Deep Residual Learning for Portfolio Optimization: With Attention and Switching Modules

Jeff Wang, Ph.D.

Prepared for NYU FRE Seminar.

March 7th, 2019

Overview

- Study model driven portfolio management strategies
  - Construct long/short portfolio from dataset of approx. 2000 individual stocks.
  - Standard momentum and reversal predictors/features from Jagadeesh and Titman (1993), and Takeuchi and Lee (2013).
  - Probability of next month's normalized return higher/lower than median value.

- Attention Enhanced Residual Network
  - Optimize the magnitude of non-linearity in the model.
  - Strike a balance between linear and complex non-linear models.
  - Proposed network can control over-fitting.
  - Evaluate portfolio performance against linear model and complex non-linear ANN.

- Deep Residual Switching Network
  - Switching module automatically sense changes in stock market conditions.
  - Proposed network switch between market anomalies of momentum and reversal.
  - Examine dynamic behavior of switching module as market conditions change.
  - Evaluate portfolio performance against Attention Enhanced ResNet.

Part One: Attention Enhanced Residual Network

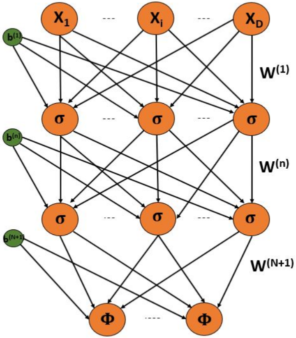


Figure 1: Fully connected hidden layer representation of multi-layer feedforward network.

Given input vector X, let $n \in \{1, 2, ..., N\}$, $i, j \in \{1, 2, 3, ..., D\}$, and $f^{(0)}(X) = X$.

- Pre-activation at hidden layer n, $z^{(n)}(X)_i = \sum_j W_{i,j}^{(n)} \cdot f^{(n-1)}(X)_j + b_i^{(n)}$
- Equivalently in Matrix Form, $z^{(n)}(X) = W^{(n)} \cdot f^{(n-1)}(X) + b^{(n)}$
- Activation at hidden layer n, $f^{(n)}(X) = \sigma(z^{(n)}(X)) = \sigma(W^{(n)} \cdot f^{(n-1)}(X) + b^{(n)})$
- Output layer $n = N + 1$, $\mathbf{F}(X) = f^{(N+1)}(X) = \Phi(z^{(N+1)}(X))$
- $\Phi(z^{(N+1)}(X)) = \left[\frac{exp(z_1^{(N+1)})}{\sum_c exp(z_c^{(N+1)})}, ..., \frac{exp(z_c^{(N+1)})}{\sum_c exp(z_c^{(N+1)})} \right]^{\mathsf{T}}$
- $\mathbf{F}(X)_c = p(y = c | X; \Theta)$, $\Theta = \{W_{i,j}^{(n)}, b_i^{(n)}\}$

Universal Approximators

Multilayer Network with ReLu Activation Function

- "Multilayer feedforward network can approximate any continuous function arbitrarily well if and only if the network's countinuous activation function is not polynomial."
- ReLu: Unbounded activation function in the form $\sigma(x) = max(0, x)$.

Definition

A set F of functions in $L^{\infty}_{loc}(R^n)$ is dense in $C(R^n)$ if for every function $g \in C(R^n)$ and for every compact set $K \subset R^n$, there exists a sequence of functions $f_j \in F$ such that

$$\lim_{f \to \infty} ||g - f_j||_{L^{\infty}(K)} = 0.$$

Theorem

(Leshno et al., 1993) Let $\sigma \in M$, where M denotes the set of functions which are in $L^{\infty}_{loc}(\Omega)$.

$$\Sigma_n = span\{\sigma(w \cdot x + b) : w \in R^n, b \in R\}$$

Then Σ_n is dense in $C(R^n)$ if and only if σ is not an algebraic polynomial (a.e.).

ANN and Over-fitting

Deep learning applied to financial data.

- Artificial Neural Network (ANN) can approximate non-linear continuous functions arbitrarily well.
- Financial markets offer non-linear relationships.
- Financial datasets are large, and ANN thrives with big datasets.

When the ANN goes deeper.

- Hidden layers mixes information from input vectors.
- Information from input data get saturated.
- Hidden units fit noises in financial data.

May reduce over-fitting with weight regularization and dropout.

- Quite difficult to control, especially for very deep networks.

Over-fitting and Generalization Power

- Generalization error decomposes into bias and variance.
- Variance: does model vary for another training dataset.
- Bias: closeness of average model to the true model F^*.

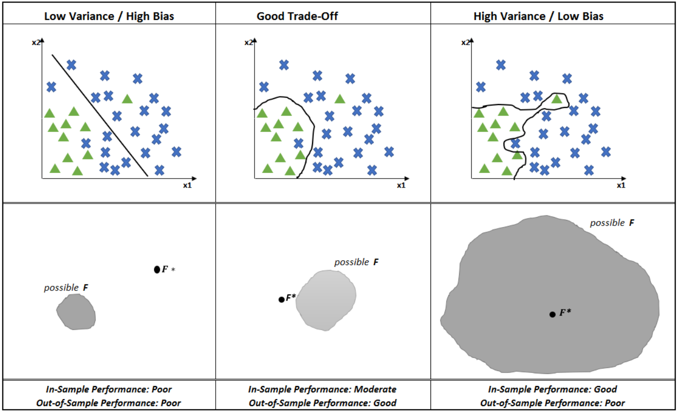

Low Variance / High Bias
Good Trade-Off
High Variance / Low Bias
x2
x1
x2
x1
x2
x1
F *
possible F
possible F
F*
possible F
F*
In-Sample Performance: Poor
Out-of-Sample Performance: Poor
In-Sample Performance: Moderate
Out-of-Sample Performance: Good
In-Sample Performance: Good
Out-of-Sample Performance: Poor

Figure 2: Bias Variance Trade-Off.

Residual Learning: Referenced Mapping

- Network architecture that references mapping.
- Unreferenced Mapping of ANN:
  - $\mathbf{Y} = \mathbf{F}(\mathbf{X}, \Theta)$
  - Underlying mapping fit by a few stacked layers.
- Referenced Residual Mapping (He et al., 2016):
  - $R(\mathbf{X}, \Theta) = \mathbf{F}(\mathbf{X}, \Theta) - \mathbf{X}$
  - $\mathbf{Y} = R(\mathbf{X}, \Theta) + \mathbf{X}$

Residual Block

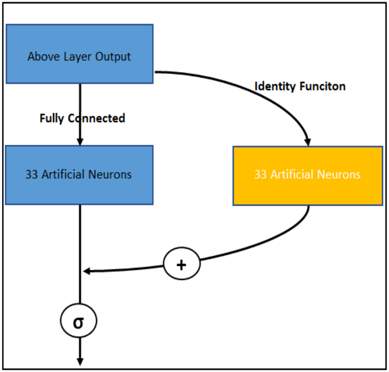


Figure 3: Fully connected hidden layer representation of multi-layer feedforward network.

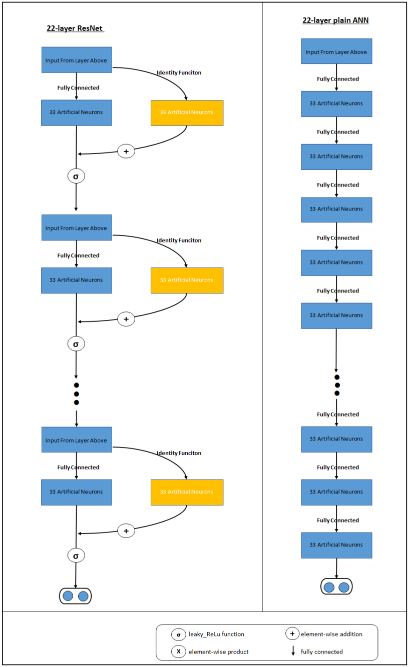
- Let $n \in \{1, 2, ..., N\}$, $i, j \in \{1, 2, 3, ..., D\}$, $f^{(0)}(X) = X$
- $z^{(n)}(X) = W^{(n)} \cdot f^{(n-1)}(X) + b^{(n)}$
- $f^{(n)}(X) = \sigma(z^{(n)}(X))$
- $z^{(n+1)}(X) = W^{(n+1)} \cdot f^{(n)}(X) + b^{(n+1)}$
- $z^{(n+1)}(X) + f^{(n-1)}(X)$
- $f^{(n+1)}(X) = \sigma(z^{(n+1)}(X) + f^{(n-1)}(X))$
- $f^{(n+1)}(X) = \sigma(W^{(n+1)} \cdot f^{(n)}(X) + b^{(n+1)} + f^{(n-1)}(X))$

In the deeper layers of residual learning system, with regularization weight decay, $W^{(n+1)} \rightarrow 0$ and $b^{(n+1)} \rightarrow 0$, and with ReLU activation function σ, we have,

- $f^{(n+1)}(X) \rightarrow \sigma(f^{(n-1)}(X))$
- $f^{(n+1)}(X) \rightarrow f^{(n-1)}(X)$

Residual Learning: Referenced Mapping

- Residual Block
  - Identity function is easy for residual blocks to learn.
  - Improves performance with each additional residual block.
  - If it cannot improve performance, simply transform via identity function.
  - Preserves structure of input features.
- Concept behind residual learning is cross-fertilizing and hopeful for algorithmic portfolio management.
- He et al., 2016. Deep residual learning for image recognition.



22-layer ResNet

Input From Layer Above

Identity Funciton

Fully Connected

33 Artificial Neurons

33 Artificial Neurons

+

σ

Input From Layer Above

Identity Funciton

Fully Connected

33 Artificial Neurons

33 Artificial Neurons

+

σ

Input From Layer Above

Identity Funciton

Fully Connected

33 Artificial Neurons

33 Artificial Neurons

+

σ

22-layer plain ANN

Input From Layer Above

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

σ leaky_ReLu function

+ element-wise addition

X element-wise product

↓ fully connected

Attention Module

- Attention Module
  - Naturally extend to residual block to guide feature learning.
  - Estimate soft weights learned from inputs of residual block.
  - Enhances feature representations at selected focal points.
  - Attention enhanced features improve predictive properties of the proposed network.
- Residual Mapping:
  - $\mathrm{R(X,\Theta)} = \mathbf{F}(\mathbf{X}, \Theta) - \mathbf{X}$
  - $\mathrm{Y} = \mathrm{R(X,\Theta)} + \mathbf{X}$
- Attention Enhanced Residual Mapping:
  - $\mathrm{Y} = (\ \mathrm{R(X,\Theta)} + \mathbf{X}) \cdot M(\mathbf{X}, \Theta)$
  - $\mathrm{Y} = (\ \mathrm{R(X,\Theta)} + W_s \cdot \mathbf{X}) \cdot M(\mathbf{X}, \Theta)$

Attention Enhanced Residual Block

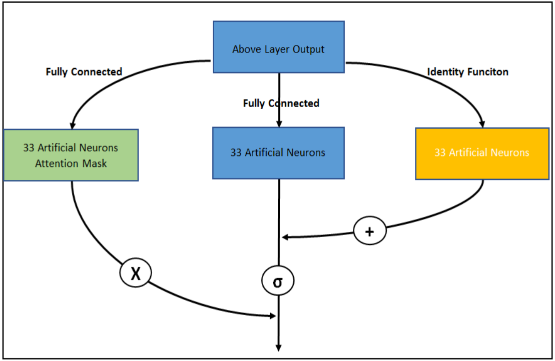
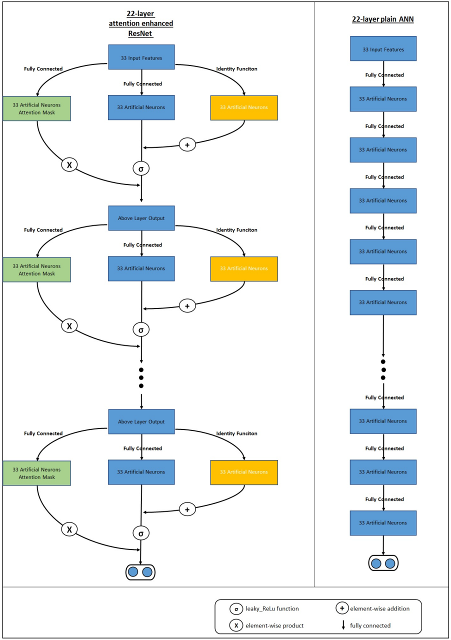


Figure 5: Representation of Attention Enhanced Residual Block, "$+$" denotes element-wise addition, σ denotes leaky-relu activation function, and "X" denotes element-wise product. The short circuit occurs before σ activation, and attention mask is applied after σ activation.

Attention Enhanced Residual Block

- $z^{a,(n)}(X) = W^{a,(n)} \cdot f^{(n-1)}(X) + b^{a,(n+1)}$
- $f^{a,(n)}(X) = \sigma(z^{a,(n)}(X))$
- $z^{a,(n+1)}(X) = W^{a,(n+1)} \cdot f^{a,(n)}(X) + b^{a,(n+1)}$
- $f^{a,(n+1)}(X) = \Phi(z^{a,(n+1)}(X))$

  where,
- $\Phi(z^{a,(n+1)}(X)) = \left[\frac{exp(z_1^{a,(n+1)})}{\sum_c exp(z_c^{a,(n+1)})}, \ldots, \frac{exp(z_c^{a,(n+1)})}{\sum_c exp(z_c^{a,(n+1)})}\right]^{\mathsf{T}}$
- $f^{(n+1)}(X) = [\sigma(z^{(n+1)}(X) + f^{(n-1)}(X))] \cdot [\Phi(z^{a,(n+1)}(X))]$



22-layer attention enhanced ResNet

33 Input Features

Fully Connected

Fully Connected

Identity Funciton

33 Artificial Neurons Attention Mask

33 Artificial Neurons

33 Artificial Neurons

+

X

σ

Above Layer Output

Fully Connected

Fully Connected

Identity Funciton

33 Artificial Neurons Attention Mask

33 Artificial Neurons

33 Artificial Neurons

+

X

σ

Above Layer Output

Fully Connected

Fully Connected

Identity Funciton

33 Artificial Neurons Attention Mask

33 Artificial Neurons

33 Artificial Neurons

+

X

σ

22-layer plain ANN

33 Input Features

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

Fully Connected

33 Artificial Neurons

σ leaky_ReLu function

+ element-wise addition

X element-wise product

↓ fully connected

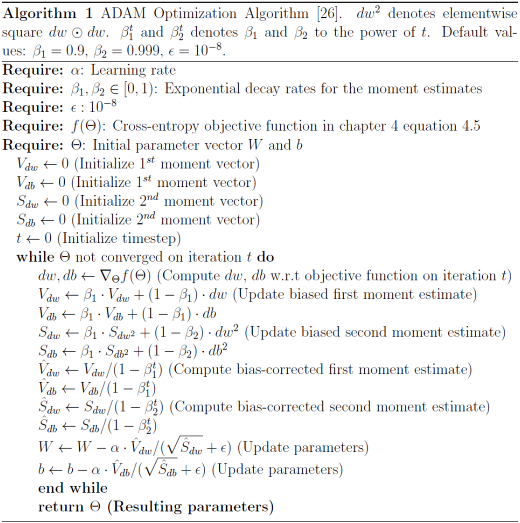
Objective Function

- Objective function minimizes the error between the estimated conditional probability and the correct target label is formulated as the following cross-entropy loss with weight regularization:
- $\underset{\Theta}{\operatorname{argmin}} \ \frac{-1}{m}\sum_m y^{(m)} \cdot log\mathbf{F}(x^{(m)};\Theta) + (1 - y^{(m)}) \cdot log(1 - \mathbf{F}(x^{(m)};\Theta)) + \lambda\sum_n ||\Theta||_F^2$
- $\Theta = \{W_{i,j}^{(n)}, b_i^{(n)}\}$; $||\cdot||_F$ is Frobenius Norm.
- Cross-entropy loss speeds up convergence when trained with gradient descent algorithm.
- Cross-entropy loss function also has the nice property that imposes a heavy penalty if $p(y = 1|X;\Theta) = 0$ when the true target label is y=1, and vice versa.

Optimization Algorithm

- Adaptive Moment (ADAM) algo combines Momentum and RMS prop.
- The ADAM algorithm have been shown to work well across a wide range of deep learning architectures.
- Cost contours: ADAM damps out oscillations in gradients that prevents the use of large learning rate.
  - Momentum: speed ups training in horizontal direction.
  - RMS Prop: Slow down learning in vertical direction.
- ADAM is appropriate for noisy financial data.
- Kingma and Ba., 2015. ADAM: A Method For Stochastic Optimization.

ADAM

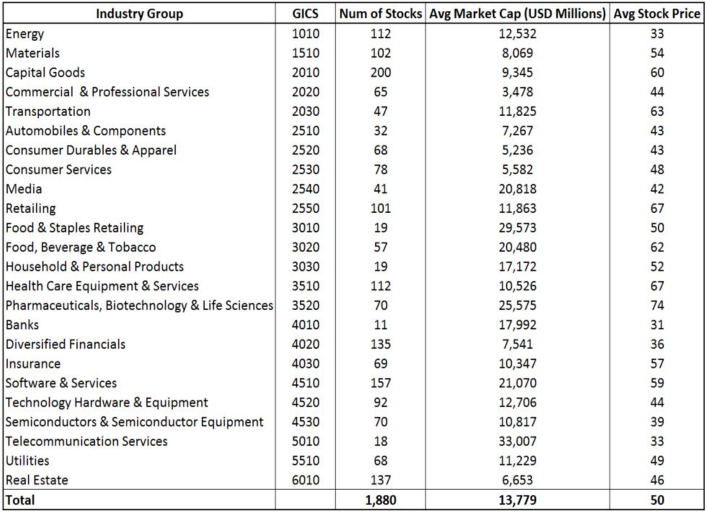


Algorithm 1 ADAM Optimization Algorithm [26]. dw^2 denotes elementwise square $dw \odot dw$. β_1^t and β_2^t denotes β_1 and β_2 to the power of t. Default values: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.

```
Require: $\alpha$: Learning rate
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $\epsilon : 10^{-8}$
Require: $f(\Theta)$: Cross-entropy objective function in chapter 4 equation 4.5
Require: $\Theta$: Initial parameter vector $W$ and $b$
  $V_{dw} \leftarrow 0$ (Initialize $1^{st}$ moment vector)
  $V_{db} \leftarrow 0$ (Initialize $1^{st}$ moment vector)
  $S_{dw} \leftarrow 0$ (Initialize $2^{nd}$ moment vector)
  $S_{db} \leftarrow 0$ (Initialize $2^{nd}$ moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while $\Theta$ not converged on iteration $t$ do
      $dw, db \leftarrow \nabla_\Theta f(\Theta)$ (Compute $dw$, $db$ w.r.t objective function on iteration $t$)
      $V_{dw} \leftarrow \beta_1 \cdot V_{dw} + (1 - \beta_1) \cdot dw$ (Update biased first moment estimate)
      $V_{db} \leftarrow \beta_1 \cdot V_{db} + (1 - \beta_1) \cdot db$
      $S_{dw} \leftarrow \beta_1 \cdot S_{dw^2} + (1 - \beta_2) \cdot dw^2$ (Update biased second moment estimate)
      $S_{db} \leftarrow \beta_1 \cdot S_{db^2} + (1 - \beta_2) \cdot db^2$
      $\hat{V}_{dw} \leftarrow V_{dw}/(1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
      $\hat{V}_{db} \leftarrow V_{db}/(1 - \beta_1^t)$
      $\hat{S}_{dw} \leftarrow S_{dw}/(1 - \beta_2^t)$ (Compute bias-corrected second moment estimate)
      $\hat{S}_{db} \leftarrow S_{db}/(1 - \beta_2^t)$
      $W \leftarrow W - \alpha \cdot \hat{V}_{dw}/(\sqrt{\hat{S}_{dw}} + \epsilon)$ (Update parameters)
      $b \leftarrow b - \alpha \cdot \hat{V}_{db}/(\sqrt{\hat{S}_{db}} + \epsilon)$ (Update parameters)
      end while
      return $\Theta$ (Resulting parameters)
```

Experiment Setting

- Model Input
  - 33 features in total. 20 normalized past daily returns, 12 normalized monthly returns for month $t-2$ through $t-13$, and an indicator variable for the month of January.
- Target Output
  - Label individual stocks with normalized monthly return above the median as 1, and below the median as 0.
- Strategy
  - Over the broad universe of US equities (approx. 2000 tickers), estimate the probability of each stock's next month's normalized return being higher or lower than median.
  - Rank estimated probabilities for all stocks in the trading universe (or by industry groups), then construct long/short portfolio of stocks with estimated probability in the top/bottom decile.
  - Long signal: $p_i > p^*$, p^*: threshold for the top decile.
  - Short signal: $p_i < p^{**}$, p^{**}: threshold for the bottom decile.



Industry Group	GICS	Num of Stocks	Avg Market Cap (USD Millions)	Avg Stock Price
Energy	1010	112	12,532	33
Materials	1510	102	8,069	54
Capital Goods	2010	200	9,345	60
Commercial & Professional Services	2020	65	3,478	44
Transportation	2030	47	11,825	63
Automobiles & Components	2510	32	7,267	43
Consumer Durables & Apparel	2520	68	5,236	43
Consumer Services	2530	78	5,582	48
Media	2540	41	20,818	42
Retailing	2550	101	11,863	67
Food & Staples Retailing	3010	19	29,573	50
Food, Beverage & Tobacco	3020	57	20,480	62
Household & Personal Products	3030	19	17,172	52
Health Care Equipment & Services	3510	112	10,526	67
Pharmaceuticals, Biotechnology & Life Sciences	3520	70	25,575	74
Banks	4010	11	17,992	31
Diversified Financials	4020	135	7,541	36
Insurance	4030	69	10,347	57
Software & Services	4510	157	21,070	59
Technology Hardware & Equipment	4520	92	12,706	44
Semiconductors & Semiconductor Equipment	4530	70	10,817	39
Telecommunication Services	5010	18	33,007	33
Utilities	5510	68	11,229	49
Real Estate	6010	137	6,653	46
Total		**1,880**	**13,779**	**50**

Table 1: Trading Universe Categorized by GICS Industry Group as of January 3, 2017.

Experiment Setting

- Holding Period: 20 trading days (one month).
- Cost Coefficient
  - Assume trades executed at the closing price of the day.
  - Assume 5 basis points transaction cost per trade.
  - To ensure liquidity, sampled stocks that traded above 5 USD.
- Profit Functions (Yearly Return)
  - $\mathrm{R}^{(1)} = \sum_{m=1}^{12} \left[0.5 \sum_{l}^{L} R_{l,m} - 0.5 \sum_{s}^{S} R_{s,m} - 2.c.(L_m + S_m) \right]$
  - $\mathrm{R}^{(2)} = \sum_{m=1}^{12} \cdot \sum_{g=1}^{24} \left[0.5 \sum_{l}^{L} R_{l,m} - 0.5 \sum_{s}^{S} R_{s,m} - 2.c.(L_m + S_m) \right]$
- Back-Test Comparison
  - 22-layer Attention ResNet.
  - 22-layer plain ANN.
  - Logistic Regression.

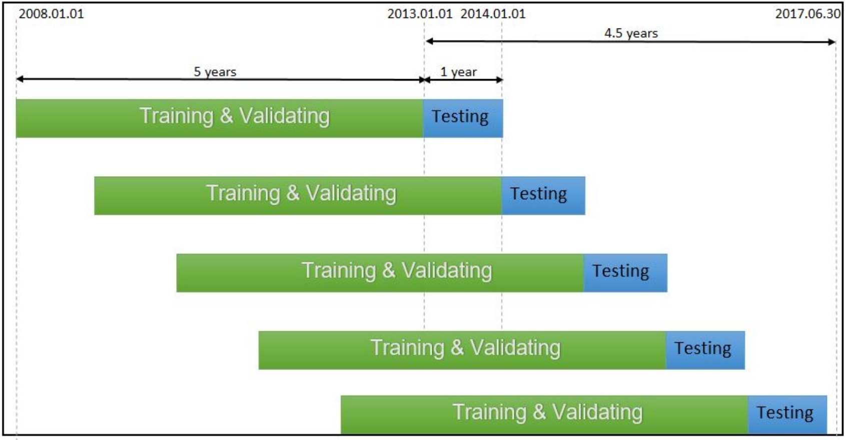


Figure 7: Rolling dataset arrangement for training, validating, and testing from 2008 to 2017.

Training Procedure

- Network trained using batch data: $\{(x_m, y_m) | x_m \in X, y_m \in Y\}_{m=1,2,\ldots,b}$. b is the batch size set as 512.
- Implemented batch normalization on every hidden layer except the output layer.
- Added random Gaussian noise $N(0, 0.1)$ to the input tensor for noise resistance and robustness.
- Initialized the network at random, the learning rate was set at 0.0001 with 0.995 exponential decay.
- Trained the model using ADAM optimization algorithm for approximately 100k steps (20 epochs) until convergence and validated our model every 10k steps to obtain optimal hyper-parameters.
- Codes are written with TensorFlow.

In Sample Results - Strategy 1

- Rank estimated probabilities for stocks in the trading universe.

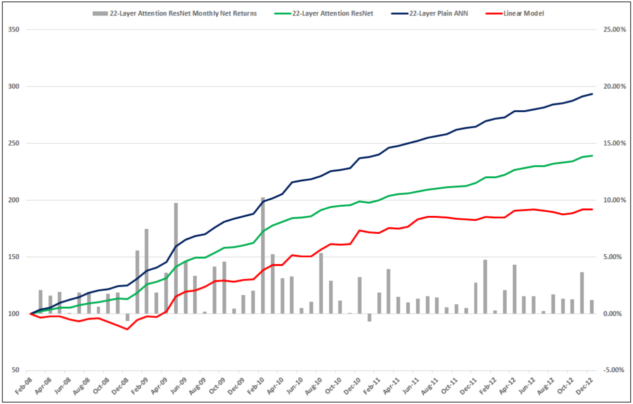


Figure 8: In-sample histoical PNL comparison for 22-layer attention ResNet, 22-layer ANN, and logistic regression.

Out of Sample Results - Strategy 1

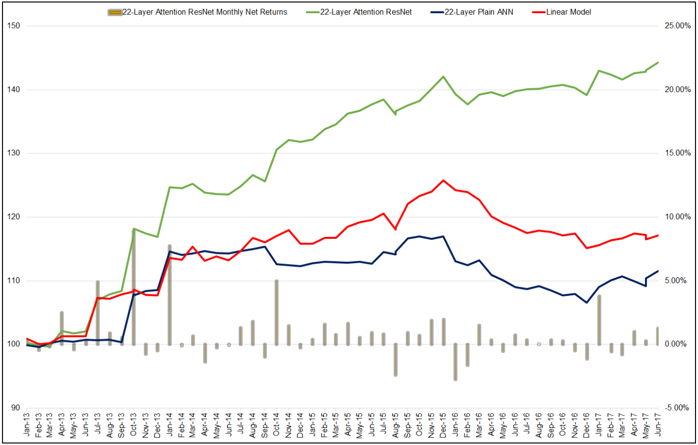


Figure 9: Out-of-sample histoical PNL comparison for 22-layer attention ResNet, 22-layer ANN, and logistic regression.

In Sample Results - Strategy 2

- Rank estimated probabilities for stocks by industry groups.

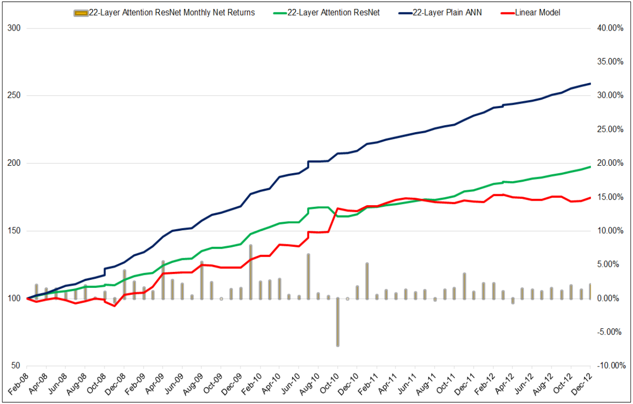
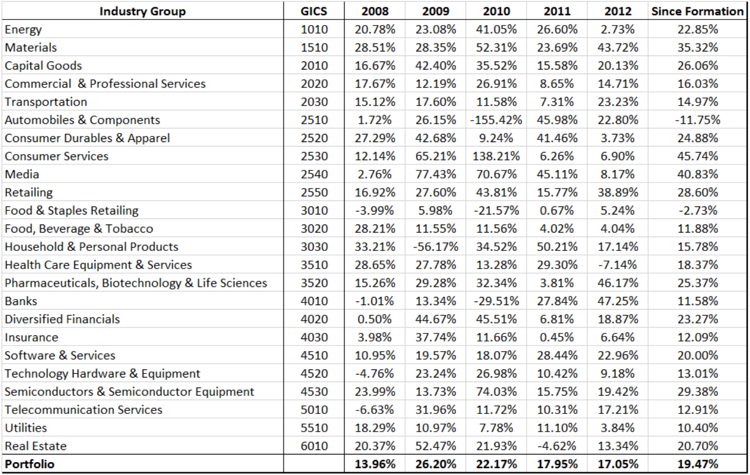


Figure 10: Industry diversified strategy's in-sample histoical PNL comparison for 22-layer attention ResNet, 22-layer ANN, and logistic regression.



Industry Group	GICS	2008	2009	2010	2011	2012	Since Formation
Energy	1010	20.78%	23.08%	41.05%	26.60%	2.73%	22.85%
Materials	1510	28.51%	28.35%	52.31%	23.69%	43.72%	35.32%
Capital Goods	2010	16.67%	42.40%	35.52%	15.58%	20.13%	26.06%
Commercial & Professional Services	2020	17.67%	12.19%	26.91%	8.65%	14.71%	16.03%
Transportation	2030	15.12%	17.60%	11.58%	7.31%	23.23%	14.97%
Automobiles & Components	2510	1.72%	26.15%	-155.42%	45.98%	22.80%	-11.75%
Consumer Durables & Apparel	2520	27.29%	42.68%	9.24%	41.46%	3.73%	24.88%
Consumer Services	2530	12.14%	65.21%	138.21%	6.26%	6.90%	45.74%
Media	2540	2.76%	77.43%	70.67%	45.11%	8.17%	40.83%
Retailing	2550	16.92%	27.60%	43.81%	15.77%	38.89%	28.60%
Food & Staples Retailing	3010	-3.99%	5.98%	-21.57%	0.67%	5.24%	-2.73%
Food, Beverage & Tobacco	3020	28.21%	11.55%	11.56%	4.02%	4.04%	11.88%
Household & Personal Products	3030	33.21%	-56.17%	34.52%	50.21%	17.14%	15.78%
Health Care Equipment & Services	3510	28.65%	27.78%	13.28%	29.30%	-7.14%	18.37%
Pharmaceuticals, Biotechnology & Life Sciences	3520	15.26%	29.28%	32.34%	3.81%	46.17%	25.37%
Banks	4010	-1.01%	13.34%	-29.51%	27.84%	47.25%	11.58%
Diversified Financials	4020	0.50%	44.67%	45.51%	6.81%	18.87%	23.27%
Insurance	4030	3.98%	37.74%	11.66%	0.45%	6.64%	12.09%
Software & Services	4510	10.95%	19.57%	18.07%	28.44%	22.96%	20.00%
Technology Hardware & Equipment	4520	-4.76%	23.24%	26.98%	10.42%	9.18%	13.01%
Semiconductors & Semiconductor Equipment	4530	23.99%	13.73%	74.03%	15.75%	19.42%	29.38%
Telecommunication Services	5010	-6.63%	31.96%	11.72%	10.31%	17.21%	12.91%
Utilities	5510	18.29%	10.97%	7.78%	11.10%	3.84%	10.40%
Real Estate	6010	20.37%	52.47%	21.93%	-4.62%	13.34%	20.70%
Portfolio		**13.96%**	**26.20%**	**22.17%**	**17.95%**	**17.05%**	**19.47%**

Table 2: Breakdown of Attention ResNet's annualized return for in-sample period. Trading signals sorted by GICS industry groups.

Out of Sample Results - Strategy 2

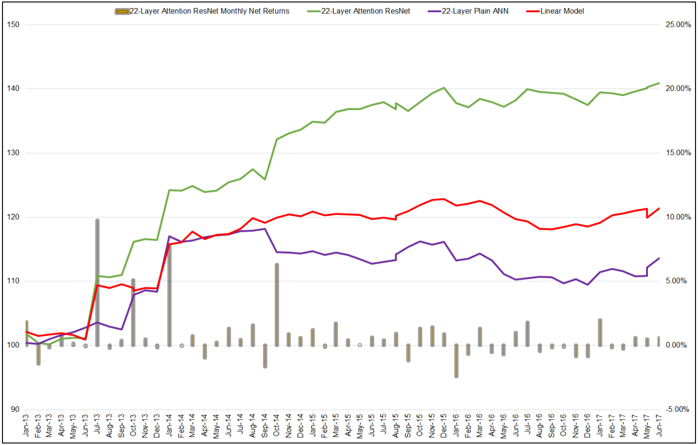
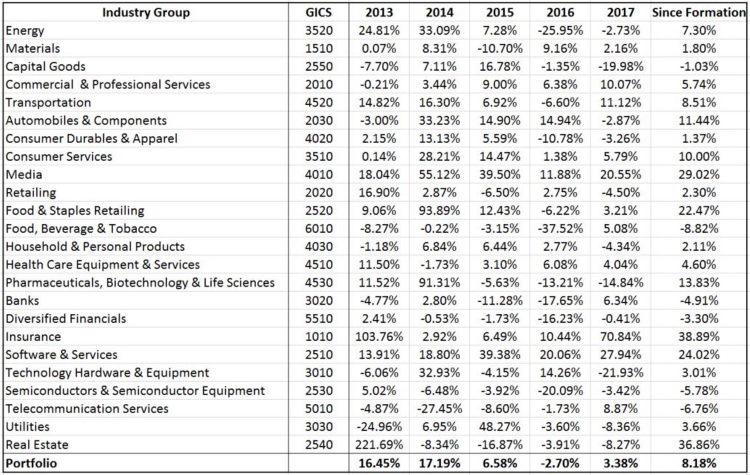
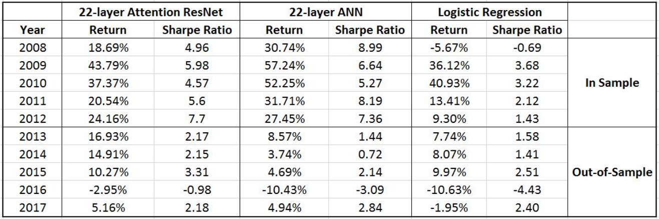


Figure 11: Industry diversified strategy's out-of-sample histoical PNL comparison for 22-layer attention ResNet, 22-layer ANN, and logistic regression.



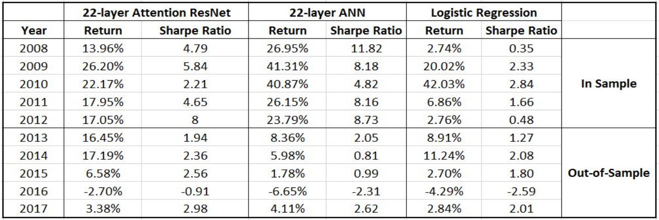
Industry Group	GICS	2013	2014	2015	2016	2017	Since Formation
Energy	3520	24.81%	33.09%	7.28%	-25.95%	-2.73%	7.30%
Materials	1510	0.07%	8.31%	-10.70%	9.16%	2.16%	1.80%
Capital Goods	2550	-7.70%	7.11%	16.78%	-1.35%	-19.98%	-1.03%
Commercial & Professional Services	2010	-0.21%	3.44%	9.00%	6.38%	10.07%	5.74%
Transportation	4520	14.82%	16.30%	6.92%	-6.60%	11.12%	8.51%
Automobiles & Components	2030	-3.00%	33.23%	14.90%	14.94%	-2.87%	11.44%
Consumer Durables & Apparel	4020	2.15%	13.13%	5.59%	-10.78%	-3.26%	1.37%
Consumer Services	3510	0.14%	28.21%	14.47%	1.38%	5.79%	10.00%
Media	4010	18.04%	55.12%	39.50%	11.88%	20.55%	29.02%
Retailing	2020	16.90%	2.87%	-6.50%	2.75%	-4.50%	2.30%
Food & Staples Retailing	2520	9.06%	93.89%	12.43%	-6.22%	3.21%	22.47%
Food, Beverage & Tobacco	6010	-8.27%	-0.22%	-3.15%	-37.52%	5.08%	-8.82%
Household & Personal Products	4030	-1.18%	6.84%	6.44%	2.77%	-4.34%	2.11%
Health Care Equipment & Services	4510	11.50%	-1.73%	3.10%	6.08%	4.04%	4.60%
Pharmaceuticals, Biotechnology & Life Sciences	4530	11.52%	91.31%	-5.63%	-13.21%	-14.84%	13.83%
Banks	3020	-4.77%	2.80%	-11.28%	-17.65%	6.34%	-4.91%
Diversified Financials	5510	2.41%	-0.53%	-1.73%	-16.23%	-0.41%	-3.30%
Insurance	1010	103.76%	2.92%	6.49%	10.44%	70.84%	38.89%
Software & Services	2510	13.91%	18.80%	39.38%	20.06%	27.94%	24.02%
Technology Hardware & Equipment	3010	-6.06%	32.93%	-4.15%	14.26%	-21.93%	3.01%
Semiconductors & Semiconductor Equipment	2530	5.02%	-6.48%	-3.92%	-20.09%	-3.42%	-5.78%
Telecommunication Services	5010	-4.87%	-27.45%	-8.60%	-1.73%	8.87%	-6.76%
Utilities	3030	-24.96%	6.95%	48.27%	-3.60%	-8.36%	3.66%
Real Estate	2540	221.69%	-8.34%	-16.87%	-3.91%	-8.27%	36.86%
Portfolio		**16.45%**	**17.19%**	**6.58%**	**-2.70%**	**3.38%**	**8.18%**

Table 3: Breakdown of Attention ResNet's out-of-sample annualized return. Trading signals sorted by GICS industry groups.



	22-layer Attention ResNet		22-layer ANN		Logistic Regression		
Year	Return	Sharpe Ratio	Return	Sharpe Ratio	Return	Sharpe Ratio	
2008	18.69%	4.96	30.74%	8.99	-5.67%	-0.69	In Sample
2009	43.79%	5.98	57.24%	6.64	36.12%	3.68	
2010	37.37%	4.57	52.25%	5.27	40.93%	3.22	
2011	20.54%	5.6	31.71%	8.19	13.41%	2.12	
2012	24.16%	7.7	27.45%	7.36	9.30%	1.43	
2013	16.93%	2.17	8.57%	1.44	7.74%	1.58	Out-of-Sample
2014	14.91%	2.15	3.74%	0.72	8.07%	1.41	
2015	10.27%	3.31	4.69%	2.14	9.97%	2.51	
2016	-2.95%	-0.98	-10.43%	-3.09	-10.63%	-4.43	
2017	5.16%	2.18	4.94%	2.84	-1.95%	2.40	

Table 4: Annualized return and Sharpe ratio for the three models. Sharpe Ratio is defined as μ-r/s, where μ, r, s are the annualized return, risk free rate and standard deviation of the PNL.



	22-layer Attention ResNet		22-layer ANN		Logistic Regression		
Year	Return	Sharpe Ratio	Return	Sharpe Ratio	Return	Sharpe Ratio	
2008	13.96%	4.79	26.95%	11.82	2.74%	0.35	In Sample
2009	26.20%	5.84	41.31%	8.18	20.02%	2.33	
2010	22.17%	2.21	40.87%	4.82	42.03%	2.84	
2011	17.95%	4.65	26.15%	8.16	6.86%	1.66	
2012	17.05%	8	23.79%	8.73	2.76%	0.48	
2013	16.45%	1.94	8.36%	2.05	8.91%	1.27	Out-of-Sample
2014	17.19%	2.36	5.98%	0.81	11.24%	2.08	
2015	6.58%	2.56	1.78%	0.99	2.70%	1.80	
2016	-2.70%	-0.91	-6.65%	-2.31	-4.29%	-2.59	
2017	3.38%	2.98	4.11%	2.62	2.84%	2.01	

Table 5: Industry diversified strategy's annualized return and Sharpe ratio for the three models. Sharpe Ratio is defined as μ-r/s, where μ, r, s are the annualized return, risk free rate and standard deviation of the PNL.

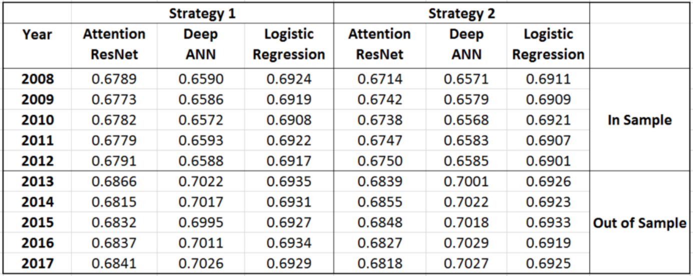
Statistical Approach: T-Test

- Conduct t-test for return differences to reach a robust conclusion.

- 54 months out-of-sample annualized return results for the three models.
- **Null hypothesis:** There is no statistically significant difference between the samples.
- For Strategy 1 (signals sorted on entire trading universe), the t-test rejects the null hypothesis at the **10 percent level**.
- For Strategy 2 (signals sorted by each GICS industry group), the t-test rejects the null hypothesis at the **10 percent level**.
- Statistical findings support that Attention ResNet has the best out-of-sample performance among the three models.

Predictive Accuracy

- In-sample predictive accuracy of the 22-layer ANN outperformed the 22-layer Attention ResNet, and both deep learning models outperformed the linear model.
- Out-of-sample predictive accuracy of the 22-layer attention ResNet **outperformed** both the 22-layer ANN and the logistic regression model.
- Predictive accuracy measured as cross entropy loss: $\frac{-1}{M}\sum_{m=1}^{M} y^{(m)} \cdot log\mathbf{F}(x^{(m)};\Theta) + (1-y^{(m)}) \cdot log(1-\mathbf{F}(x^{(m)};\Theta))$



	Strategy 1			Strategy 2			
Year	**Attention ResNet**	**Deep ANN**	**Logistic Regression**	**Attention ResNet**	**Deep ANN**	**Logistic Regression**	
2008	0.6789	0.6590	0.6924	0.6714	0.6571	0.6911	**In Sample**
2009	0.6773	0.6586	0.6919	0.6742	0.6579	0.6909	
2010	0.6782	0.6572	0.6908	0.6738	0.6568	0.6921	
2011	0.6779	0.6593	0.6922	0.6747	0.6583	0.6907	
2012	0.6791	0.6588	0.6917	0.6750	0.6585	0.6901	
2013	0.6866	0.7022	0.6935	0.6839	0.7001	0.6926	**Out of Sample**
2014	0.6815	0.7017	0.6931	0.6855	0.7022	0.6923	
2015	0.6832	0.6995	0.6927	0.6848	0.7018	0.6933	
2016	0.6837	0.7011	0.6934	0.6827	0.7029	0.6919	
2017	0.6841	0.7026	0.6929	0.6818	0.7027	0.6925	

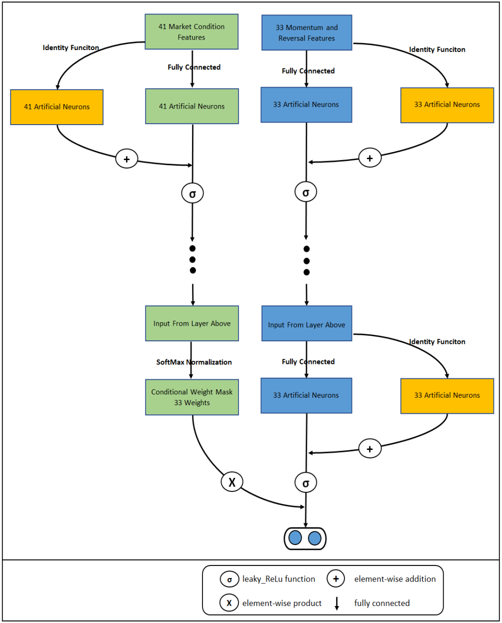
Part One Summary

- Present a novel neural network architecture for portfolio optimization.
- Attention ResNet captures the appropriate magnitude of non-linearity, and strikes a balance between linear and complex non-linear models for financial modeling.
- Attention ResNet incorporates attention mechanism to enhance financial feature learning.
- Increased network depth to tens of hidden layers, quite deep for financial modeling using DL methodologies.
- The scope of this work extends to portfolio management, nonetheless, this work is hopeful for other financial fields where over-fitting is an obstacle for DL models.

- There are various stylized features/predictors for portfolio management.
- Broadly categorized by prominent market anomalies of momentum and reversal.
- Each style is effective periodically, but rarely all the time (Voyanos and Woolley 2008).
  - Momentum/reversal is effective in a bullish/bearish market regime. Bullish/bearish regime is associated with lower/higher market variance (Cooper, Gutierrez and Hameed 2004).
- Based on this notion, it would be extremely beneficial if the DL system can figure out which style (momentum or reversal) is more effective and switch accordingly.

Deep Residual Switching Network

- Develop the residual switching network (Switching ResNet).
- Combines two separate ResNets: Switching Module and ResNet from Part 1.
- Switching Module learns market condition features and computes a conditional weight mask. This mask applied to ResNet from Part 1 via element wise product.
- The switch or weight change conditions are market conditions like squared VIX, realized volatility and variance risk premium for SP500 index.



41 Market Condition Features

33 Momentum and Reversal Features

Identity Funciton

Fully Connected

Fully Connected

Identity Funciton

41 Artificial Neurons

41 Artificial Neurons

33 Artificial Neurons

33 Artificial Neurons

+

+

σ

σ

Input From Layer Above

Input From Layer Above

SoftMax Normalization

Fully Connected

Identity Funciton

Conditional Weight Mask 33 Weights

33 Artificial Neurons

33 Artificial Neurons

+

X

σ

σ leaky_ReLu function

+ element-wise addition

X element-wise product

↓ fully connected

ResNet

- $z^{(n)}(X) = W^{(n)} \cdot f^{(n-1)}(X) + b^{(n)}$
- $f^{(n)}(X) = \sigma(z^{(n)}(X))$
- $z^{(n+1)}(X) = W^{(n+1)} \cdot f^{(n)}(X) + b^{(n+1)}$
- $z^{(n+1)}(X) + f^{(n-1)}(X)$
- $f^{(n+1)}(X) = \sigma(z^{(n+1)}(X) + f^{(n-1)}(X))$

Switching Module

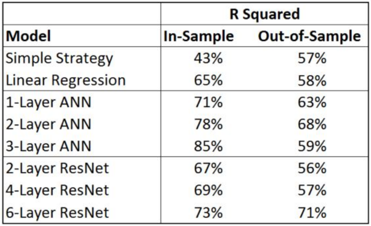
- $z^{s,(n)}(X^s) = W^{s,(n)} \cdot f^{s,(n-1)}(X^s) + b^{s,(n)}$
- $f^{s,(n)}(X^s) = \sigma(z^{s,(n)}(X^s))$
- $z^{s,(n+1)}(X^s) = W^{s,(n+1)} \cdot f^{s,(n)}(X^s) + b^{s,(n+1)}$
- $z^{s,(n+1)}(X^s) + f^{s,(n-1)}(X^s)$
- $f^{s,(n+1)}(X^s) = \sigma(z^{s,(n+1)}(X^s) + f^{s,(n-1)}(X^s))$
- Output layer $n = N + 1$,
- $f^{s,(N+1)}(X^s) = \Phi(z^{s,(N+1)}(X^s))$
- $\Phi(z^{s,(N+1)}(X^s)) = \left[\frac{exp(z_1^{s,(N+1)}(X^s))}{\sum_c exp(z_c^{s,(N+1)}(X^s))}, \ldots, \frac{exp(z_c^{s,(N+1)}(X^s))}{\sum_c exp(z_c^{s,(N+1)}(X^s))} \right]^{\mathsf{T}}$

Combined

- Output layer $n = N + 1$,
- $f^{(N+1)}(X) = \Phi\left[z^{(N+1)}(X) \cdot f^{s,(N+1)}(X^s)\right]$

Intermediate Result

- Evaluate switching module's predictive power on SP500 index RV.
- Input Features: Past values of return variance and squared VIX.
- In-Sample period 2005-2015. Out-of-Sample period 2015-Aug 2018.
- Following Bollerslev et al. 2016, out-of-sample R^2 formulated as $R^2 = 1 - \sum_{t=1}^{T}(RV^{M}_{t+22} - RV^{P}_{t+22})^2 / \sum_{t=1}^{T}(RV^{M}_{t+22} - RV^{LR}_{t})^2$.
  - RV^{LR}_{t} is long-run volatility factor over the full sample up to time t.



	R Squared	
Model	**In-Sample**	**Out-of-Sample**
Simple Strategy	43%	57%
Linear Regression	65%	58%
1-Layer ANN	71%	63%
2-Layer ANN	78%	68%
3-Layer ANN	85%	59%
2-Layer ResNet	67%	56%
4-Layer ResNet	69%	57%
6-Layer ResNet	73%	71%

Table 6: In-sample and out-of-sample R^2 scores for SP500 realized volatility prediction.

Experiment Setting

- Strategy (Same as in Part 1)
  - Over the broad universe of US equities (approx. 2000 tickers), estimate the probability of each stock's next month's normalized return being higher or lower than median.
  - Long signal: $p_i > p^*$, p^*: threshold for the top decile.
  - Short signal: $p_i < p^{**}$, p^{**}: threshold for the bottom decile.
- Target Output: Same as in Part 1.
- Model Input:
  - **Individual stock level:** 33 features in total. 20 normalized past daily returns, 12 normalized monthly returns for month $t-2$ through $t-13$, and an indicator variable for the month of January (Jagadeesh and Titman, 1993; Takeuchi and Lee, 2013).
  - **Market Conditions:** 41 features in total. 23 past daily VIX values, 12 SP500 return variance values for month $t-2$ through $t-13$, and 6 SP500 variance risk premium values for month $t-1$ through $t-6$.

Variance Risk Premium

- Carr and Wu (2016): Variance risk premium can be quantified as the difference of variance swap rate and ex post realized variance.

$$SW_{t,T} = \mathbb{E}_t^{\mathbb{P}}[m_{t,T} RV_{t,T}] = \mathbb{E}_t^{\mathbb{P}}[RV_{t,T}] + \mathrm{Cov}_t^{\mathbb{P}}(m_{t,T}, RV_{t,T}) \qquad (1)$$

- $\mathbb{E}_t^{\mathbb{P}}[RV_{t,T}]$ is the conditional mean of realized variance.
- $\mathrm{Cov}_t^{\mathbb{P}}(m_{t,T}, RV_{t,T})$ is the conditional co-variance between realized variance and normalized pricing kernel $m_{t,T}$, the negative of this term defines variance risk premium.
- We formulate the variance risk premium for SP500 return as the difference between VIX and SP500 index's one month realized variance.
  - VIX is an approximator for 30-day variance swap rate on SP500 index

Training Procedure

- Network trained using batch data: $\{(x_m, x_m^s, y_m) | x_m \in X, x_m^s \in X^s, y_m \in Y\}_{m=1,2,\dots,b}$, b is the batch size set as 512.
- Added random Gaussian noise $N(0, 0.1)$ to the input tensor for noise resistance and robustness.
- Implemented batch normalization on every hidden layer except the output layer.
- Initialized the network at random, the learning rate was set at 0.0003 with 0.995 exponential decay.
- Trained the model using ADAM optimization algorithm for approximately 120k steps (24 epochs) until convergence and validated our model every 10k steps to obtain optimal hyper-parameters.
- Codes are written with TensorFlow.

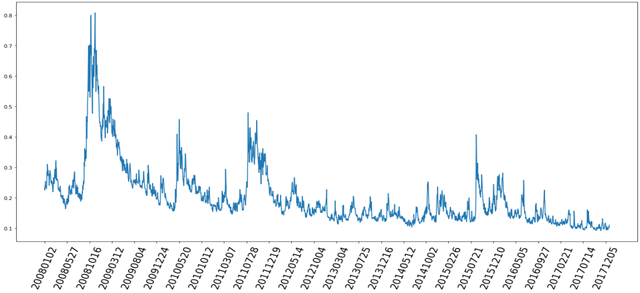
Conditional Weight Mask: Dynamic Behavior

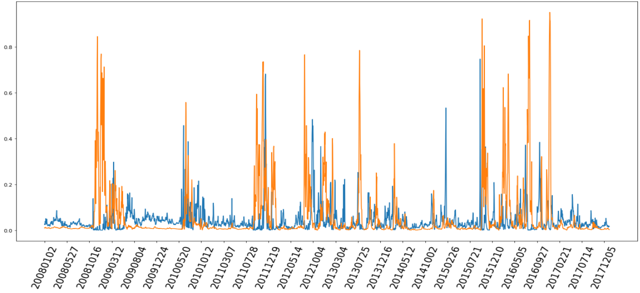
- The focus of the experiment is to evaluate the ability of the switching module to guide feature selection as market conditions change.

- For each day in the entire sample period, the switching module computes a conditional weight mask comprised of 33 weights, one for each of the 33 individual stock level feature representations.

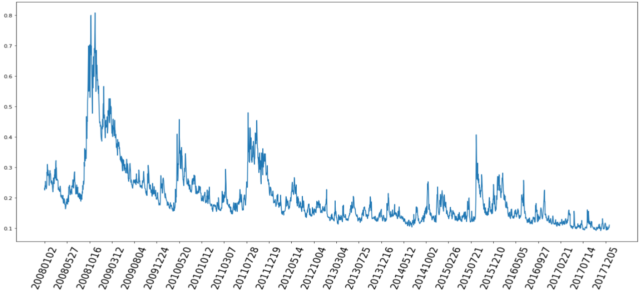
- **Observe two patterns for conditional weight mask:**
    - Weights on reversal representations jumps higher during periods of higher market volatility and are positively correlated with VIX.
    - Weights on momentum representations are lower during periods of higher market volatility and are negatively correlated with VIX.

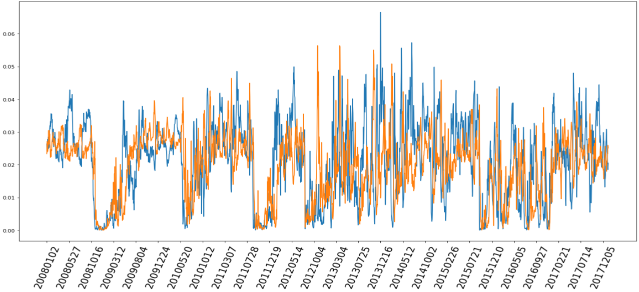
Conditional Weight Mask: Reversal

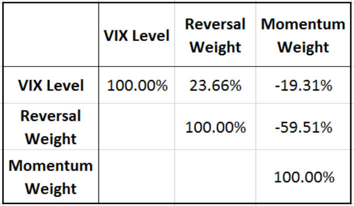




Conditional Weight Mask: Momentum

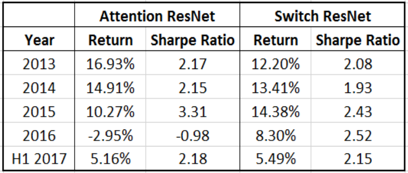






	VIX Level	Reversal Weight	Momentum Weight
VIX Level	100.00%	23.66%	-19.31%
Reversal Weight		100.00%	-59.51%
Momentum Weight			100.00%

Table 7: Correlation matrix for VIX level and conditional weights assigned to momentum latent representation of normalized past monthly return lag 3, and conditional weights assigned to reversal latent representation of normalized past daily return lag 17.



	Attention ResNet		Switch ResNet	
Year	Return	Sharpe Ratio	Return	Sharpe Ratio
2013	16.93%	2.17	12.20%	2.08
2014	14.91%	2.15	13.41%	1.93
2015	10.27%	3.31	14.38%	2.43
2016	-2.95%	-0.98	8.30%	2.52
H1 2017	5.16%	2.18	5.49%	2.15

Table 8: Table records out-of-sample annualized return and Sharpe ratio for both the Attention ResNet and the Switching ResNet.

Part Two Summary

- Presents a novel residual switching network architecture which combines two separate ResNets: a switching module that learns market conditions, and a ResNet for individual stock level features.

- Dynamic behavior of the switching module is in excellent agreement with changes in stock market conditions.
  - During periods of higher market volatility, the switching module concentrates on reversal latent representations.
  - For periods of lower market volatility, the condition weight mask switches concentration back to momentum.

- Switching module can automatically sense changes in stock market conditions and guide the proposed DL framework to switch between the momentum and reversal anomalies.

Thank You!