Deep Vector Autoregression for Macroeconomic Data

Marc Agustí (marc.agusti@barcelonagse.eu)
Patrick Altmeyer (patrick.altmeyer@barcelonagse.eu)
Ignacio Vidal-Quadras Costa (ignacio.vidalquadrascosta@barcelonagse.eu)

mayo, 2021

Abstract

Vector autoregression models have been the traditional technique in the last decades when it comes to the forecasting of time series data. In particular, the VAR framework has demonstrated its outperformance in precision and accuracy of predicting the next lags of time series, hence becoming a useful tool for policy makers, whcih rely on this methodology to construct forecasts or even for the analysis of impulse response functions. However, with the recent advancement in computational power of computers, and more importantly, the development of more advanced machine learning algorithms and approaches such as deep learning, new algorithms are developed to analyze and forecast time series data. This paper aims to contribute to the time series literature by using the Long Short-Term Memory (LSTM) to see whether this new advanced methodology is superior to the usual VAR framework. By fitting each regression of the VAR with a neural network instead of with a simple OLS regression, we are able to outperform both in-sample and out-of-sample the predictions of the usual VAR for variables such as GDP, FED funds rate and inflation, ...

Keywords— Vector Autoregression, Deep Learning, Neural Networks, Macroeconomic Timeseries

1 Literature review

Does monetary policy affect the real economy? There is a large agreement among economists on the fact that monetary policy has a short-term influence on the economic activity. Friedman and Schwartz (2008) found that monetary policy actions are followed by movements in real output that may last for two years or more (Romer and Romer (1989); Bernanke (1990)). However, what are the forces that trigger this effect is of interest for most economists, in particular, economists aim to understand the monetary transmission mechanism. If monetary policy affects the real economy, what is the transmission mechanism by which these effects occur? This is one of the questions which is among the most important and controversial in macroeconomics.

Given the importance of this question, measuring accurately the effects of changes in monetary policy on the economy is of foremost importance. Doing this accurately has a positive impact in policy-making and also helps economists to choose the right macroeconomic theory when trying to get a better understanding of the underlying mechanisms of the economic system.

In the aftermath of the oil price shock in the 1970's, interest was raised in understanding business cycles. To do that, most economists made use of large-scale macroeconomic models, which was criticized by Lucas Jr (1976), stating that the assumption of invariant behavioral equations was inconsistent with the dynamic maximizing behavior. Hence, New Classical economists started making use of the so-called market clearing models of economic fluctuations. With the goal of really taking into account productivity shocks, Real Business Cycle models were developed (Kydland and Prescott (1982)).

After the failure of the large-scale macroeconomic models when trying to predict business cycles, the economic profession tried to solve this by means of the use of structural vector autoregression (VAR) models to analyze business cycles, which were useful to capture the impact of policy-actions. Sims and others (1986) suggested that VARs were useful to evaluate macroeconomic models. One of the advantage of VARs is that they are not a large and complicated structure, and hence are easily interpretable (do not suffer from the "black box" problem).

In the last decades the use of VAR's in order to do time series forecasting has been quite extensive. Actually, a lot of different models have been proposed with the intention to model and predict time series data. When it comes to the VAR framework, the different factors in the projected VAR models are difficult to understand, and that is why researchers rely heavily on impulse response functions (IRF) (Enders (2008)).

As for now, the models we have seen are more classical econometric based models that are not able to capture nonlinear relationships in the data, which might be sometimes a limitation. In the case of economic time series, specifically gdp, inflation and so on, nonlinear trends are likely to appear and be present in the essence of the data generating process as shown by Brock et al. (1991).

In the past years, authors have therefore started using nonlinear techniques for forecasting. Machine Learning has contributed a lot to this field. The most popular machine learning techniques which do not assume a linear relationship between inputs and outputs are K-Nearest Neighbors (first introduced by Fix and Hodges (1951)), Support Vector Machines (mostly developed by Cortes and Vapnik (1995)), Random Forests (first introduced in 1995 by Ho (1995)) and Neural Networks (NN) (first proposed in 1943 by McCulloch and Pitts (1990)).

The most recent and complex algorithm of the ones mentioned above is NN. NN are a relatively new nonlinear technique to which a lot of authors have made contributions. One of the main advantages of NN compared to the linear models is that they can approximate any nonlinear functions without any apriori information about the properties of the data series. In our case we are interested in the application of NN to time series.

The main idea of NN is to reproduce the inner working of the human mind. NN consist of thousands or even millions of simple processing neurons that are densely interconnected. These neurons are organized in layers. A NN contains an input layer, one or more hidden layers and an output layer and information flows from one layer to another using weight. One neuron receives information form other neurons in the previous layer and passes the processed information into the next layer. This information can flow only in one direction (from the previous layer to the next layer), which means the NN in "feed-forward" or can retrieve information. Each connection between neurons is weighted. When the network is active, the neuron receives a different data (information) from each of its connections and multiplies it by the associated weight. It then adds the resulting products together and passes this output through an activation function that maps the output, yielding the final output. Finally, this output is then passed onto the next neurons. This processed is repeated until we reach the output layer.

With this new algorithm coming into play, Zhang, Patuwo, and Hu (1998) used NN for forecasting. Recently, artificial neural networks (ANN) have played attention enhancing devotions in the field of time series predicting (Hamzaçebi (2008), Zhang (2003), Kihoro, Otieno, and Wafula (2004)). In particular, ANNs have the advantage of not assuming

the statistical distribution followed by the values, being able of proficiently capturing non-linearities. That is, they are self-adaptive (Zhang, Patuwo, and Hu (1998), Zhang (2003)).

A class of ANN is the called recurrent neural network (RNN). RNN allows to use previous outputs as inputs, this allows the model to retain information about the past, making it very efficient for time series. This has been shown by Dorffner (1996). In this article, the author highlights the power of RNN for forecasting compared with standard linear models.

For this reason, a lot of authors interested to forecast economic series have compared linear models with nonlinear models. In economic time series, in the short run, the series is expected to behave more or less the same way it has been behaving up to this point, but on the other hand, if we are interested in forecasting at a big window, then then is when chaos and instability appear, meaning that nonlinear relations may arise, making it more appealing to use ANN as they are capable of identifying these turning points as they do not assume a linear relationship of inputs and outputs.

This was shown by a recent paper of the Bank of England, Joseph et al. (2021). In this paper they run a horse race for forecasting inflation among different horizons comparing the performance of linear and nonlinear algorithms. The results support out hypothesis that NN and other nonlinear Machine Learning algorithms are useful for forecasting at a longer horizon given that, the longer the horizon, the more likely it is to find the chaos and instability Brock et al. (1991) talked about. And, as previously exposed, these turning points are hard to spot with linear relationships of inputs and outputs, while they might be easier to spot with nonlinear models, like SVM or NN.

One of the problems of RNN is the long-term dependency. To illustrate it with an example, some decisions are made taking into account information that happened way back in the past. RNN struggle to keep this very old information threfore limiting its forecasting power. In order to solve this problem, Hochreiter and Schmidhuber (1997) introduced the LSTM in the paper Long Short-Term Memory. This is the reason why a lot of authors use this type of RNN when forecasting any kind of time series.

Yet, when we are interested in the monetary transition mechanism, we are not just interested in the forecasting accuracy of the model we are use but we are also interested in the inference. Central banks need to know if rates granger cause one variable or not or they also need to know the IRF in order to implement the correct monetary policy.

The linear additive relationship of linear models allows the model to observe which variable granger cause another and what are the IRF of each variable with respect to another. Unfortunately, in the case of nonlinear models, its nature makes it impossible to recover these insights because the outputs and the inputs do not have a linear additive structure.

Therefore, on the one hand, the nonlinear structure of NN helps us for forecasting in the case that there are some nonlinear relationships in the series, but on the other hand we lose the interpretability of the model, making it impossible to recover IRF or to even to assess if one variable granger causes another. This problem is also known as the black box problem. This is because the data fed into the input layer passes through the succeeding layers, getting multiplied, added together and transformed in complex and different ways, until it finally arrives, radically transformed, at the output layer. Therefore, it is impossible to assess what happened with one input and how it affected the output.

Whether neural networks can do the job that so far has been done by VARs (and its extensions) is not so clear. In the past years some part of the research in this field has been put towards the study of neural networks for time series modelling. The research question investigated in this article is that whether and how the newly developed deep learningbased algorithms for forecasting time series data, such as LSTM are superior to the traditional algorithms such as VAR. Therefore, assessing the accuracy of forecasts is necessary when employing various forms of forecasting methods, and more specifically forecasting using regression analysis as they have several limitations in applications

2 Methodology

In conventional Vector Autoregression (VAR) dependencies of any system variable on past realizations of itself and its covariates are modelled through linear equations. This corresponds to a particular case of the broader class of Deep Vector Autoregressions investigated here and will serve as the baseline for our analysis.

2.1 Vector Autoregression

Let $\mathbf{y}_t$ denote the $(K \times 1)$ vector of variables at time t. Then the VAR(p) with p lags and a constant deterministic term is simply a linear system of stochastic equations of the following form:

$$\mathbf{y}_t = \mathbf{c} + \mathbf{A}_1\mathbf{y}_{t-1} + \mathbf{A}_2\mathbf{y}_{t-2} + ... + \mathbf{A}_p\mathbf{y}_{t-p} + \mathbf{u}_t \tag{1}$$

The matrices $\mathbf{A}_m$, $m \in \{1, ..., p\}$ contain the reduced form coefficients and $\mathbf{u}_t$ is a vector of errors for which $\mathbb{E}\mathbf{u}_t$, $\mathbb{E}\mathbf{u}_t\mathbf{u}_t^T = \Sigma$ and $\mathbb{E}\mathbf{u}_t\mathbf{u}_s^T = \mathbf{0}$ for all $t \neq s$. We refer to (1) as the *reduced form* representation of the VAR(p) because all right-hand side variables are predetermined (Kilian and Lütkepohl 2017).

To facilitate the discussion of Deep VARs below it is helpful to be explicit about how each individual time series is modelled. In particular, it follows from (1) that

$$y_{it} = c_i + \sum_{m=1}^{p}\sum_{j=1}^{K} a_{jm}y_{jt-m} + u_{it}$$

which corresponds to the key modelling assumption that at any point in time t any time series $i \in 1, ..., K$ is just a weighted sum of past realizations of itself and all other variables in the system. This assumption makes the estimation VAR(p) processes remarkably simple. Perhaps more importantly, the assumption of linearity also greatly facilitates inference about VARs.

For implementation purposes and in order to state stationarity conditions it is convenient to restate the K-dimensional VAR(p) process more compactly in companion form as

$$\mathbf{Y}_t = \begin{pmatrix} \mathbf{c} \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \mathbf{A}\mathbf{Y}_{t-1} + \begin{pmatrix} \mathbf{u}_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2}$$

where $\mathbf{Y}_t = (\mathbf{y}_t^T, ..., \mathbf{y}_{t-p+1}^T)^T$ and $\mathbf{A}$ is referred to as the companion matrix (Kilian and Lütkepohl 2017). Letting $\mathbf{Z} = (\mathbf{1}, \mathbf{Y})$ we have a simple closed form solution for estimating reduced form coefficients through ordinary least squares (OLS):

$$\widehat{\left(\begin{pmatrix} \mathbf{c} \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \mathbf{A}\right)} = \left(\mathbf{Z}_{t-1}^T\mathbf{Z}_{t-1}\right)^{-1}\mathbf{Z}_{t-1}^T\mathbf{Y}_t \tag{3}$$

2.2 Model selection

The simplicity that characterizes the linear VAR model

2.2.1 Stationarity

When working with time series we are generally concerned about stationarity. Broadly speaking stationarity ensures that the future is like the past and hence any predictions we make based on past data adequately describe future outcomes. In the context of VARs stationarity is follows from stability: a VAR(p) is stable if the effects of shocks to the system eventually die out. Stability can assessed through the system's autoregressive roots or equivalently by looking at the eigenvalues of companion matrix $\mathbf{A}$ (Kilian and Lütkepohl 2017). In particular, for the VAR(p) in (2) to be stable we condition that the Kp eigenvalues λ that satisfy

$$\det(\mathbf{A} - \lambda\mathbf{I}_{Kp}) = 0$$

are all of absolute value less than one. Stability implies that the first and second moments of the VAR(p) process are time-invariant, hence ensuring weak stationarity (Kilian and Lütkepohl 2017).

A straight-forward way to deal with stationarity of VARs is to simply ensure that the individual time series entering the system are stationary. This usually involves differencing the time series until they are stationary: for any time series y_i that is integrated of order $I(\delta)$, there exists a δ-order difference that is stationary. An immediate drawback of this approach is the loss of information contained in the levels of the time series. Modelling approaches that take into account conintegration of individual time series can ensure system stationarity and still let individually non-stationary time series enter the system in levels (Hamilton 1994).

2.2.2 Lag order

2.3 Deep Vector Autoregression

2.4 Recurrent Neural Networks (RNN)

Recurrent neural networks are based on the idea of persistent thoughts: thinking is modelled as a continuous process that instead of continuously reinventing itself and starting from scratch, evolves gradually and at each step uses information about its prior states. This hierarchical, chain-like nature of RNNs makes them particularly useful for problems that involves sequences, for example, speech recognition or time series analysis. The latter is the focus of this paper, so let us dwell on this a little further. Specially in time series analysis, taking into account prior information that emerges from the context is of foremost importance. It can be really difficult to learn the evolution of a time series of a single time series just from its closest pasr: to understand the evolution of GDP or inflation in the context of a recession, it is not enough to look at a few menths earlier on time, but to look at the whole picture. In fact, a too small choice of the context window may lead to wrong conclusions about the future evolution of a time series. In order to account for long-term dependencies we can use a Long Short Term Memory (LSTM) network, a special kind of RNN.

2.4.1 Long Short-Term Memory (LSTM)

Long Short-Term Memory networks were introduced by Hochreiter and Schmidhuber (1997).

3 Empirics

References

Bernanke, Ben. 1990. "The Federal Funds Rate and the Channels of Monetary Transnission." National Bureau of Economic Research.

Brock, William A, David Arthur Hsieh, Blake Dean LeBaron, William E Brock, and others. 1991. *Nonlinear Dynamics, Chaos, and Instability: Statistical Theory and Economic Evidence*. MIT press.

Cortes, Corinna, and Vladimir Vapnik. 1995. "Support-Vector Networks." *Machine Learning* 20 (3). Springer: 273–97.

Dorffner, Georg. 1996. "Neural Networks for Time Series Processing." In *Neural Network World*. Citeseer.

Enders, Walter. 2008. *Applied Econometric Time Series*. John Wiley & Sons.

Fix, E, and J Hodges. 1951. "An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation." *International Statistical Review* 3 (57): 233–38.

Friedman, Milton, and Anna Jacobson Schwartz. 2008. *A Monetary History of the United States, 1867-1960*. Princeton University Press.

Hamilton, James Douglas. 1994. *Time Series Analysis*. Princeton university press.

Hamzaçebi, Coşkun. 2008. "Improving Artificial Neural Networks' Performance in Seasonal Time Series Forecasting." *Information Sciences* 178 (23). Elsevier: 4550–9.

Ho, Tin Kam. 1995. "Random Decision Forests." In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1:278–82. IEEE.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9 (8). MIT Press: 1735–80.

Joseph, Andreas, Eleni Kalamara, George Kapetanios, and Galina Potjagailo. 2021. "Forecasting Uk Inflation Bottom up."

Kihoro, J, RO Otieno, and C Wafula. 2004. "Seasonal Time Series Forecasting: A Comparative Study of Arima and Ann Models." African Journal of Science; Technology (AJST).

Kilian, Lutz, and Helmut Lütkepohl. 2017. *Structural Vector Autoregressive Analysis*. Cambridge University Press.

Kydland, Finn E, and Edward C Prescott. 1982. "Time to Build and Aggregate Fluctuations." *Econometrica: Journal of the Econometric Society*. JSTOR, 1345–70.

Lucas Jr, Robert E. 1976. "Econometric Policy Evaluation: A Critique." In *Carnegie-Rochester Conference Series on Public Policy*, 1:19–46. North-Holland.

McCulloch, Warren S, and Walter Pitts. 1990. "A Logical Calculus of the Ideas Immanent in Nervous Activity." *Bulletin of Mathematical Biology* 52 (1-2). Springer: 99–115.

Romer, Christina D, and David H Romer. 1989. "Does Monetary Policy Matter? A New Test in the Spirit of Friedman and Schwartz." *NBER Macroeconomics Annual* 4. MIT Press: 121–70.

Sims, Christopher A, and others. 1986. "Are Forecasting Models Usable for Policy Analysis?" *Quarterly Review* 10 (Win). Federal Reserve Bank of Minneapolis: 2–16.

Zhang, G Peter. 2003. "Time Series Forecasting Using a Hybrid Arima and Neural Network Model." *Neurocomputing* 50. Elsevier: 159–75.

Zhang, Guoqiang, B Eddy Patuwo, and Michael Y Hu. 1998. "Forecasting with Artificial Neural Networks:: The State of the Art." *International Journal of Forecasting* 14 (1). Elsevier: 35–62.

A R Code and Package

All code used for the empirical analysis presented in this article can be found on the corresponding GitHub repository. Researchers interested in using Deep VARs more generally for their own empirical work may find the R `deepvars` package useful which is being maintained by one of the authors. The package is still under development and as of now only available on GitHub. To install the package in R simply run:

```
devtools::install_github("pat-alt/deepvars", build_vignettes=TRUE)
```

Package vignettes will take you through the basic package functionality. Once the package has been installed simply run `utils::browseVignettes()` to access the documentation.