The Compustat dataset is a frequently used dataset in corporate finance and macroeconomics due to its long observation history, dating back to the 1960s for some firms, wide coverage of nearly all U.S. public firms, and detailed data on hundreds of balance sheet items. Compustat generates its data mostly from the legally required regulatory filings of public firms.

One of the main challenges when using the Compustat database is the abundant precesses of missing values for specific balance sheet items. This problem is especially acute for smaller firms and items that lie deeper in the balance sheet. These missing values can be due to either the firm reporting only annual data for a period of time, leaving three of the four quarters missing for some years, or the firm not reporting specific data items. Researchers handle these missing values by either only using annual data, naive interpolation, limiting the balance sheet items they use in their analysis or by dropping entire firms with many missing values. Missing values become even more problematic when one wants to exploit the panel structure of Compustat and conduct time series analysis where a missing value at time t affects the analysis at neighboring times as well.

This paper proposes a novel method from the machine learning literature to interpolate missing values in Compustat by training a Variational Recurrent Neural Net (VRNN) auto encoder. For a given firm, the VRNN can exploit correlations between data items in the same time period, and correlations across time periods allowing it to use all the information available to predict the value of a balance sheet item that is missing. The VRNN can also handle the large discrete jumps which can occur between time periods for an individual firm.

1 Model

Our model follows that in Chung et al. (2015). It is a variational recurrent neural net autoencoder which embeds a variational autoencoder within a hidden state model. The variational autoencoder compresses information about the cross section of balance sheet items within a firm into a probabilistic sparse representation while the hidden state model introduces time dependency.

To begin we will outline the variational autoencoder (VAE) that underlines our VRNN. A VAE consists of two random variables, whose individual observations are vectors, the data x and an unobserved latent random variable z which have the joint distribution that can be decomposed as

$$p(x, z) = p(x|z)p(z) \tag{1}$$

This model is common in the Bayesian framework and for simple conditional distributions, $p(x|z)$ is relatively straight forward to estimate with traditional Bayesian Markov chain Monte Carlo methods; however, in the VAE framework we usually allow $p(x|z)$ to be a complex distribution, which does not allow us to use traditional Bayesian methods.

In the VAE the prior distribution $p(z)$ is invariant across time periods and we assume it to be a simple multivariate Gaussian distribution, with each element uncorrelated with each

other,

$$z_i| \sim \mathcal{N}(0, I) \tag{2}$$

The conditional distribution $p(x|z)$ is given a parametric form but is allowed to be arbitrarily complex. In our case, we have model it as

$$x_i|z_i \sim \mathcal{N}(\mu_x, \text{diag}(\sigma_x)) \text{ where } [\mu_x, \sigma_x] = \varphi^{\text{dec}}(z_i) \tag{3}$$

$\varphi^{\text{dec}}(z_i)$ is modeled as neural net. This complexity makes calculating the posterior $p(z|x)$ intractable. The VAE instead uses a variational approximation, $q(z|x)$, in place of $p(z|x)$, with our approximation taking the form

$$z_i|x_i \sim \mathcal{N}(\mu_z, \text{diag}(\sigma_z)) \text{ where } [\mu_z, \sigma_z] = \varphi^{\text{enc}}(x_i) \tag{4}$$

where $\varphi^{\text{enc}}(x_i)$ is also modeled as neural net.

Generally in this type of model the dimensionality of z_i will be much smaller than x_i, forcing the model to collapse the data into the most important features, possibly in a non-linear way, that still retains as much information as possible about the original data. The model is estimated by estimating the encoder neural net $\varphi^{\text{enc}}(x_i)$, and the decoder neural net $\varphi^{\text{dec}}(z_i)$ jointly to minimize the divergence between the estimated $p(x_i|z_i)$ and $q(x_i|z_i)$ and their true distributions by maximizing the evidence lower bound. The evidence lower bound (ELBO) for this problem is [1] $-KL(q(z|x)||p(z)) + \mathbb{E}_{q(z|x)}[\log p(x|z)]$.

1.1 Variational Recurrent Neural Net Autoencoder

The variational recurrent neural net autoencoder takes the variational autoencoder above and combines it with a hidden state model to allow for time dependence. Similarly to a Kalman filter or Hidden Markov Model it does so by introducing a latent state variable which evolves over time, but unlike those models, in a non-linear way. In this section we will write the model explicitly as only a time of t, but naturally extends to a panel data setting.

The first alteration of the VAE is conditioning the prior on the latent random variable so that the mean and variance depend on the state variable at the previous period.

$$z_t|h_{t-1} \sim \mathcal{N}(\mu_{0,t}, \text{diag}(\sigma_{0,t})), \text{ where } [\mu_{0,t}, \sigma_{0,t}] = \varphi^{\text{prior}}(h_{t-1})$$

$\varphi^{\text{prior}}(h_{t-1})$ is modeled as a neural net.

The posterior distribution will also be conditional on the latent state variable.

$$z_t|x_t, h_{t-1} \sim \mathcal{N}(\mu_{z,t}, \text{diag}(\sigma_{z,t})), \text{ where } [\mu_{z,t}, \sigma_{z,t}] = \varphi^{\text{enc}}(\varphi^x(x_t), h_{t-1})$$

Similarly for the decoder distribution we have

$$x_t|z_t, h_{t-1} \sim \mathcal{N}(\mu_{x,t}, \text{diag}(\sigma_{x,t})), \text{ where } [\mu_{x,t}, \sigma_{x,t}] = \varphi^{\text{dec}}(\varphi^z(z_t), h_{t-1})$$

[1] This is derived in appendix A.

Finally we have the state update function

$$h_t = f(\varphi^x(x_t), \varphi^z(z_t), h_{t-1})$$

We model $f(\cdot)$ as a Long Short Term Memory (LSTM) cell which is widely used in the broader machine learning literature for its numerical stability and robustness in wide range of applications (Hochreiter and Schmidhuber, 1997). $\varphi^x(x_t)$ and $\varphi^z(z_t)$ are referred to as the feature extractors of x_t and z_t and are common to the encoder, decoder and state update function.

The ELBO for this model is a sum of per-period ELBOs

$$\sum_{t=1}^{T} -KL(q(z_t|x_{\leq t}, z_{<t})||p(z_t|x_{<t}, z_{<t})) + \mathbb{E}_{q(z_{\leq T}|x_{\leq T})}[\log p(x_t|z_{\leq t}, x_{<t})]. \tag{5}$$

with the individual terms implicitly given by the recursive model above. Like the standard VAE, each distribution and the hidden state update function are jointly estimated by maximizing the variational lower bound.

1.2 Prediction

Once the autoencoder has been estimated on data with no missing values, we can use it to interpolate data for firms with missing data using expectation maximization (EM) methods. For each firm we initialize the hidden state h_0 and then run the following algorithm for each observation with no missing data:

1. Update the value of the latent random variable as $z_t = \mu_{z,t}$
2. Update the value of the latent state variable as $h_t = f(\varphi^x(x_t), \varphi^z(z_t), h_{t-1})$
3. Proceed to next time period

For an observation with a missing data point we instead do the following:

1. Update the value of the latent random variable according to the prior, $z_t = \mu_{0,t}$
2. Update the missing values in x_t equal to mean of the decoder distribution $x_t = \mu_{x,t}$
3. Update the value of the latent random variable as $z_t = \mu_{z,t}$, using the full values of x_t to calculate $\mu_{z,t}$
4. Repeat steps 2 and 3 until the missing values in x_t converge
5. Update the value of the latent state variable as $h_t = f(\varphi^x(x_t), \varphi^z(z_t), h_{t-1})$
6. Proceed to next time period

Similar to the Kalman filter, the LSTM model accumulates information about the firm as it proceeds through each time period. While the estimation procedure accounts for the full information content of the data in all periods when estimating the value of the hidden state at each period, during the prediction phase, it only uses information up to that point in time. This leads to less accurate predictions in early periods. To circumvent this, we separately estimate two versions of the VRNN model, with the only difference being that the flow of time is reversed so that the data is fed in backwards to the second model. The final imputed value of the missing values in x_t are a weighted average of the predictions from the forward and backward models with the weights given the relative location in the chain, $\frac{t}{T}$ and $\frac{T-t}{T}$. This can be viewed as an ad hoc approximation to more formal methods which jointly estimate the two LSTM models in one path estimation and then use the joint prediction of x_t from within the single model.

2 Compustat Data

The data that we use are the size of the firm as measured by log(assets) and financial ratios relating to balance sheet and cash flow items. These measures are generally well behaved, but we do apply some precleaning by setting the minimum value of assets to 0.01 before calculating any financial ratios and winsorizing the financial ratios at the 97.5 percentile and at 0 for some of the financial ratios. We then formally model each item as

$$x_{jit} = \delta_{jst} + g(x_i)_{jt} + \epsilon_{jit} \tag{6}$$

where i denotes the firm, t denotes time, j denotes the particular variable and s denotes the sector of the firm according to its 2-digit SIC code. δ_{jst} is the sector-time fixed effect for that firm's balance sheet item while $g(x_i)$ is the predictions from the VRNN run with that firm's data. This model is estimated in a two-step procedure where we first demean the data within each sector-time cell and then the VRNN is trained on this demeaned data. In order to keep the properties of the training and prediction data consistent the sector-time fixed effects are calculated using every non-missing observation in the full dataset for the estimation and prediction steps. The VRNN is trained on observations from firms that have 40 continuous observations with no missing values after January 1, 1990 and uses only the first 40 observations to produce a balanced panel which allows for significant speed up in estimation time. To increase the panel size we linearly interpolate missing values of financial ratios if only one quarter is missing which produces a significantly larger panel with minimal interpolation.

References

Chung, Junyoung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. 2015. "A Recurrent Latent Variable Model for Sequential Data." *CoRR*, abs/1506.02216.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation*, 9(8): 1735–1780.

A Variational Inference

The central object of interest in Bayesian statistics is the posterior distribution of latent variables, z given the data x which is given by Bayes Theorem:

$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x,z)\,\mathrm{d}\,z} \tag{7}$$

Calculating the denominator is challenging for complicated models. Variational inference is an alternative method which aims to approximate $p(z|x)$ with a simpler distribution $q(z;\lambda)$, where λ parametrizes the model and is chosen to produce the optimal approximation. The key idea to variational inference is that the parameters λ can be optimized without knowing the posterior distribution. In the case of the variational autoencoder we use the conditional distribution within the approximation $q(z|x;\lambda)$. For the rest of this paper we drop the explicit notation of λ in $q(z|x)$ for notational simplicity.

The measure of divergence between the approximate and posterior distributions is usually taken to be the Kullback-Leibler divergence which for our problem can be written as

$$KL(q(z|x)||p(z|x)) = \int q(z|x)\log\frac{q(z|x)}{p(z|x)}\,\mathrm{d}\,z \tag{8}$$

Calculating this directly is infeasible as $p(z|x)$ is what we do not know. Instead we can rewrite this as

$$\begin{aligned}\int q(z|x)\log\frac{q(z|x)}{p(z|x)}\,\mathrm{d}\,z &= \int q(z|x)\left(\log\frac{q(z|x)}{p(x,z)} + \log p(x)\right)\mathrm{d}\,z \\ &= -\int q(z|x)\log\frac{p(x,z)}{q(z|x)}\,\mathrm{d}\,z + \log p(x)\end{aligned}$$

The term $\int q(z|x)\log\frac{p(x,z)}{q(z|x)}\,\mathrm{d}\,z \equiv \mathcal{L}$ is called the evidence lower bound (ELBO) and the above expression shows that minimizing the Kullback-Leibler divergence between $q(z|x)$ and $p(z|x)$ is the same as maximizing the ELBO. We can further rewrite the ELBO as follows

$$\begin{aligned}\mathcal{L} &= \int q(z|x)\log\frac{p(x|z)p(z)}{q(z|x)}\,\mathrm{d}\,z \\ &= \int q(z|x)\left(\log\frac{p(z)}{q(z|x)} + \log p(x|z)\right)\mathrm{d}\,z \\ &= -KL(q(z|x)||p(z)) + \mathbb{E}_{q(z|x)}[\log p(x|z)]\end{aligned}$$

The prior $p(x)$ is specified outside of the model, so calculating this quantity for a given $q(z|x)$ is feasible, which therefore allows us to optimize over it to maximize the ELBO.

When optimizing our model, $KL(q(z|x)||p(z))$ can be calculated directly since both distributions are Normal, allowing for a closed form solution. The second term can be approximated with Monte Carlo methods by drawing a $z_i|x_i$ from the proposed distribution $q(z|x)$ and calculating $\log p(x_i|z_i)$ for each observation, and then averaging these probabilities over the observations.