Hybrid Autoregressive-RNN Algorithm for Cryptocurrency Pricing

Joel Persson (joelpe@),[2] Ian Shaw (ieshaw@),[1] Andrew Slottje (slottje@)[1]

(1) Institute for Computational and Mathematical Engineering and (2) Management Science & Engineering, Stanford University

Objectives

- **Discover arbitrage opportunities** in cryptocurrency markets using exchange trading data to predict returns.
- **Attenuate cryptocurrency market volatility** through the use of this algorithm to correct mispricings.

Background

Predicting asset prices is a difficult problem because of the "no arbitrage principle," which states that markets price out excess returns. There is also limited previous research on cryptocurrency price prediction. Previous findings indicate suitability of LSTM models for this type of problem. [1, 2] One such model, the "R2N2," uses vector autoregression as RNN input. [3] We adapt this model to a multiple-asset cryptocurrency setting and extend the model architecture to include residuals from a VARMAX regression on the feature set.

Data and Feature Selection

We use data published by the founder of Bittrex, an online cryptocurrency exchange. The data comprise hourly observations of various trading data as well as BTC-denominated price series in several coins. We select the top 5 by capitalization at initial date.

- **Features**: Volume, base volume, and spread in selected coins, with vector autoregression residuals as additional features.
- **Response**: Log hourly returns in 5 coins.
- **Data splitting**: **Training set** over 9/2015 - 12/2016 (**70%**), **cross-validation set** over 12/2016 - 3/2017 (**15%**), **test set** over 3/2017 - 6/2017 (**15%**).
- **Standardization**: We use `StandardScaler()` from the sklearn library to normalize and center the feature set.

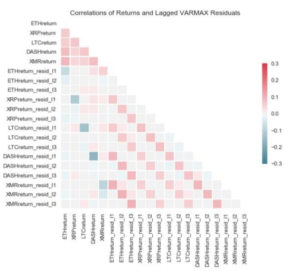

Figure: Correlation of returns with VARMAX-estimated residuals. Note the cross-coin correlation and strong inverse response to previous term's residual.

VARMAX Model Development

Vector autoregressive methods are used for multidimensional time series problems. In the R2N2 algorithm, residuals from the VAR prediction are used as inputs to an RNN. [3] We expand this method to include an adaptation of the **more flexible and dynamic** VARMAX model, which allows for moving average terms and exogenous predictors: [4]

$$y_t = c + A_1 y_{t-1} + \cdots + A_p y_{t-p} + B x_{t-1} + \varepsilon_t + \Sigma_{i=1}^{q} M_i \varepsilon_{t-i}$$

We specify our feature set as the exogenous predictors (x_t) and the 5 coins' returns as endogenous ($y_t \in \mathbb{R}^{5\times 1}$). We provide **massive dimensionality reduction** in the autoregressive optimization problem. This **important computational improvement** in R2N2 time series prediction is suitable for use in high-dimensional covariate spaces.

We find the moving average terms are non-statistically significant. Following our VAR analysis of AIC-selected optimal lag length, we use a lag of 3 to specify the model.

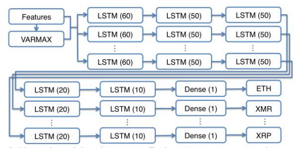

Figure: Calibrated model architecture. Each output represents a binary prediction for significant positive price movement in a single coin in the classification problem and an absolute prediction in the trading problem.

RNN Model Development

Strong results in the literature indicate use of an LSTM network for this problem. LSTM cells are able to store values through many backpropagation epochs, thereby providing superior results for time-series data.

- **Architecture search**: Random decreasing-size architecture search selects 6 layers to be optimal, and random activation search selects a mix of tanh and leaky ReLU. These results are robust to problem specification.
- **Hyperparameter search**: Small values for dropout and regularization hyperparameters work best, as shown at right for Litecoin. Batch sizes smaller than 72 were unstable under Adam optimization. LTC required additional regularization.
- **Loss function**: We train on balanced class weights. We fit mean-squared error loss for trading predictions and cross-entropy loss against binary response for prediction classification.
- **Transfer learning**: We iterate from a single-coin model to find regularization parameters and from the VAR training to give the LSTM duration.

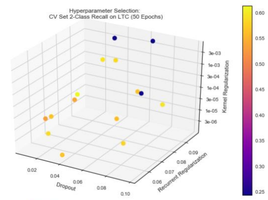

Figure: Hyperparameter search illustration.

Results

Our model **improves prediction accuracy by as much as 20% on both classes** across coins, while our trading strategy is subject to volatility that diminishes returns. We plot the ROC and tabulate accuracy on the cross-validation and test sets.

Although we obtain very good AUC and accuracy on the cross-validation set, these results do not fully transfer to the test set. While the loss functions decrease in tandem, indicating appropriate regularization, some variance is unavoidable as we have sampled from different durations to avoid market regime bias: the cross-validation set comes from a distribution which is closer in time to the training set than the test set.

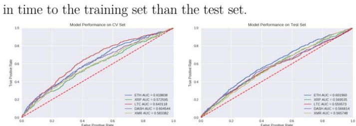

Figure: ROC curves on cross-validation and test sets for full currency set under slightly relaxed regularization.

Our best AUC is for ETH - achieving 60% on the test set - which is the most heavily capitalized coin during this period, indicating data sparsity may be contributing to the variance in our results. Additionally, it was easiest to achieve high dual-class accuracy on Litecoin, which has only 11% volatility over the study period (other coins have volatility ranging from 16% to 20%), indicating volatility may be encumbering our results as well.

	Train Set Accuracy		CV Set Accuracy		Test Set Accuracy		Correlation	
Coin	Class 0	Class 1	Class 0	Class 1	Class 0	Class 1	CV Set	Test Set
ETH	0.73	0.45	0.63	0.52	0.59	0.55	16%	15%
XRP	0.60	0.74	0.55	0.54	0.40	0.66	10%	9%
LTC	0.65	0.85	0.57	0.63	0.54	0.55	19%	10%
DASH	0.58	0.77	0.52	0.62	0.52	0.57	16%	11%
XMR	0.59	0.70	0.56	0.51	0.50	0.59	10%	9%

Figure: Accuracy (at cross-validated threshold) and predictive correlation.

By a common benchmark for trading strategies (1% correlation), our prediction obtains a **high correlation** with the realized return for all the coins, and especially ETH, DASH, and LTC. This indicates a **highly tradeable strategy** by industry standards.

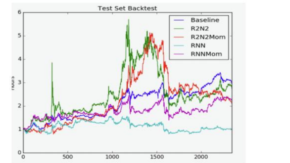

Figure: Returns accruing to trading strategy implementation. Our model outperforms the market baseline until a correction during the test set period.

Discussion

By our correlation metrics, we have shown that it is possible to build a strategy which generates **excess returns** in a portfolio of cryptocurrencies. This demonstrates inefficiencies in the cryptocurrency market, in line with the "no arbitrage principle" which states that no such excess returns exist otherwise.

It appears that high volatility and low liquidity negatively affect our classification accuracy and trading strategy results, in contrast to our expectation that our algorithm could exploit structure in currency volatility to boost accuracy. In particular, our results evince complex interplay between **mean-reversion effects** from the previous period's returns and **momentum effects that we hypothesize predominate under certain information regimes**, which we believe to induce observed excess volatility under information asymmetry. We accordingly suggest the following directions for future work:

- **Adjusting investment frequency**: There is a close link between prediction accuracy and currency volatility. Considering different timeframes (e.g. longer-term horizons) may assist in decoupling these two.
- **News data**: Incorporating news sentiment data into the RNN implementation may build resilience to volatility.
- **Economic data**: Including macroeconomic features in the VARMAX may boost the model's ability to learn seasonality.

References

[1] Martin Längkvist, Lars Karlsson, and Amy Loutfi.
A review of unsupervised feature learning and deep learning for time-series modeling.
Pattern Recognition Letters, 42, 2014.

[2] J. B. Heaton, N. G. Polson, and J. H. Witte.
Deep learning in finance.
2017.

[3] Hardik Goel, Igor Melnyk, and Arindam Banerjee.
R2n2: Residual recurrent neural networks for multivariate time series forecasting.
2017.

[4] StatsModels Documentation.