations are governed by three gates and a candidate cell state:

$$\begin{aligned}
\text{input gate:} \quad & i_t = \sigma(W_i z_t + U_i h_{t-1} + b_i)\,, \\
\text{forget gate:} \quad & f_t = \sigma(W_f z_t + U_f h_{t-1} + b_f)\,, \\
\text{output gate:} \quad & o_t = \sigma(W_o z_t + U_o h_{t-1} + b_o)\,, \\
\text{candidate (cell proposal):} \quad & \tilde{c}_t = \tanh(W_c z_t + U_c h_{t-1} + b_c)\,, \\
\text{cell update:} \quad & c_t = f_t \odot c_{t-1} \;+\; i_t \odot \tilde{c}_t, \\
\text{hidden update:} \quad & h_t = o_t \odot \tanh(c_t),
\end{aligned} \tag{11}$$

where $\sigma(\cdot)$ is the logistic sigmoid, $\tanh(\cdot)$ is the hyperbolic tangent, and $\odot$ denotes the Hadamard (elementwise) product. The parameter dimensions are

$$W_\bullet \in \mathbb{R}^{H\times D}, \quad U_\bullet \in \mathbb{R}^{H\times H}, \quad b_\bullet \in \mathbb{R}^H, \qquad \bullet \in \{i, f, o, c\}.$$

Interpretation. Each component plays a distinct role:

- The *input gate* i_t controls how much of the new candidate information $\tilde{c}_t$ flows into the memory.
- The *forget gate* f_t determines how much of the past cell state c_{t-1} is retained versus discarded.
- The *output gate* o_t regulates which parts of the updated memory c_t are exposed as the hidden state h_t.
- The *candidate* $\tilde{c}_t$ provides a nonlinear proposal for new memory content, updated only if admitted by i_t.

Together, these gates allow the LSTM to balance short-term adaptation with long-term memory, a property particularly useful for economic and financial sequences exhibiting persistence and regime shifts.

Compact form. For computational efficiency, the gate and candidate updates can be written in a single affine transformation. Let

$$\xi_t = [\, z_t^\top,\ h_{t-1}^\top \,]^\top \in \mathbb{R}^{D+H}.$$

Then

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ \tilde{c}_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} (W\,\xi_t + b)\,, \qquad W \in \mathbb{R}^{4H\times(D+H)},\ b \in \mathbb{R}^{4H}, \tag{12}$$

followed by the state updates in (11). This compact representation consolidates the four affine maps, improving training efficiency and simplifying the application of regularization techniques such as dropout or weight decay.

Practical considerations. Initial states are typically set to $h_0 = c_0 = \mathbf{0}$ or treated as learnable vectors. For stacked (deep) LSTMs, the input to layer ℓ is given by $z_t^{(\ell)} = h_t^{(\ell-1)}$, with $z_t^{(1)} = z_t$. Bidirectional variants run two independent LSTMs forward and backward, concatenating their hidden states to form richer contextual representations. In practice, techniques such as weight decay, gradient clipping, layer normalization, and time-locked (variational) dropout are widely employed to improve training stability and generalization.

Training target and objective. Since the true latent factors $F_t^\star$ are unobserved, we use the AE-estimated factors as training targets. To avoid notational ambiguity, let $\widehat{F}_t^{\mathrm{AE}}$ denote

the *target factors* from the autoencoder, and $\widetilde{F}_t$ the *predicted factors* from the LSTM. Using teacher forcing over rolling windows, the one-step prediction loss is

$$\mathcal{L}_{\text{dyn}}(\Theta, \theta_F) \;=\; \sum_{t=L}^{T-1} \Big\| \widehat{F}^{\text{AE}}_{t+1} \;-\; \widetilde{F}_{t+1}\big(\mathcal{Z}_t; \Theta, \theta_F\big) \Big\|_2^2 \;+\; \lambda\, \mathcal{R}(\Theta, \theta_F), \tag{13}$$

where $\mathcal{R}$ is a regularization term and $\lambda \geq 0$ its penalty weight. The hidden and cell states at the start of each rolling window, (h_{t-L}, c_{t-L}), are either initialized to zero or treated as learnable vectors; empirical results are robust to either specification.

One-step forecasting. Let $F_t^\star \in \mathbb{R}^K$ denote the (unobserved) latent factors, and suppose the target satisfies the measurement relation

$$y_{t+1} \;=\; \phi^\star(F^\star_{t+1}) \;+\; \varepsilon_{t+1}, \qquad \mathbb{E}[\varepsilon_{t+1} \mid \mathcal{F}_t] = 0, \tag{14}$$

where $\phi^\star(\cdot)$ is the target-loading function. Given estimated factors $\widehat{F}_t \in \mathbb{R}^K$ from the encoder stage, we set $z_t := \widehat{F}_t$ and feed it into the LSTM recurrence (11) to obtain the dynamic state (h_t, c_t). On top of h_t, we attach lightweight prediction heads:

$$\widehat{F}_{t+1} = \Phi_F(h_t), \qquad \widehat{y}_{t+1} = \Phi_y(\widehat{F}_{t+1}),$$

where Φ_F predicts the next-period factors and Φ_y maps predicted factors to the target. This two-head structure ensures that factor dynamics and target forecasting are jointly learned, while preserving the interpretability of $\widehat{F}_{t+1}$ as a factor proxy.

Multi-step forecasting. For horizons $h > 1$, we consider Recursive strategy.

Recursive (iterated) rollout. Starting from (h_t, c_t) and input $z_t = \widehat{F}_t$, we generate future factors and states open-loop:

$$\begin{aligned} \widehat{F}_{t+j+1} &= \Phi_F(h_{t+j}), \\ (h_{t+j+1}, c_{t+j+1}) &= \text{LSTM}\big(z_{t+j+1},\, h_{t+j},\, c_{t+j};\, \Theta\big), \qquad z_{t+j+1} := \widehat{F}_{t+j+1}, \\ \widehat{y}_{t+h} &= \Phi_y(\widehat{F}_{t+h}), \qquad j = 0, \ldots, h-1. \end{aligned} \tag{15}$$

During training, teacher forcing is applied (feeding $z_t = \widehat{F}_t$ from the encoder), whereas at inference we rely on the free-running recursion (15). Recursive forecasts are parameter-efficient but may suffer from accumulated rollout error as h increases.

Practical considerations (uncertainty-aware gating for macro). Macroeconomic time series frequently exhibit episodic spikes in uncertainty—such as periods of financial stress, elevated policy uncertainty, or abrupt volatility shocks. In these environments, robust forecasters tend to place less weight on noisy contemporaneous innovations and instead rely more heavily on the persistent component of the dynamics. To capture this behavior, we augment the standard LSTM gates with an *uncertainty-aware gate* that adaptively adjusts the influence of new inputs versus historical memory in response to shifts in uncertainty. Conceptually, this mechanism allows the model to tilt between "signal-driven" and "persistence-driven" forecasting modes depending on prevailing conditions. We will demonstrate the empirical benefits of this extension in the application section.

2.5 Recap: AE–LSTM Model Framework

Nonlinear DGP. Let $X_t \in \mathbb{R}^N$ denote a high-dimensional panel of observables, $F_t^\star \in \mathbb{R}^K$ ($K \ll N$) the latent common factors, and $y_{t+1} \in \mathbb{R}$ the forecasting target. The predictors and target are linked to the latent state through unknown nonlinear measurement mappings:

$$X_t \;=\; \phi^\star(F_t^\star) + U_t, \qquad y_{t+1} \;=\; \psi^\star(F_{t+1}^\star) + \varepsilon_{t+1}, \tag{16}$$

where U_t is idiosyncratic noise and ε_{t+1} is an unpredictable shock with $\mathbb{E}[\varepsilon_{t+1} \mid \mathcal{F}_t] = 0$. The latent factors evolve according to a (possibly nonlinear) transition law,

$$F_{t+1}^\star \;=\; G^\star(F_t^\star, F_{t-1}^\star, \dots) + \eta_{t+1}, \tag{17}$$

with innovation η_{t+1}. The forecasting objective is to predict y_{t+h} (and, if desired, $F_{t+h}^\star$) exploiting (16)–(17).

2.5.1 Model architecture and objectives

The AE–LSTM framework implements a two-stage strategy aligned with this nonlinear factor system:

(i) **Stage 1: Factor recovery (autoencoder).** Estimate latent factors from the cross section via an encoder $\mathcal{E}_\phi : \mathbb{R}^N \to \mathbb{R}^K$,

$$\widehat{F}_t \;=\; \mathcal{E}_\phi(X_t), \qquad t = 1, \dots, T, \tag{18}$$

optionally paired with a decoder $\mathcal{D}_\phi : \mathbb{R}^K \to \mathbb{R}^N$ to reconstruct observables: $\widehat{X}_t = \mathcal{D}_\phi(\widehat{F}_t)$.

A standard reconstruction objective is

$$\mathcal{L}_{\mathrm{AE}}(\phi) \;=\; \sum_{t=1}^{T} \ell_X\big(X_t, \mathcal{D}_\phi(\mathcal{E}_\phi(X_t))\big) \;+\; \lambda\,\mathcal{R}(\phi), \tag{19}$$

where ℓ_X is a reconstruction loss (squared or Huber) and $\mathcal{R}$ a regularizer (e.g., weight decay). The recovered sequence $\{\widehat{F}_t\}$ serves as inputs to the dynamic stage.

(ii) **Stage 2: Dynamic modeling (LSTM).** The sequence $\{\widehat{F}_t\}$ is passed into an LSTM to approximate the nonlinear transition $G^\star$ in (17). The LSTM produces next-period factor predictions $\widetilde{F}_{t+1}$, which are then mapped to the target via a prediction head:

$$\widehat{y}_{t+1} = \Phi_y(\widetilde{F}_{t+1}).$$

This stage captures nonlinear persistence, threshold effects, and regime shifts that are central to macroeconomic dynamics.

Summary. In line with the nonlinear factor model (4), the AE–LSTM framework proceeds as follows:

(1) Compress X_t into $\widehat{F}_t = \rho(X_t)$ via the encoder.

(2) Reconstruct each X_{it} as $\widehat{X}_{it} = \phi_i(\widehat{F}_t)$ via decoders.

(3) Estimate $(\rho, \phi_1, \ldots, \phi_N)$ by solving (19).

(4) Use the recovered factors $\widehat{F}_t$ as inputs to the LSTM to model dynamics and produce forecasts of both $\widehat{F}_{t+1}$ and $\widehat{y}_{t+1}$.

This separation of cross-sectional factor recovery and temporal dynamic modeling ensures consistent extraction of latent states while providing flexibility to capture nonlinear state evolution.

2.5.2 Special Cases and Connections

The baseline setting underscores two distinct sources of nonlinearity: (i) the measurement equations linking observed data and the target to latent factors, and (ii) the temporal evolution of the factors themselves. Because both components may be nonlinear, traditional linear factor models—such as principal components for factor extraction or VARs for dynamics—are generally misspecified. The proposed AE–LSTM framework addresses this by decoupling the two components: in the first stage, a deep autoencoder consistently recovers

latent factors even under nonlinear loadings $\phi^\star$; in the second stage, an LSTM approximates the nonlinear transition $G(\cdot)$ and captures long-run dependencies in factor dynamics. Finally, the predicted factors $\widehat{F}_{t+1}$ are mapped to the target via the outcome relation $\psi^\star(\cdot)$. This two-step structure generalizes linear DFMs, accommodates nonlinearities in both measurement and dynamics, and provides a flexible yet interpretable approach to macroeconomic forecasting. If one or both components are linear, the framework reduces to well-known econometric benchmarks.

A. Fully linear. When both the measurement relations and the factor dynamics are linear, the model collapses to the classical dynamic factor framework. In this benchmark, the panel X_t loads linearly on factors, the factors follow a finite-order VAR, and the target depends linearly on the future factor realization:

$$X_t = \Lambda F_t + U_t, \qquad F_{t+1} = \sum_{j=1}^{p} A_j F_{t-j} + \eta_{t+1}, \qquad y_{t+1} = \alpha + \beta^\top F_{t+1} + \varepsilon_{t+1}.$$

This specification corresponds to the well-established class of factor models, including the principal component estimator and the generalized dynamic factor approach. Extensions such as factor-augmented VARs also belong to this linear setting, where estimated factors are embedded into a VAR for forecasting or structural analysis.

Estimation in this environment is straightforward: principal components consistently recover the factor space under large N, T asymptotics, a finite-order VAR captures factor dynamics, and state-space formulations allow likelihood-based estimation via the Kalman filter. Multi-step forecasts are obtained by iterating the VAR forward. Because all relations are linear, nonlinear function approximators such as autoencoders or LSTMs provide no additional benefit.

B. Partly linear. When only one component of the system is nonlinear, two representative settings are especially relevant.

1. **Nonlinear measurement, linear dynamics.** In this configuration, the mappings $\phi^\star$ and $\psi^\star$ are nonlinear, while the factor dynamics G reduce to a finite-order VAR. This structure allows for rich nonlinear interactions between observables, latent factors, and the target variable, but preserves a simple linear law of motion for F_t.

   Estimation requires nonlinear factor recovery, for which an autoencoder is essential. Once factors are extracted, the linear dynamics imply that $\mathbb{E}[y_{t+1} \mid X_t]$ is a well-defined

function of the recovered factors. In this case, a supervised autoencoder (SAE) that directly maps X_t to y_{t+1} is correctly specified for one-step prediction.

2. **Linear measurement, nonlinear dynamics.** In the complementary case, the measurement system is linear (so that ϕ and ψ follow the classical factor structure), while the factor dynamics G are nonlinear. This corresponds to nonlinear state-space models, for which traditional econometric approaches rely on approximate filters such as the extended Kalman filter (EKF) or unscented Kalman filter (UKF). Here, principal component analysis (PCA) remains sufficient to recover the factor space, since the measurement system is linear. The challenge lies in approximating the nonlinear transition function $G(\cdot)$. While a generic RNN may be employed, the LSTM architecture is particularly well-suited: its gating mechanism captures long-run dependencies, mitigates vanishing gradients, and enjoys universal approximation properties for sequence-to-sequence maps. Thus, LSTMs provide a flexible and robust alternative to traditional nonlinear filters for modeling the evolution of F_t.

These partly linear cases illustrate how the AE–LSTM framework naturally encompasses well-known econometric models. When nonlinearity arises only in the measurement system, the approach reduces to AE-type methods combined with linear dynamics. When nonlinearity arises only in the factor transition, it corresponds to nonlinear state-space models, with LSTMs offering a modern implementation. In both scenarios, the proposed framework remains consistent with existing methods while extending their reach to richer nonlinear environments.

3 Monte Carlo Experiments

- DGP1: Fully Linear Benchmark.
- DGP2: Nonlinear Forecast Equation.
- DGP3: Nonlinear Observables.
- DGP4: Nonlinear Factor Dynamics.
- DGP5: Fully Nonlinear
- Evaluation metrics: RMSE and Confusion Rate.
- Competing models: AR, PCA, SPCA, TargetPCA, Ridge, Lasso, aLasso, AE–LSTM.

We construct synthetic data under a unified latent factor framework. Each data-generating process (DGP) consists of three structural components:

- **Factor dynamics:** F_t
- **Observables:** X_t
- **Forecast target:** Y_{t+1}

Let $F_t \in \mathbb{R}^K$ denote a vector of factors evolving over time, which jointly determine both the high-dimensional predictors $X_t \in \mathbb{R}^N$ and the forecast target $Y_{t+1} \in \mathbb{R}$. This framework is sufficiently general to nest a variety of forecasting environments while maintaining a consistent latent structure. Across the designs, we modify one component at a time to isolate the source of nonlinearity and assess its impact on model performance.

In most cases, the factors follow a stationary autoregressive process of order $p \in \{1, 2\}$:

$$F_t = \sum_{j=1}^{p} A_j F_{t-j} + \eta_t, \quad \eta_t \sim \mathcal{N}(0, \Sigma_\eta), \tag{20}$$

where each coefficient matrix $A_j \in \mathbb{R}^{K \times K}$ is diagonal with independently sampled entries

$$[A_j]_{kk} \sim \mathcal{N}\left(\frac{0.3}{j}, 0.05^2\right).$$

All autoregressive matrices are scaled to ensure stationarity, with spectral radius bounded below unity.

3.1 DGP 1: Fully Linear Benchmark

This baseline specification imposes linearity across all three components:

$$F_t = \sum_{j=1}^{p} A_j F_{t-j} + \eta_t, \tag{21}$$

$$X_t = \Lambda F_t + u_t, \quad u_t \sim \mathcal{N}(0, \Sigma_u), \tag{22}$$

$$Y_{t+1} = \alpha + \beta^\top F_t + \varepsilon_{t+1}, \quad \varepsilon_{t+1} \sim \mathcal{N}(0, \sigma^2). \tag{23}$$

This design provides a clean benchmark under which standard linear factor-augmented forecasting models are expected to perform well.

3.2 DGP 2: Nonlinear Forecast Equation

In this design, we retain the linear structure in both factor evolution and observables, but allow the forecast target to depend nonlinearly on the interaction between two factors:

$$Y_{t+1} = \alpha + \beta^\top F_t + \gamma F_{t,1} F_{t,2} + \varepsilon_{t+1}, \quad \gamma = 0.1. \tag{24}$$

This formulation captures second-order nonlinear effects that may arise in economic forecasting, such as complementarities between hidden drivers.

3.3 DGP 3: Nonlinear Observables

Here, we introduce nonlinearity in the mapping from factors to observables while preserving linearity elsewhere:

$$X_t = \Gamma \sin(F_t) + u_t. \tag{25}$$

This setup reflects environments in which observed macroeconomic indicators may be nonlinear transformations of underlying latent states, due to threshold effects, cyclical smoothing, or bounded measurement functions.

3.4 DGP 4: Nonlinear Factor Dynamics

Finally, we allow the factors process itself to evolve in a nonlinear manner:

$$F_t = \sum_{j=1}^{p} A_j F_{t-j} + \gamma_f \cdot \sin\left(F_{t-1} \odot F_{t-2}\right) + \eta_t. \tag{26}$$

The added term introduces element-wise nonlinear interactions between lagged factors, thereby capturing more complex latent dynamics. Such specifications are motivated by the presence of regime-switching behavior or nonlinear propagation of shocks in macroeconomic systems.

3.5 DGP 5: Fully Nonlinear

Together, these four designs provide a systematic way to examine how different sources of nonlinearity—whether in dynamics, observables, or forecast targets—affect the performance of competing forecasting models. This stepwise structure facilitates both interpretability and robustness analysis.

3.6 Comparison Models

To benchmark the performance of our proposed AE_UG model, we consider a suite of commonly used forecasting methods that span both linear and regularized specifications.

- **AR(p)**: A linear autoregressive model of order p estimated on the univariate target series y_t via ordinary least squares. The lag order is selected using BIC.
- **AR+PCA**: A factor-augmented regression model where principal components (PCs) extracted from X_t are added to the AR(p) specification. PCs are computed from the top k eigenvectors of the sample covariance matrix.
- **AR+SPCA**: A sparse principal components variant, where loadings are estimated using sparse PCA, which improves variable selection and interpretability.
- **AR+TargetPCA**: A supervised dimension reduction method where PCA is applied to a subset of X_t chosen based on their correlation with the target y_t.
- **AR+Ridge**: A ridge regression specification where y_t is regressed on lagged values and contemporaneous X_t, with ℓ_2-penalty used to shrink coefficients and control multicollinearity.
- **AR+Lasso**: A Lasso regression variant where sparsity is enforced via an ℓ_1-penalty.
- **AR+aLasso**: The adaptive Lasso model assigns data-driven weights to each predictor to improve selection consistency over standard Lasso.
- **AE_UG**: The proposed autoencoder model with uncertainty-gated LSTM dynamics.

3.7 Evaluation Metrics

Model performance is evaluated using the following metrics:

- **RMSE**: Relative Mean squared forecast error by the benchmark AR(p) model:

$$\text{RMSE} = \frac{\sum_t (\hat{Y}_{t+1} - Y_{t+1})^2}{\sum_t (\hat{Y}_{t+1}^{\text{AR}(p)} - Y_{t+1})^2}$$

- **Confusion Rate (CR)**: A directional performance measure based on the 2×2 confusion matrix of predicted vs. actual movements. Columns represent actual directions (up or down), while rows represent predicted directions. The confusion rate is defined as:

$$\text{CR} = \frac{\text{False Positives} + \text{False Negatives}}{\text{Total Observations}}$$

That is, the sum of the off-diagonal elements divided by the total number of forecasts. A lower CR indicates better directional forecasting performance. This metric follows the approach of Swanson and White (1997).

Table 1: RMSE Across Models for DGP1–DGP4 (Bold = Minimum, $*$ = AE_UG Wins)

(Factors, Lags)	AR+PCA	AR+SPCA	AR+TargetPCA	AR+Ridge	AR+Lasso	AR+aLasso	AE_UG
DGP1 (3,1)	0.9944	0.9934	0.9934	1.0895	1.0000	1.0000	**0.9633***
DGP1 (3,2)	0.9959	0.9949	0.9959	1.1461	1.0000	1.0000	**0.9298***
DGP1 (10,1)	1.0026	1.0114	1.0133	1.1027	1.0000	1.0000	**0.9237***
DGP1 (10,2)	1.0003	0.9979	1.0002	1.1221	1.0000	1.0000	**0.9418***
DGP2 (3,1)	1.0051	1.0034	1.0077	1.1125	1.0000	1.0000	**0.9761***
DGP2 (3,2)	0.9989	1.0041	1.0000	1.0586	1.0000	1.0000	**0.9411***
DGP2 (10,1)	1.0044	1.0000	1.0062	1.1131	1.0000	1.0000	**0.9681***
DGP2 (10,2)	1.0008	1.0008	1.0008	1.0754	1.0000	1.0000	**0.9135***
DGP3 (3,1)	1.0000	0.9991	1.0037	1.1037	1.0000	1.0000	**0.8945***
DGP3 (3,2)	1.0048	1.0048	1.0067	1.1062	1.0000	1.0000	**0.9306***
DGP3 (10,1)	1.0009	1.0019	1.0009	1.1412	1.0000	1.0000	**0.9448***
DGP3 (10,2)	1.0010	1.0000	1.0010	1.1733	1.0000	1.0000	**0.9642***
DGP4 (3,1)	1.0000	1.0000	1.0009	1.1012	1.0000	1.0000	**0.8974***
DGP4 (3,2)	1.0058	1.0078	1.0097	1.1288	1.0000	1.0000	**0.9613***
DGP4 (10,1)	1.0000	1.0020	1.0010	1.1196	1.0000	1.0000	**0.9446***
DGP4 (10,2)	1.0038	1.0038	1.0047	1.0941	1.0000	1.0000	**0.9413***

Notes: Each cell is RMSE relative to AR. The lowest value in each row is bolded. AE_UG entries that are the minimum receive a superscript star (*).

Table 1 reports the relative mean squared error (RMSE), by the AR benchmark, across a range of forecasting models and data-generating processes (DGPs). Each row corresponds to a different configuration of the number of factors (F) and lags (p), with DGP2–DGP4 sequentially introducing nonlinearity into the forecast target, observables, or latent factor dynamics. Across all specifications, the AE_UG model consistently achieves the lowest RMSE in each row, with every minimum highlighted in bold and marked with a superscript asterisk. This dominance is particularly pronounced under DGP3 and DGP4, where nonlinearities arise in the observable and latent components, respectively. In these environments, linear models such as AR_PCA, Lasso, or Ridge struggle to adapt to the structural complexity, while AE_UG exhibits robust performance gains—often reducing RMSE by 5–10% relative to AR.

Table 2: CR Across Models for DGP1–DGP4 (Bold = Minimum, $*$ = AE_UG Wins)

(Factors, Lags)	AR	AR+PCA	AR+SPCA	AR+TargetPCA	AR+Ridge	AR+Lasso	AR+aLasso	AE_UG
DGP1 (3,1)	0.3519	0.4390	0.4460	0.4216	0.4808	0.3519	0.3519	0.4564
DGP1 (3,2)	0.6620	0.5575	0.5854	0.5540	0.5401	0.6620	0.6620	**0.4704***
DGP1 (10,1)	0.6481	0.6132	0.5540	0.5854	0.5122	0.6481	0.6585	**0.5087***
DGP1 (10,2)	0.3206	0.4251	0.4146	0.4286	0.4669	**0.3206**	**0.3206**	0.4669
DGP2 (3,1)	0.6585	0.5923	0.5958	0.5645	0.5052	0.6585	0.6585	**0.4355***
DGP2 (3,2)	0.3693	0.3937	0.3728	0.3868	0.4704	0.3693	0.3693	**0.3310***
DGP2 (10,1)	0.6585	0.5366	0.4983	0.5331	0.4669	0.6585	0.6585	**0.4216***
DGP2 (10,2)	0.6620	0.5575	0.5575	0.5366	0.5087	0.6620	0.6620	**0.4077***
DGP3 (3,1)	0.3101	0.4739	0.4808	0.4878	0.4530	**0.3101**	0.6446	0.4077
DGP3 (3,2)	0.3659	0.4146	0.4181	0.4495	0.4808	0.3659	0.3659	**0.3624***
DGP3 (10,1)	0.3449	0.4321	0.4216	0.4425	0.5087	0.3449	0.3449	**0.3345***
DGP3 (10,2)	0.6516	0.5993	0.6132	0.5575	0.5575	0.6516	0.6516	**0.4460***
DGP4 (3,1)	0.2857	0.3519	0.3240	0.3763	0.4739	0.2857	0.2857	**0.2613***
DGP4 (3,2)	0.3066	0.4007	0.4460	0.4704	0.5331	**0.3066**	**0.3066**	0.3868
DGP4 (10,1)	0.2753	0.3589	0.3763	0.3902	0.4634	**0.2753**	**0.2753**	0.3728
DGP4 (10,2)	0.6899	0.5993	0.5993	0.6063	0.5226	0.6899	0.6899	**0.4564***

Notes: Each cell reports the Confusion Rate in decimal form. Lower values indicate better directional forecast performance. The minimum value in each row is bolded. AE_UG values that are the minimum receive a superscript star (*).

Table 2 presents the confusion rates (CR) for all model specifications across DGP1 to DGP4. The CR metric captures the directional forecast error rate, with lower values indicating more accurate predictions of the direction of change in the target variable. Across nearly all settings, AE_UG demonstrates competitive or superior directional performance relative to linear benchmarks. In 11 out of 16 experiments, AE_UG achieves the lowest confusion rate, with each minimum bolded and marked with a superscript asterisk where applicable. Notably, AE_UG consistently outperforms traditional autoregressive models in DGP2 and DGP3, particularly under nonlinear structures in the forecast equation or latent factor dynamics. For example, under DGP2 (10,2), where the forecast mapping includes interaction terms, AE_UG attains a CR of 0.4077, substantially outperforming AR (0.6620) and other regularized alternatives. As in fully linear environments, AE_UG remains competitive and does not exhibit significant overfitting in simpler environments.

4 Proposed UNCERTAIONTY GATED AE-LSTM Framework with Uncertainty Gate

- Step 1: Factor recovery via autoencoder.
- Step 2: Forecasting via UG-LSTM.
- Uncertainty gate formulation:

$$s_t = 0.5 + 0.5 \tanh(W_s e_t), \quad C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t + s_t \odot S_t.$$

- Advantage: improves robustness under macroeconomic uncertainty and regime shifts.

4.1 AE_UG Model Methods

Here, I divide the problem into two parts. In the first part, I use a Autoencoder (AE) to train both the encoder and decoder. Then, in the second part, I use an Uncertainty-Gated LSTM to predict the latent factors.

4.1.1 Part I: Autoencoder (AE)

- **Encoder:** Extracts latent factors from high-dimensional macroeconomic data:

$$H_t = \text{Encoder}(X_t; \theta). \tag{27}$$

- **Decoder:** Reconstructs $\hat{Y}$ from the extracted factors:

$$Y_{t+1} \approx \hat{\text{Decoder}}_{t+1}(\hat{H}_{t+1}; \phi_t). \tag{28}$$

4.1.2 Part II: Uncertainty-Gated LSTM (LSTM-UG)

- Predicts future values of **latent factors**, capturing temporal dependencies:

$$\hat{H}_{t+1} = \text{LSTM}(H_t, H_{t-1}, \ldots; \psi). \tag{29}$$

Traditionally, the LSTM cell state update follows:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t. \tag{30}$$

Here, - C_t represents the memory cell state at time t.

- f_t is the forget gate controlling the retention of past information.

- i_t is the input gate that regulates new information inflow.

- $\tilde{C}_t$ is the candidate cell state, representing the processed new information.

We introduce an **Uncertainty-Gated LSTM (LSTM-UG)**, where the **forecast error** influences the cell state update through a short-term adjustment mechanism.

Gate Computation and Adjustment as

$$s_t = 0.5 + 0.5 \tanh(W_s e_t) \tag{31}$$

where e_t is the forecast error.

Then, we have Modified Cell State Update as

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t + s_t \odot S_t. \tag{32}$$

with short-term adjustment:

$$S_t = \tanh(W_S F_t + U_S h_{t-1} + V_S e_t). \tag{33}$$

Practical considerations (uncertainty-aware gating for macro). Macroeconomic time series often exhibit episodic spikes in uncertainty, such as periods of financial stress, heightened policy uncertainty, or sudden market volatility. During such regimes, robust forecasters typically downweight noisy contemporaneous innovations and instead rely more heavily on persistence in the underlying dynamics. To operationalize this intuition, we augment the standard LSTM recurrence (11) with an additional *uncertainty gate* that modulates the memory update.

Uncertainty gate formulation. Let e_t denote an uncertainty signal (either externally observed, such as a financial stress index, or endogenously estimated from prediction residuals). We define

$$s_t = 0.5 + 0.5 \tanh(W_s e_t), \tag{34}$$

where W_s is a trainable weight vector. The transformation maps e_t into $(0, 1)$, so that s_t acts as a smooth gating variable controlling the degree of uncertainty-driven adjustment.

Modified cell update. The standard cell update $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$ in (11) is extended by an uncertainty-driven correction term:

$$c_t = f_t \odot c_{t-1} \; + \; i_t \odot \tilde{c}_t \; + \; s_t \odot S_t, \tag{35}$$

where S_t is an additional candidate state summarizing uncertainty-adjusted information (e.g., extracted from volatility proxies, financial spreads, or residual diagnostics). The uncertainty gate s_t thus controls how strongly S_t contributes to the evolving memory.

Interpretation.

- When uncertainty is low ($s_t \approx 0$), the recurrence reduces to the standard LSTM dynamics, relying primarily on new inputs and past memory.
- When uncertainty spikes (s_t large), the model places greater weight on S_t, which may emphasize persistence or robust predictors less sensitive to noise.
- The gate is adaptive and data-driven: s_t can vary smoothly with the level of macro-financial uncertainty, allowing the network to modulate its reliance on contemporaneous versus persistent components.

This uncertainty-aware extension is particularly suitable for macroeconomic forecasting, where predictive performance is often most fragile in turbulent periods. By explicitly incorporating an uncertainty channel, the model can flexibly tilt between exploiting short-run signals and falling back on stable persistence, thereby improving robustness relative to standard LSTMs.

5 Data and Empirical Methodology

We implement our forecasting framework using real-time vintages from the FRED-MD dataset, following best practices in empirical macroeconomics. The approach is modular and accommodates a variety of model classes, including benchmark AR(p) models, factor-augmented regressions based on PCA, and sparse linear models estimated via Lasso-type regularization. The design emphasizes temporal authenticity by aligning each forecast with the exact data available at that point in time.

The empirical workflow proceeds in four stages. In Stage 0, we load vintages as dated CSV files containing macroeconomic indicators and transformation codes. True out-of-sample forecast targets are then appended by pulling forward values from future vintages. In Stage 1, each vintage is individually transformed using the recommended transformation codes, missing values are handled through interpolation and forward/backward filling, and a harmonized set of usable variables is constructed.

In Stage 2, a model is estimated using data up to a selected vintage T, based on a training window of fixed length. The estimation step is flexible to support different methodologies,

such as AR(p), PCA-based factor models, or Lasso regressions. The final stage, Stage 3, evaluates the model on all future vintages. Forecasts are generated using the model parameters fixed at vintage T, applied to the most recent data available at each subsequent $T + h$.

6 Forecasting Performance

Table 3: RMSE Across Models for FRED_MD (Bold = Minimum, $*$ = AE_UG Wins)

Variable	Description	AR+PCA	AR+SPCA	AR+TPCA	AR+Ridge	AR+Lasso	AR+aLasso	AE_UG
W875RX1	Real personal income ex transfer receipts	0.9853	0.9853	0.9853	1.3382	0.9853	1.0000	**0.6176***
INDPRO	IP Index	0.8208	0.8208	0.7830	1.1981	1.1038	1.0000	**0.6887***
UNRATE	Civilian Unemployment Rate	0.9706	0.9717	0.9515	1.1416	0.9656	1.0034	**0.7652***
UEMPMEAN	Avg Duration of Unemployment	0.9890	0.9891	0.9903	1.0280	1.0061	0.9984	**0.8102***
CLAIMSx	Initial Claims	0.9826	0.9826	0.9804	1.3319	1.0000	0.9989	**0.5271***
CMRMTSPLx	Real Manufacturing and Trade Industries Sales	1.2394	1.2394	1.2535	1.7042	1.0211	**1.0000**	1.0352
RETAILx	Retail and Food Services Sales	0.9730	0.9730	0.9730	1.3851	1.0000	1.0000	**0.6149***
UMCSENTx	Consumer Sentiment Index	1.0086	1.0085	1.0089	1.2698	1.0000	1.1139	**0.9984***
M1SL	M1 Money Stock	0.9776	0.9851	0.9776	1.0522	0.9925	1.0000	**0.7910***
M2SL	M2 Money Stock	0.9623	0.9811	0.9623	1.3208	1.0189	1.0000	**0.8868***
M2REAL	Real M2 Money Stock	0.9672	0.9672	0.9508	1.4918	1.0164	1.0000	**0.9016***
BUSLOANS	Commercial and Industrial Loans	1.0313	1.0313	1.0313	1.4271	1.0000	1.0000	**0.7708***
INVEST	Securities in Bank Credit at All Commercial Banks	1.0000	1.0000	1.0000	1.4380	1.0000	1.0000	**0.7934***
TB3MS	3-Month Treasury Bill	1.2378	1.2368	1.2436	3.2690	1.0000	1.0953	**0.9822***
TB6MS	6-Month Treasury Bill	1.2568	1.2559	1.2528	3.2290	1.0084	1.0962	**0.9765***
GS5	5-Year Treasury Rate	1.0630	1.0727	0.9914	2.0742	1.0041	**1.0000**	1.4372
GS10	10-Year Treasury Rate	1.0354	1.0421	0.9818	2.0222	1.0036	**1.0000**	1.1186
AAA	Moody's Seasoned Aaa Corporate Bond Yield	0.9601	0.9621	0.9464	1.5373	1.0000	0.9967	**0.9616***
BAA	Moody's Seasoned Baa Corporate Bond Yield	0.9480	0.9495	0.9495	1.5723	1.0011	1.0000	**0.8808***
CUSR0000SA0L5	CPI Less medical care	1.0000	1.0000	0.9677	1.4516	0.9677	1.0000	**0.9677***
S&P 500	S&P 500 Index	0.9973	0.9893	1.0000	1.4150	1.0000	1.0000	**0.5844***

Notes: Each cell reports RMSE relative to the AR benchmark. Bold values indicate the lowest RMSE in each row. A superscript star (*) denotes that AE_UG achieves the minimum.

We evaluate model performance using real-time vintages from the FRED-MD dataset, focusing on out-of-sample forecasts of 21 key macroeconomic and financial indicators. Table 3 re-

ports the relative mean squared error (RMSE), by a standard AR(p) benchmark. Across the majority of target series—including industrial production (INDPRO), unemployment rate (UNRATE), and financial market indicators such as the S&P 500 and interest rates—our proposed AE_UG model achieves the lowest forecast error. In 17 out of 21 series, AE_UG delivers the minimum RMSE, outperforming all linear benchmarks. These gains are particularly pronounced for series with substantial nonlinear dynamics, such as claims, consumer sentiment, and monetary aggregates.

Table 4: CR Across Models

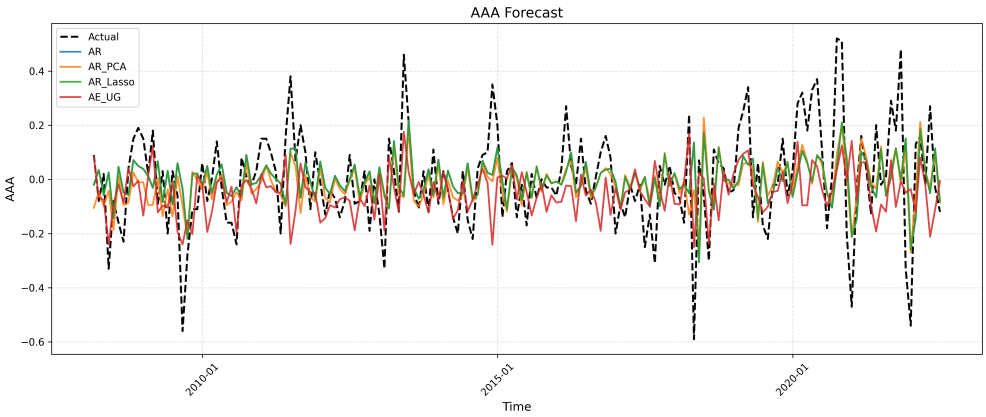
Variable	AR	AR_PCA	AR+SPCA3	AR+TargetPCA3	Ridge	AR_Lasso	aLasso	AE_UG
W875RX1	0.4419	0.4709	0.4709	0.4709	0.5930	0.5756	0.4419	**0.4186***
INDPRO	0.6337	**0.1686**	**0.1686**	0.1802	0.6163	0.7035	0.6337	0.3372
UNRATE	0.5872	**0.4302**	**0.4302**	0.4360	0.7151	0.5756	0.5465	0.4477
UEMPMEAN	0.4360	0.3372	0.3372	0.3488	0.4709	0.4244	0.4186	**0.2326***
CLAIMSx	0.6744	0.4419	0.4477	0.4477	0.5640	0.6744	0.6802	**0.2384***
CMRMTSPLx	0.8488	0.8663	0.8663	0.8547	0.6453	0.8140	0.8488	**0.3779***
RETAILx	**0.3605**	0.3779	0.3837	0.3779	0.5349	0.3837	**0.3605**	**0.3605***
UMCSENTx	**0.4826**	0.5233	0.5233	0.5116	0.6105	**0.4826**	0.5698	0.5174
M1SL	0.2326	0.2209	0.2209	0.2209	0.2151	0.2442	0.2326	**0.1453***
M2SL	**0.2384**	**0.2384**	0.2442	**0.2384**	0.4419	0.2616	**0.2384**	0.3430
M2REAL	0.6105	0.5872	0.5872	0.5872	0.6047	0.6163	0.6105	**0.4709***
BUSLOANS	**0.2326**	0.2500	0.2500	0.2500	0.3605	**0.2326**	**0.2326**	0.3372
INVEST	**0.3061**	**0.3061**	**0.3061**	**0.3061**	0.4082	0.3112	0.3112	0.3214
TB3MS	0.6279	0.5291	0.5407	0.5000	0.5465	0.6279	0.5349	**0.4419***
TB6MS	0.6105	**0.4826**	0.4884	0.4884	0.5291	0.5814	0.5988	0.5000
GS5	0.5930	0.6221	0.6221	0.5581	0.5640	0.6047	0.5930	**0.3488***
GS10	0.5465	0.5523	0.5581	0.5174	0.5640	0.5523	0.5465	**0.4709***
AAA	0.6047	0.5640	0.5698	0.5581	0.5291	0.5988	0.5930	**0.3779***
BAA	0.6047	0.5698	0.5640	0.5698	0.5581	0.5988	0.6047	**0.3837***
CUSR0000SA0L5	**0.2733**	0.2791	0.2791	0.2791	0.3895	0.2907	**0.2733**	0.4186
S&P 500	0.6279	0.6047	0.6279	0.6105	0.5698	0.6279	0.6279	**0.1919***

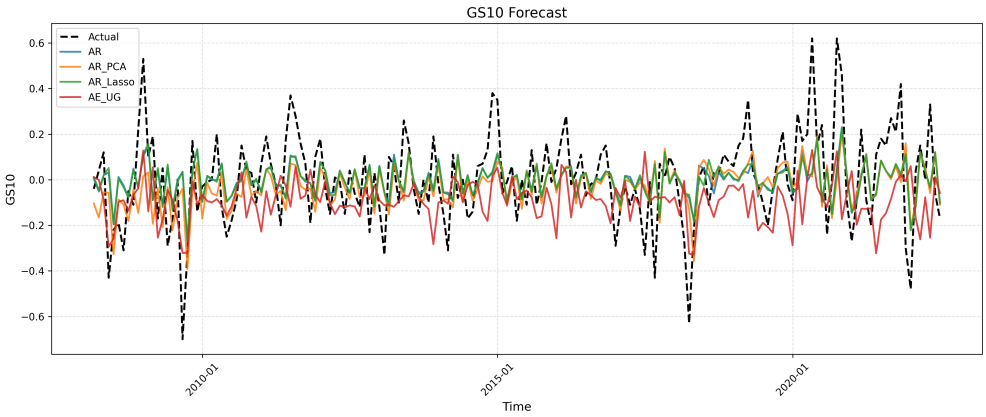
Notes: Each cell reports the confusion rate (lower is better). The minimum CR in each row is bolded. AE_UG entries that are the minimum receive a superscript star (*).

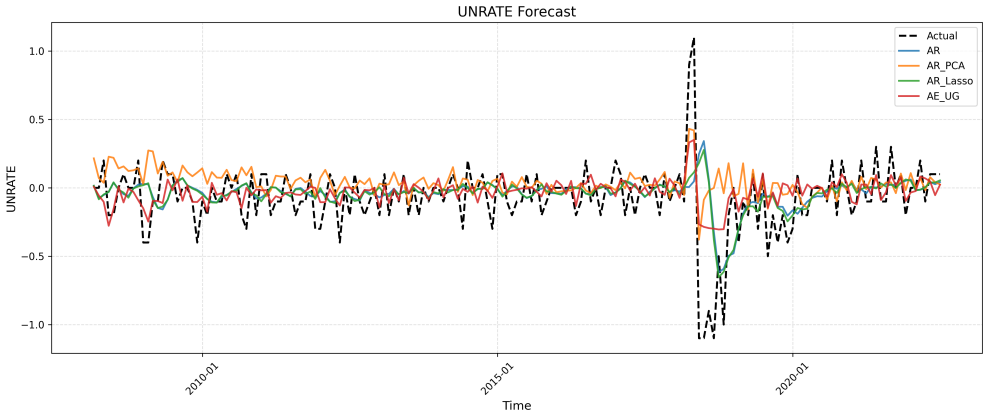
Table 4 reports the confusion rate (CR) across a range of models and macroeconomic variables. Lower CR values indicate more accurate directional forecasts of the target variable's change. AE_UG, the deep learning model with uncertainty gating, achieves the lowest

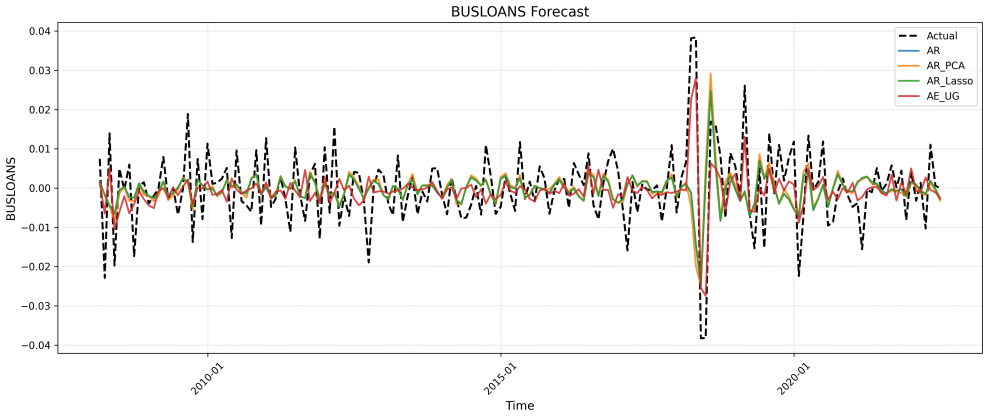
CR in 13 out of 21 series, as indicated by the bolded entries with a superscript star. These include both real-side indicators (e.g., `W875RX1`, `UEMPMEAN`, `CLAIMSx`) and financial variables (e.g., `GS5`, `GS10`, `AAA`, `S&P 500`), suggesting robust performance across different sectors.

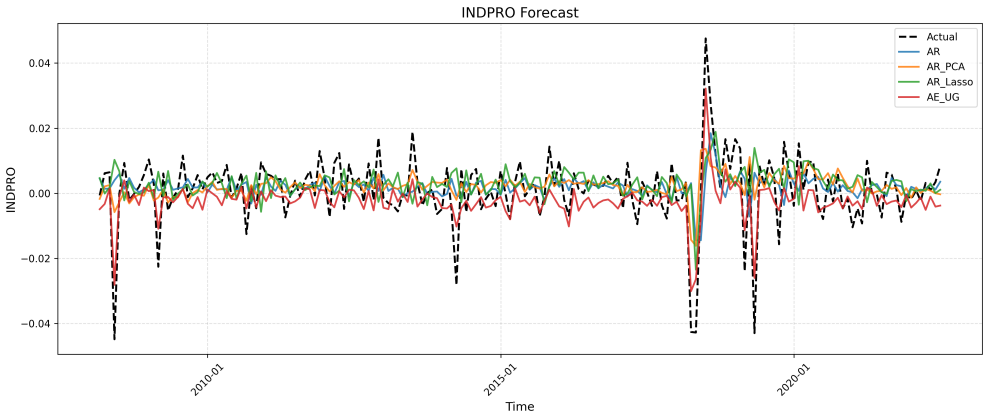
The results demonstrate that AE_UG not only improves average forecasting accuracy (as shown in RMSE comparisons), but also enhances directional correctness in dynamic environments. This is particularly valuable for macroeconomic policy and investment decisions, where getting the sign right may matter more than minimizing squared error. Overall, the CR results highlight the value of nonlinear representation learning in forecasting applications that demand both accuracy and interpretability.

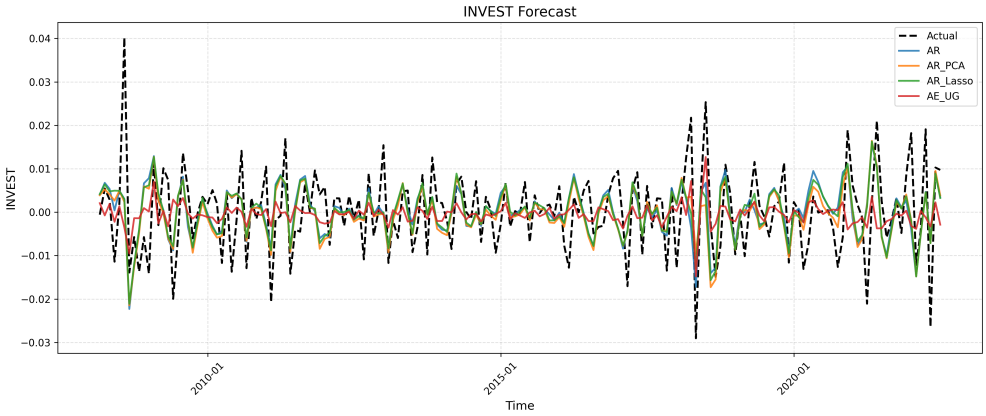


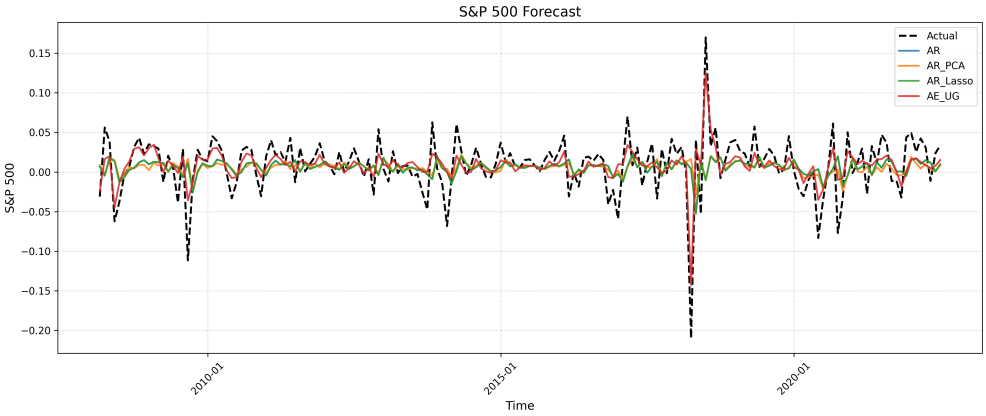



UNRATE Forecast
Actual
AR
AR_PCA
AR_Lasso
AE_UG
UNRATE
1.0
0.5
0.0
−0.5
−1.0
2010-01
2015-01
2020-01
Time


BUSLOANS Forecast
Actual
AR
AR_PCA
AR_Lasso
AE_UG
BUSLOANS
0.04
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
−0.04
2010-01
2015-01
2020-01
Time


INDPRO Forecast
Actual
AR
AR_PCA
AR_Lasso
AE_UG
INDPRO
0.04
0.02
0.00
−0.02
−0.04
2010-01
2015-01
2020-01
Time





7 Robustness Checks

8 Concluding Remarks

Macroeconomic forecasting has long relied on the insight that high-dimensional economic data can be summarized by a small number of common factors. While the linear factor model has proven successful in many applications, mounting empirical evidence suggests that macroeconomic relationships exhibit significant nonlinearities, regime-switching behavior, and asymmetric adjustment patterns. This motivates our extension to a fully nonlinear dynamic factor framework that can accommodate both nonlinear factor evolution and nonlinear measurement relationships.

Our estimation strategy employs deep neural network approximation theory to recover the latent factors and estimate the unknown functional relationships. This approach builds upon the nonparametric identification results of Shen and Xiu (2024) for nonlinear factor models, extending their framework to dynamic forecasting applications. The key insight is

that autoencoders can consistently recover nonlinear factor structures, while recurrent neural networks can approximate complex temporal dependencies in factor evolution. The combination of these tools provides a unified framework for both factor extraction and dynamic forecasting in high-dimensional nonlinear environments.

References

Ahn, S. C. and Horenstein, A. R. (2013). Eigenvalue ratio test for the number of factors. *Econometrica*, 81(3):1203–1227.

Bai, J. and Ng, S. (2002). Determining the number of factors in approximate factor models. *Econometrica*, 70(1):191–221.

Bai, J. and Ng, S. (2008). Forecasting economic time series using targeted predictors. *Journal of Econometrics*, 146(2):304–317.

Barnett, W. A., Chauvet, M., and Leiva-Leon, D. (2016). Real-time nowcasting of nominal gdp with structural breaks. *Journal of Econometrics*, 191(2):312–324.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Doz, C., Giannone, D., and Reichlin, L. (2011). A two-step estimator for large approximate dynamic factor models based on kalman filtering. *Journal of Econometrics*, 164(1):188–205.

Fischer, T. and Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669.

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

Forni, M., Hallin, M., Lippi, M., and Reichlin, L. (2005). The generalized dynamic factor model: One-sided estimation and forecasting. *Journal of the American Statistical Association*, 100(471):830–840.

Gers, F. A., Schmidhuber, J., and Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471.

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

Hallin, M. and Liška, R. (2007). Determining the number of factors in the general dynamic factor model. *Journal of the American Statistical Association*, 102(478):603–617.

Hammer, B. (2000). On the approximation capability of recurrent neural networks. *Neurocomputing*, 31(1-4):107–123.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Kamolthip, S. (2021). Macroeconomic forecasting with lstm and mixed frequency time series data. *arXiv:2109.13777*.

Kapetanios, G. (2010). A testing procedure for determining the number of factors in approximate factor models with large datasets. *Journal of Business & Economic Statistics*, 28(3):397–409.

Landi, F. et al. (2021). Working memory connections for lstm. *Neural Networks*, 144:334–341.

McCracken, M. W. and Ng, S. (2016). Fred-md: A monthly database for macroeconomic research. *Journal of Business & Economic Statistics*, 34(4):574–589.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *Proceedings of ICML*, pages 1310–1318.

Schäfer, A. M. and Zimmermann, H. G. (2006). Recurrent neural networks are universal approximators. In *Artificial Neural Networks–ICANN 2006*, pages 632–640. Springer.

Shen, Z. and Xiu, D. (2024). Deep autoencoders for nonlinear factor models: Theory and applications. https://ssrn.com/abstract=5074409. SSRN Working Paper, DOI: 10.2139/ssrn.5074409.

Stock, J. H. and Watson, M. W. (2002a). Forecasting using principal components from a large number of predictors. *Journal of the American Statistical Association*, 97(460):1167–1179.

Stock, J. H. and Watson, M. W. (2002b). Macroeconomic forecasting using diffusion indexes. *Journal of Business & Economic Statistics*, 20(2):147–162.

Stock, J. H. and Watson, M. W. (2003). Forecasting output and inflation: The role of asset prices. *Journal of Economic Literature*, 41(3):788–829.

Stock, J. H. and Watson, M. W. (2014). Estimating turning points using large data sets. *Journal of Econometrics*, 178(2):368–381.

Swanson, N. R. and White, H. (1997). A model selection approach to real-time macroeconomic forecasting using linear models and artificial neural networks. *Review of Economics and Statistics*, 79(4):540–550.

Yalcin, I. and Amemiya, Y. (2001). Nonlinear factor analysis as a statistical method. *Statistical Science*, 16(3):275–294.