Predictions on Private Equity Fund's Contribution and Distribution with CNN Encoder-Decoder LSTM Model

Columbia University

Annan Chen
Curren Sameer Tipnis
Jiaqi Liu

Executive Summary

Our project is trying to model the contribution and distribution of PE funds. They are time related characters, especially when using the rate of contribution (percentage of capital commitment). Based on the work done by the previous team and discussion with the Professori and TAs, we decided to keep the LSTM in our model structure.

The previous team used a 1-layer LSTM model to predict the future contribution and distribution for Buyout(different sizes), VC and Real Estate funds. Starting from where they have built, we tried different angles to improve the model, including adding more data, trying different interpolation methods, adding macro factors, changing model structures, extending the prediction horizon and so on.

Finally, we managed to extend the model in the following ways: First is to add more macroeconomic factors, such as GDP, interest rate and CPI, etc., making the inputs of LSTM a multidimensional vector. Second, as the number of features increases in the model, we modify the architecture of LSTM by adding an autoencoder and convolutional layers before the LSTM to further capture the pattern of the multidimensional time series data. Third, we also build a model with multiple prediction horizons. For model extension part, for example, for all Buyout Fund's contribution cash flow, our CNN-LSTM multivariate model lowered the test set MSE from 0.008 to 0.001 with 100 training epochs.

For the coming groups, we recommend them trying the following starting from these aspects. First and foremost, get more data. Currently, we used the data provided by the previous group. The original data are 5-year quarterly benchmark data for the three kinds of funds mentioned above. Then implementing linear interpolation with Gaussian noise to get the daily data. The daily data is neither realistic for PE funds nor sufficient enough to build a data-hungry deep learning model. We explored Preqin and Capital IQ

databases but found nothing helpful. Secondly, we also recommend the later teams to try different interpolation methods, another path we tried hard but failed. We found CGANs might be a way to realize this but we failed to build a CGANs model for financial time series data from scratch.

In general, our CNN Encoder-decoder LSTM could fit the training data well and capture the pattern and trend for the testing set. However, this is based on the assumption that the input data could correctly demonstrate the Private Equity Funds' cash flow. Therefore, we suggest the next group to focus on how to get high quality input data, including getting more original data as well as finding better ways to interpolate.

For the project collaboration, Annan focused on preprocessing the data and modifying the architecture of the model, changing the hyperparameters and comparing the result. Curren focused on working with the models and experimenting with different time horizon predictions than the previous group by changing the training structure of the model. Jiaqi explored databases trying to get data of higher quality, helped with collecting macro factors data, explored ways of feature engineering and data interpolation.

Table of Contents

Introduction

Private Equity, a fast growing part in the past years in the financial sector, attracts investors attention as they give alternatives investment for people who would like to take illiquidity risk with higher rewards in return. PE firms mainly focus on primary markets. They invest in companies in their early stages to own a share of such company and sell it later to other investors or to the public. This business model is recognized as a very profitable one in the industry. However, from an investor's perspective, it is hard to predict the cash flows of PE companies as well as returns for investors due to illiquid nature of assets and lack of accessible data.

Research papers were published regarding the cash flow prediction for PE funds. Based on the concepts and assumptions brought up by some classic papers in this area, the previous team built LSTM models to predict the cash flows of such variables for t + n , where n can be any time within a year.

In our work, we kept the LSTM structure in our models since the memory capability of the LSTM works perfect in this situation. The cell state is saving some of the cash flows that were observed in the previous quarters, the forget layer decides which of those cash flows to forget, the middle layers train the data and pass it through the cell state (adding new data or updating previous data that was stored) and the output layer performs the double filter to obtain the predictions.

We made improvements to the work done by the previous team and gained better results by adding macro factors, changing the model architectures, such as encoding-decoding layers, convolutional layers and CNN layers, tuning hyperparameters, expanding the prediction horizons and so on. Details are discussed as follows.

Data

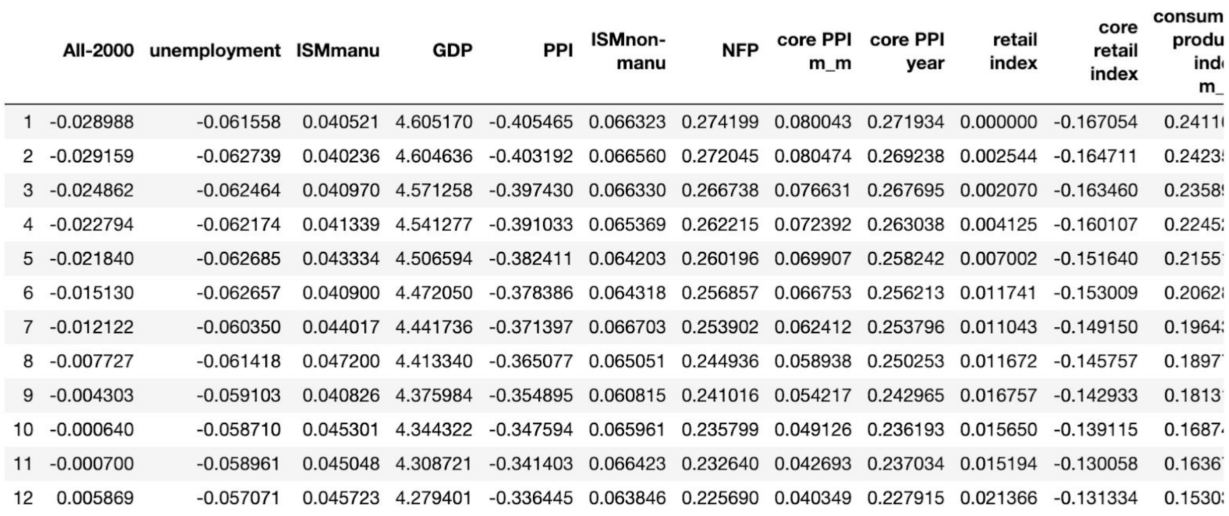
The data part includes the description of the data we used, the process procedure and other experiments and exploration we did to get more data.

Data Description and Processing

We used the 5-year quarterly benchmarks data for the 3 types of funds downloaded from Preqin as source data. Based on the assumption used in many papers, we also assume that the call-up and distribution of PE funds take lognormal distribution. Therefore, instead of replicating the call-up and distribution themselves, we computed the change of the data after log transformation followed by linear interpolation and adding Gaussian noise to expand the log difference to daily data.

In addition, we selected 20 macro factors and got their historical data of the same time period.The look-ahead bias is considered when we match the funds data with the macro factors data. To be specific, take the unemployment rate as an example, when we match the funds data for May 2010, we choose the unemployment rate that was available at the time. We also checked the distribution for each macro factor and chose different processings for different distributions, for example, calculated difference for factors taking normal distribution and log difference for those taking lognormal distribution. However, the different process mode makes the feature data differ in scale with the dependent variable thus causing an exploding gradient problem. We tried different ways and this problem can’t be solved. At last, we decided to apply the same log difference method followed by linear interpolation for all the factors.

The following is a sample of our data after processing.



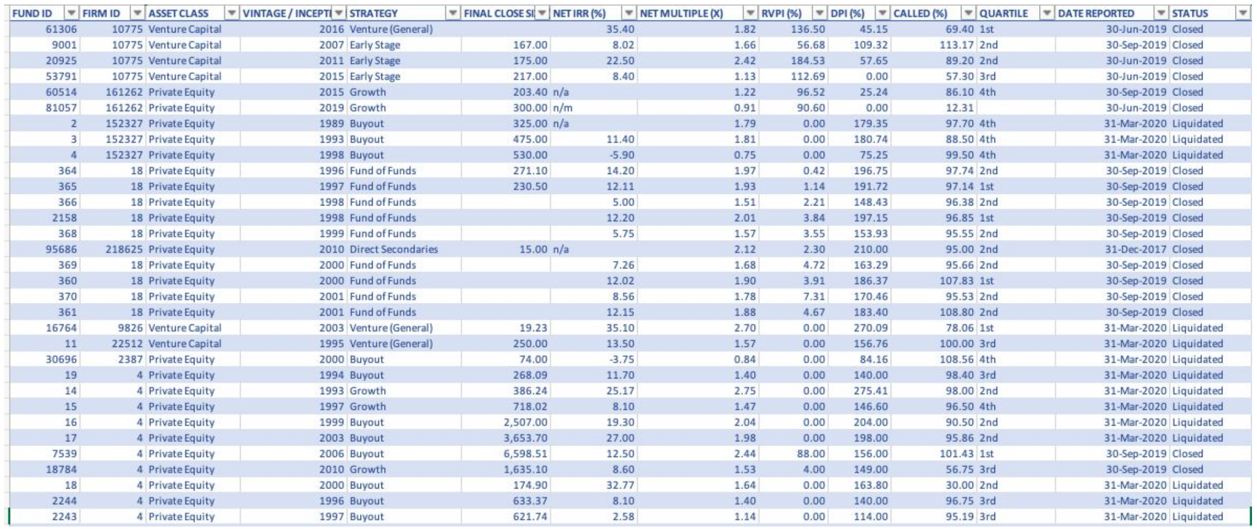
	All-2000	unemployment	ISMmanu	GDP	PPI	ISMnon-manu	NFP	core PPI m_m	core PPI year	retail index	core retail index	consum produ ind m_
1	-0.028988	-0.061558	0.040521	4.605170	-0.405465	0.066323	0.274199	0.080043	0.271934	0.000000	-0.167054	0.2411
2	-0.029159	-0.062739	0.040236	4.604636	-0.403192	0.066560	0.272045	0.080474	0.269238	0.002544	-0.164711	0.2423
3	-0.024862	-0.062464	0.040970	4.571258	-0.397430	0.066330	0.266738	0.076631	0.267695	0.002070	-0.163460	0.2358
4	-0.022794	-0.062174	0.041339	4.541277	-0.391033	0.065369	0.262215	0.072392	0.263038	0.004125	-0.160107	0.2245
5	-0.021840	-0.062685	0.043334	4.506594	-0.382411	0.064203	0.260196	0.069907	0.258242	0.007002	-0.151640	0.2155
6	-0.015130	-0.062657	0.040900	4.472050	-0.378386	0.064318	0.256857	0.066753	0.256213	0.011741	-0.153009	0.2062
7	-0.012122	-0.060350	0.044017	4.441736	-0.371397	0.066703	0.253902	0.062412	0.253796	0.011043	-0.149150	0.1964
8	-0.007727	-0.061418	0.047200	4.413340	-0.365077	0.065051	0.244936	0.058938	0.250253	0.011672	-0.145757	0.1897
9	-0.004303	-0.059103	0.040826	4.375984	-0.354895	0.060815	0.241016	0.054217	0.242965	0.016757	-0.142933	0.1813
10	-0.000640	-0.058710	0.045301	4.344322	-0.347594	0.065961	0.235799	0.049126	0.236193	0.015650	-0.139115	0.1687
11	-0.000700	-0.058961	0.045048	4.308721	-0.341403	0.066423	0.232640	0.042693	0.237034	0.015194	-0.130058	0.1636
12	0.005869	-0.057071	0.045723	4.279401	-0.336445	0.063846	0.225690	0.040349	0.227915	0.021366	-0.131334	0.1530

Data Exploration and Databases

Though we basically inherit the data from the previous team, we put a lot of effort on trying to get better data. Since we are trying to mimic the contribution and distribution of funds in a time sequence, it is more helpful if we can have data that demonstrates the activities in its life cycle, for example from vintage to close. However, the benchmark data is the yearly situation of the market, which does not necessarily align with the life cycle of a fund. Therefore, we think it would be better if we can have other data. Limited to the accessibility of the databases, we mainly explored the following two databases: Preqin and Capital IQ.

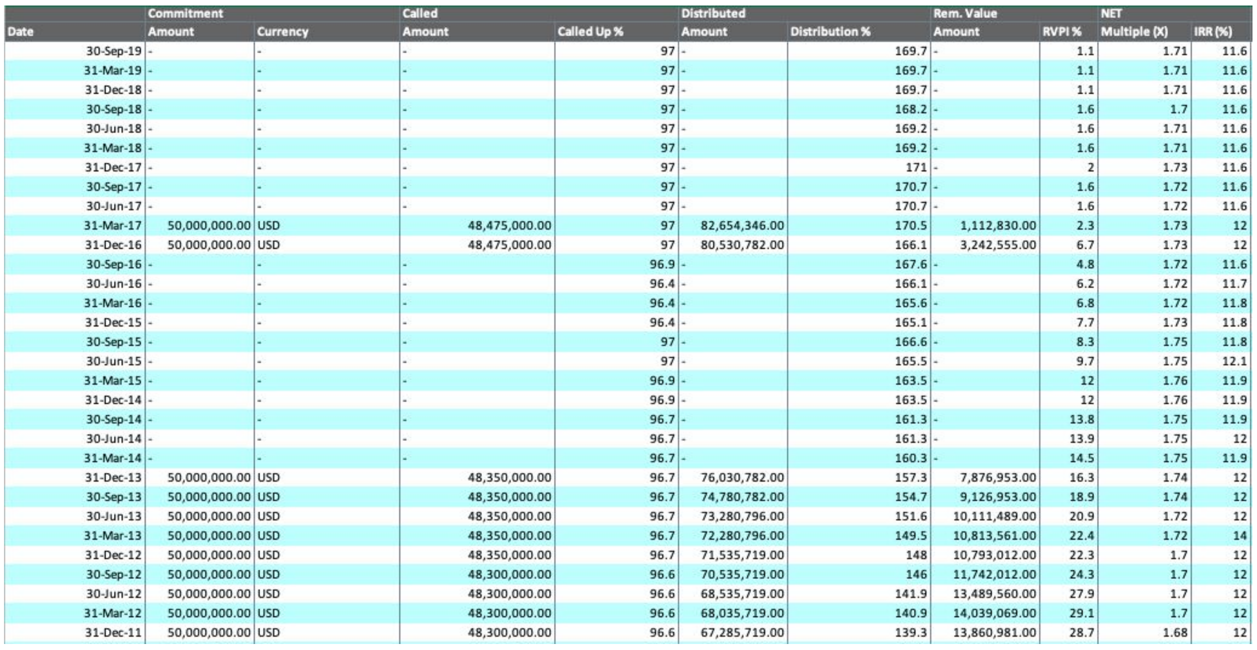
Preqin is a database focusing on PE and VC funds. Under the Performance tag, we can see fund specific historical performance data. Users have access to download two kinds of performance data: historical funds info data and funds specific historical performance data.

The former one includes the call-up and distributions but at a specific time point, now or at the point the fund closed depends on whether the fund is closed or not. The following picture shows a slice of this data. This is not helpful because data at a specific time point can’t help with tracking the activities.



FUND ID	FIRM ID	ASSET CLASS	VINTAGE / INCEPTI	STRATEGY	FINAL CLOSE SI	NET IRR (%)	NET MULTIPLE (X)	RVPI (%)	DPI (%)	CALLED (%)	QUARTILE	DATE REPORTED	STATUS
61306	10775	Venture Capital	2016	Venture (General)		35.40	1.82	136.50	45.15	69.40	1st	30-Jun-2019	Closed
9001	10775	Venture Capital	2007	Early Stage	167.00	8.02	1.66	56.68	109.32	113.17	2nd	30-Sep-2019	Closed
20925	10775	Venture Capital	2011	Early Stage	175.00	22.50	2.42	184.53	57.65	89.20	2nd	30-Jun-2019	Closed
53791	10775	Venture Capital	2015	Early Stage	217.00	8.40	1.13	112.69	0.00	57.30	3rd	30-Jun-2019	Closed
60514	161262	Private Equity	2015	Growth	203.40	n/a	1.22	96.52	25.24	86.10	4th	30-Sep-2019	Closed
81057	161262	Private Equity	2019	Growth	300.00	n/m	0.91	90.60	0.00	12.31		30-Jun-2019	Closed
2	152327	Private Equity	1989	Buyout	325.00	n/a	1.79	0.00	179.35	97.70	4th	31-Mar-2020	Liquidated
3	152327	Private Equity	1993	Buyout	475.00	11.40	1.81	0.00	180.74	88.50	4th	31-Mar-2020	Liquidated
4	152327	Private Equity	1998	Buyout	530.00	-5.90	0.75	0.00	75.25	99.50	4th	31-Mar-2020	Liquidated
364	18	Private Equity	1996	Fund of Funds	271.10	14.20	1.97	0.42	196.75	97.74	2nd	30-Sep-2019	Closed
365	18	Private Equity	1997	Fund of Funds	230.50	12.11	1.93	1.14	191.72	97.14	1st	30-Sep-2019	Closed
366	18	Private Equity	1998	Fund of Funds		5.00	1.51	2.21	148.43	96.38	2nd	30-Sep-2019	Closed
2158	18	Private Equity	1998	Fund of Funds		12.20	2.01	3.84	197.15	96.85	1st	30-Sep-2019	Closed
368	18	Private Equity	1999	Fund of Funds		5.75	1.57	3.55	153.93	95.55	2nd	30-Sep-2019	Closed
95686	218625	Private Equity	2010	Direct Secondaries	15.00	n/a	2.12	2.30	210.00	95.00	2nd	31-Dec-2017	Closed
369	18	Private Equity	2000	Fund of Funds		7.26	1.68	4.72	163.29	95.66	2nd	30-Sep-2019	Closed
360	18	Private Equity	2000	Fund of Funds		12.02	1.90	3.91	186.37	107.83	1st	30-Sep-2019	Closed
370	18	Private Equity	2001	Fund of Funds		8.56	1.78	7.31	170.46	95.53	2nd	30-Sep-2019	Closed
361	18	Private Equity	2001	Fund of Funds		12.15	1.88	4.67	183.40	108.80	2nd	30-Sep-2019	Closed
16764	9826	Venture Capital	2003	Venture (General)	19.23	35.10	2.70	0.00	270.09	78.06	1st	31-Mar-2020	Liquidated
11	22512	Venture Capital	1995	Venture (General)	250.00	13.50	1.57	0.00	156.76	100.00	3rd	31-Mar-2020	Liquidated
30696	2387	Private Equity	2000	Buyout	74.00	-3.75	0.84	0.00	84.16	108.56	4th	31-Mar-2020	Liquidated
19	4	Private Equity	1994	Buyout	268.09	11.70	1.40	0.00	140.00	98.40	3rd	31-Mar-2020	Liquidated
14	4	Private Equity	1993	Growth	386.24	25.17	2.75	0.00	275.41	98.00	2nd	31-Mar-2020	Liquidated
15	4	Private Equity	1997	Growth	718.02	8.10	1.47	0.00	146.60	96.50	4th	31-Mar-2020	Liquidated
16	4	Private Equity	1999	Buyout	2,507.00	19.30	2.04	0.00	204.00	90.50	2nd	31-Mar-2020	Liquidated
17	4	Private Equity	2003	Buyout	3,653.70	27.00	1.98	0.00	198.00	95.86	2nd	31-Mar-2020	Liquidated
7539	4	Private Equity	2006	Buyout	6,598.51	12.50	2.44	88.00	156.00	101.43	1st	30-Sep-2019	Closed
18784	4	Private Equity	2010	Growth	1,635.10	8.60	1.53	4.00	149.00	56.75	3rd	30-Sep-2019	Closed
18	4	Private Equity	2000	Buyout	174.90	32.77	1.64	0.00	163.80	30.00	2nd	31-Mar-2020	Liquidated
2244	4	Private Equity	1996	Buyout	633.37	8.10	1.40	0.00	140.00	96.75	3rd	31-Mar-2020	Liquidated
2243	4	Private Equity	1997	Buyout	621.74	2.58	1.14	0.00	114.00	95.19	3rd	31-Mar-2020	Liquidated

The later one is what we want but there are the problems regarding the data. The figure below is a sample of this data.



	Commitment		Called		Distributed		Rem. Value		NET	
Date	Amount	Currency	Amount	Called Up %	Amount	Distribution %	Amount	RVPI %	Multiple (X)	IRR (%)
30-Sep-19	-	-	-	97	-	169.7	-	1.1	1.71	11.6
31-Mar-19	-	-	-	97	-	169.7	-	1.1	1.71	11.6
31-Dec-18	-	-	-	97	-	169.7	-	1.1	1.71	11.6
30-Sep-18	-	-	-	97	-	168.2	-	1.6	1.7	11.6
30-Jun-18	-	-	-	97	-	169.2	-	1.6	1.71	11.6
31-Mar-18	-	-	-	97	-	169.2	-	1.6	1.71	11.6
31-Dec-17	-	-	-	97	-	171	-	2	1.73	11.6
30-Sep-17	-	-	-	97	-	170.7	-	1.6	1.72	11.6
30-Jun-17	-	-	-	97	-	170.7	-	1.6	1.72	11.6
31-Mar-17	50,000,000.00	USD	48,475,000.00	97	82,654,346.00	170.5	1,112,830.00	2.3	1.73	12
31-Dec-16	50,000,000.00	USD	48,475,000.00	97	80,530,782.00	166.1	3,242,555.00	6.7	1.73	12
30-Sep-16	-	-	-	96.9	-	167.6	-	4.8	1.72	11.6
30-Jun-16	-	-	-	96.4	-	166.1	-	6.2	1.72	11.7
31-Mar-16	-	-	-	96.4	-	165.6	-	6.8	1.72	11.8
31-Dec-15	-	-	-	96.4	-	165.1	-	7.7	1.73	11.8
30-Sep-15	-	-	-	97	-	166.6	-	8.3	1.75	11.8
30-Jun-15	-	-	-	97	-	165.5	-	9.7	1.75	12.1
31-Mar-15	-	-	-	96.9	-	163.5	-	12	1.76	11.9
31-Dec-14	-	-	-	96.9	-	163.5	-	12	1.76	11.9
30-Sep-14	-	-	-	96.7	-	161.3	-	13.8	1.75	11.9
30-Jun-14	-	-	-	96.7	-	161.3	-	13.9	1.75	12
31-Mar-14	-	-	-	96.7	-	160.3	-	14.5	1.75	11.9
31-Dec-13	50,000,000.00	USD	48,350,000.00	96.7	76,030,782.00	157.3	7,876,953.00	16.3	1.74	12
30-Sep-13	50,000,000.00	USD	48,350,000.00	96.7	74,780,782.00	154.7	9,126,953.00	18.9	1.74	12
30-Jun-13	50,000,000.00	USD	48,350,000.00	96.7	73,280,796.00	151.6	10,111,489.00	20.9	1.72	12
31-Mar-13	50,000,000.00	USD	48,350,000.00	96.7	72,280,796.00	149.5	10,813,561.00	22.4	1.72	14
31-Dec-12	50,000,000.00	USD	48,350,000.00	96.7	71,535,719.00	148	10,793,012.00	22.3	1.7	12
30-Sep-12	50,000,000.00	USD	48,300,000.00	96.6	70,535,719.00	146	11,742,012.00	24.3	1.7	12
30-Jun-12	50,000,000.00	USD	48,300,000.00	96.6	68,535,719.00	141.9	13,489,560.00	27.9	1.7	12
31-Mar-12	50,000,000.00	USD	48,300,000.00	96.6	68,035,719.00	140.9	14,039,069.00	29.1	1.7	12
31-Dec-11	50,000,000.00	USD	48,300,000.00	96.6	67,285,719.00	139.3	13,860,981.00	28.7	1.68	12

The first problem is that the database doesn’t allow batch downloading. We can only download the data fund by fund, which is mission impossible to manually download thousands of funds data and then combine them together. We tried web scraping to download the data but the scraping was blocked by the database. Second is about the data quality. The data of most PE funds are not public. The database gathers these information from different websites or sources. Therefore, the information of all the funds in the database are not of the same quality. Some of them have a longer time period from the start of the fund to the end while some only have a few rows. Some of them are completed while some have many NaNs. As a data user, it’s hard for us to distinguish between when the fund actually didn’t take any action and when is data missing. This flaw increases the difficulty of using this data.

Capital IQ is a more broadly focusing database. It contains mainly public data, which is also true for the PE funds. In the database, we can screen by company category. Specific funds data can be seen after selecting PE companies and choosing the funds raised by this company. However, like we mentioned above, this database provides mainly public data. Therefore, the funds data in the database are financial statement data, which is different from what we want.

Unfortunately, we don’t have access to more databases, thus can’t do more exploration. Given the situation described above, we failed to get more detailed, large scaled data for our models. Therefore, we kept the funds data used by the previous team and did some modifications as written in the previous section.

What’s more, we tried GANs to generate new data instead of the linear interpolation. However, we have so little data that the generator of the basic GANs model gave random numbers and the discriminator failed to distinguish the real numbers and the fake numbers. Therefore, we suggest the future group try more complex interpolation methods.

Models Architectures and Input Variables

The previous group creatively applied LSTM model to Private Equity data, however, because of the limitation on model modification in Pytorch, they decided to stick on the original LSTM architecture with sigmoid, tanh and linear activation function for memory, prediction and state cell respectively. Our group's objective is to improve the performance of the model by changing the model architecture, while keeping the interpolation method (stochastic interpolation), prediction horizon (360 days in the future), and length of historical data for prediction (270 days) the same. Based on the prototype model they have built, our group keep the optimizer and training epoch the same (Adam Optimizer & 100 epochs), and mainly modify the architecture to create the following models:

1. Baseline LSTM Model
2. Encoder-Decoder LSTM Model
3. CNN-LSTM Encoder-Decoder Model
4. ConvLSTM Encoder-Decoder Model

For each architectures, we test them on different types of input:

1. Univariate input with only Fund's cash-flow data
2. Multivariate input with cash-flow data and all macroeconomic factors we found
3. Multivariate input with cash-flow data and macroeconomic factors selected by Principal Component Analysis (PCA)

We also built a multi-horizon prediction model based on the best model we trained on all of the buyout fund data in order to look not only one year in advance but for quarters after that year as well to predict.

In terms of specific funds and type of cash flow selection, due to the number of models and different kinds of input, we choose to use the contribution cash flow in All Buyout

Funds, Read Estate Funds, and Venture Capital Funds. As for distribution data, as stated in the previous report, the funds usually have 0% distribution at the beginning one or two years, which makes the number of data input less than the contribution. Lso, the varying nature of distributions with different stages of a fund's life make the prediction of distributions from historical benchmark data intrinsically harder than that of contributions (Takahashi, 2002). Since our group's focus is to examine the model performance under different architectures and compare them with the previous results, therefore we decide to first concentrate on contribution data only in this report. Nevertheless, the distribution cash flow is also an important factor considered by fund managers and limited partners, which could be examined further by getting more data on this or trying different interpolation methods.

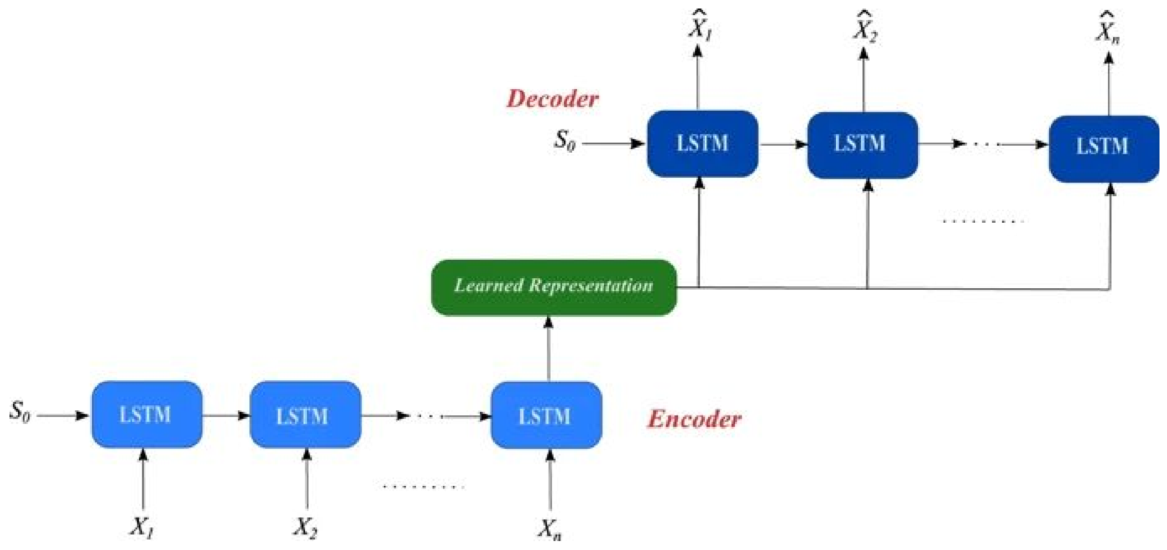
Architecture

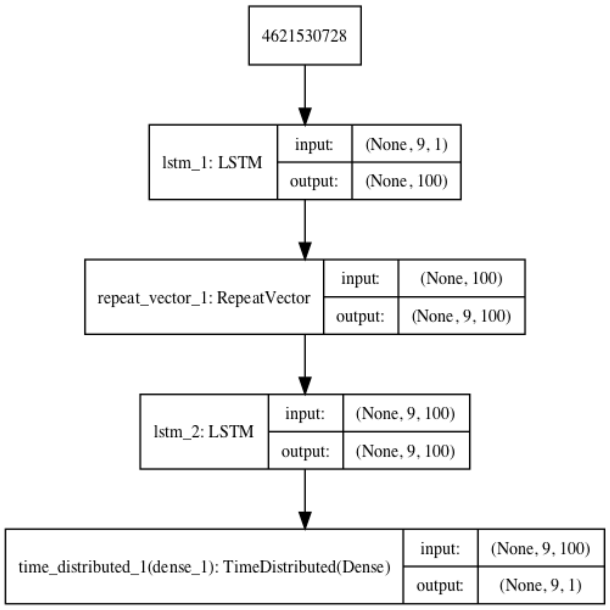
Baseline LSTM Model

For the vanilla LSTM model, we try to keep the most of the previous architecture, such as optimizer, the number of layers, epochs and loss function. However, we encounter the exploding gradient problem when we make the prediction. Though we have tried to use clip value, R2 regularization and batch normalization to put constraints on the weight, it doesn't perform as desired. Also, as the previous mentioned that they experienced vanishing gradient problems and periodic results, for the first problem we changed the state activation function to ReLU and keep this change for the further models, for the second one we continue to use mini-batch size equal to 100 to avoid periodic prediction.

Encoder-Decoder LSTM Model

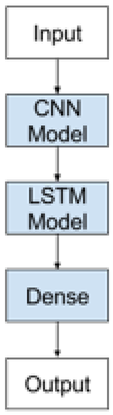
Rather than directly output the sequence to the fully connected layer, the encoder-decoder LSTM model has two sub-models, one as the encoder to take original input sequence and encode it, then the decoder read the encoded output as the next LSTM input sequence. The main difference here is that the LSTM model used in the decoder will know the predicted output from the previous LSTM and use that as an input to further generate the outputs, then passing to the fully connected layers, which in this case will be a TimeDistributed wrapper functionally the same as fully connected layers with only the encoder outputs become three dimensions, and example is demonstrated below (Srivastava, Mansimov, Salakhutdinov, 2015). This structure is more often used and performs better in multivariate time series forecasting problems(Sagheer & Kotb, 2019).

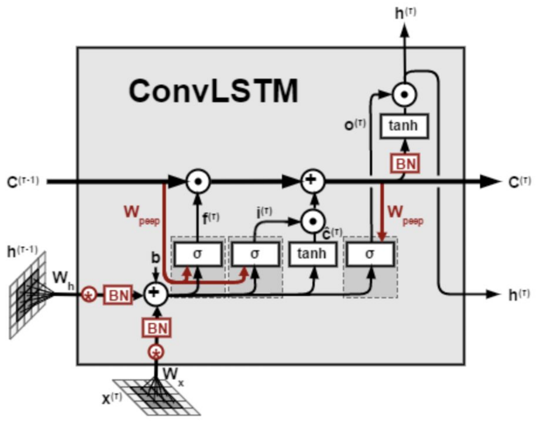




CNN-LSTM Encoder-Decoder Model

While LSTM could be used as the encoder, a convolutional neural network is able to perform similarly as an encoder as well. Though it is used in computer vision in most cases(Donahue al), CNN-LSTM model can also be used for time series data forecasting, especially for multivariate inputs(Du, Li & Yang, 2018). The CNN encoder can learn the relationship between time series data horizontally, in our case is how Funds' cash flow interacts with macroeconomic features like GDP, interest rate, etc. In this model we add two 1-d convolutional layers, and use 64 filters with kernel size equal to 3, and finally use the max pooling (2) to keep 1/4 of the values.





A ConvLSTM cell

ConvLSTM Encoder-Decoder Model

On the top of CNN-LSTM model, ConvLSTM is a further extension to what we have by now, which directly uses convolutions in reading the raw data, rather than take it to encoder CNN and interpret the output (see A ConvLSTM cell above). This model has a better performance in capturing spatiotemporal correlations(Shi, Chen & Wang, 2015). In our case, the combination of funds' cash flow macro economic factors could be interpreted as the two-dimension pixel data in the image input. Then we set the number of subsequences in ConvLSTM2D layer yo be 3 and change the length of each subsequence to 90, which makes the multiple of them equal to the original history length 270. This hyperparameter set up is intuitively straightforward as our raw data is on a quarterly basis, and we have performed stochastic interpolation between two given data values to generate 90 days data.

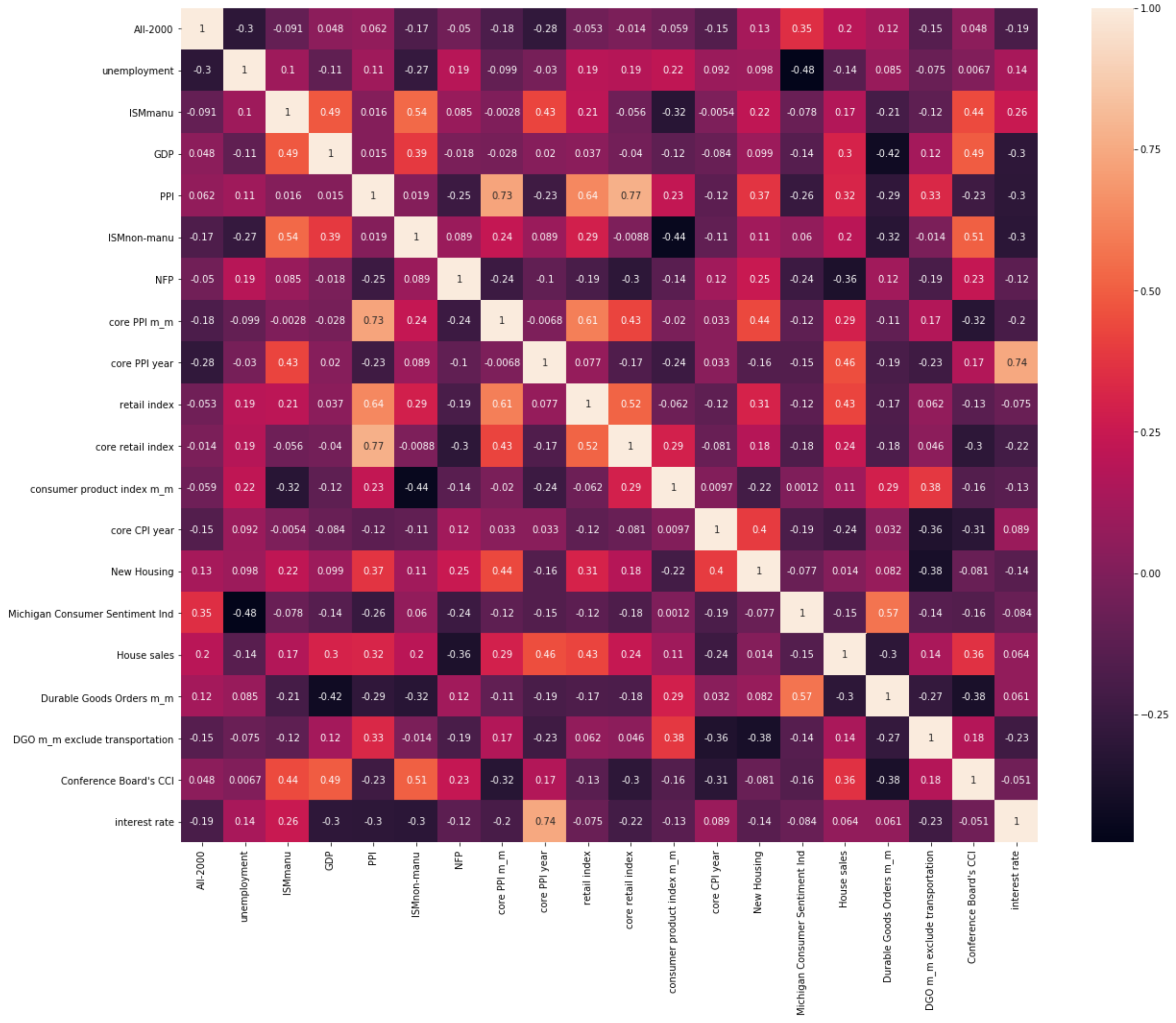
The future groups can examine those parameters in the Encoder-Decoder part since the parameters used in the convolutional layer are most experimented by image inputs. The number of filters, kernel size or units in the LSTM and fully connected layers still have a lot of room for improvement to find the best combination for each model and fund.

Variables

The primary feature we used to forecast is the interpolated log return of contribution cash flow (Called Up) for Buyout (All), Real Estate, and Venture Capital Funds. The data and interpolation method is exactly the same with the previous group so that we can make the results more comparable.

The extension we made on variables is to turn the inputs from one-dimensional to multidimensional by adding 19 macroeconomic factors obtained from EastMoney, a Chinese financial and stock information website provider. The correlation map among these factors is shown below (Example: Real Estate). We believe that the database from a U.S. financial institution would be more robust but do not have the access to any. Assuming that the macro factors are correct, we concat the data frame to make the input shape equal to (n, hidden_size, number of features), where n is the size of training or testing data, hidden_size is the time length we used to predict the future (270 past days), and the number of features is 20 if we do not apply feature selection method and is 10 after using Principal Component Analysis. The result from PCA indicates that 9 components can explain over 99% of the total variance, therefore we set the number of components equal to 9, which is also an examinable parameter.

With these four main architectures and three different input types, we fit each model with different inputs to evaluate the performances.


All-2000
unemployment
ISMmanu
GDP
PPI
ISMnon-manu
NFP
core PPI m_m
core PPI year
retail index
core retail index
consumer product index m_m
core CPI year
New Housing
Michigan Consumer Sentiment Ind
House sales
Durable Goods Orders m_m
DGO m_m exclude transportation
Conference Board's CCI
interest rate
1.00
0.75
0.50
0.25
0.00
−0.25

Result Comparison and Discussion

The previous team has achieved impressive Mean Square Errors under both training set and testing with 100 epochs and the sequence length of 270 for the data we used in All-Buyout, Real Estate and VC funds, which are all below 0.01. However, they experienced noisy predictions for both train and test set and the prediction curve seems to fluctuate up and down for all types of funds. Additionally, the models they built tend to overpredict the log return value in the test set by 0.05. After the changes in the architecture, our models tend to be more smooth and the overprediction problem has been eased in some of the models, especially in the Real Estate funds data - the data that achieve the best MSE performance for the previous group. It could possibly be the change in architecture, parameters and activation functions that make this improvement possible.

The disappearance of overprediction and non-noisy prediction gives rise to less MSE in our model, which are demonstrated in the following tables. To be more specific, the minimum test set MSE we have achieved for All Buyout Funds is 0.001238 using ConvLSTM Model with all macroeconomic features and contribution as input, 85% less than the previous best result 0.008121. For Real Estate Funds our CNN- LSTM Model with univariate input performs the best and gives a MSE of 0.000202, 92% less than the previous 0.0023928. Last but not least, VC Funds ConvLSTM Model with univariate input provides the minimum MSE of 0.00146, which is 98% less than the previous value.

We observe that for different types of funds, the best result is achieved by different models and inputs. Some funds might behave according to the macroeconomic conditions while others could have considered it as noise and perform worse if we add them into the input. Also, the minimum MSE does not indicate that the combination of model and input is the only optimal one. Some of the MSEs are very close to each other and because of the limit in training epochs(100) and different initialization state, the evaluation could also be slightly different.

The possible reasons for outperforming in the MSE compared to the former results might be explained by the following modifications: Firstly, we add some macroeconomic factors, which intuitively correlate to the contribution cash flow of a fund, especially when we are using general data for a type of funds rather than one particular fund, the macroeconomic features could be an indicator for Private Equity market. However, adding more factors might not always lead to a better outcome. For future groups, they can collaborate with PE with the Market Indicators team to find out more about the influential elements for Private Equity. Secondly, we change the baseline model's units (from 100 to 200) in LSTM and activation function (linear to ReLU). However the change in hyperparameters is still examinable and we can not assure it is the best combination. Finally, the encoder-decoder LSTM model and convolutional layers has already been proved that it can better capture the multivariate time series features as it can figure out the important messages hidden in the cash flow pattern.

Despite the improvement of MSE values, we need to also focus on the "real data" that is not interpolated by the stochastic process, which are those points representing the actual log return for the quarter - the peak values. However, a detailed look into our test set predictions and actual values will discover some mismatches between the two. For example, the actual value goes to 0.1 to 0 while the predicted value goes from 0 to 0.1. Even though this prediction will give us a relatively less MSE than predicting the value from 0.5 to 0.4, this model misunderstands the trend and might lead to an unsatisfactory result. Indeed, no matter how accurate the model we use, the future prediction won't go beyond the historical pattern and mismatch happens. It will still be a great challenge in time series forecasting, especially in the Private Equity sector where the data is usually difficult to find and the time gap between each data released date is longer than other financial assets like stocks, futures, etc.

The following tables show the Testing set MSE comparison between our models and the previous best result. Since the different weight initializations will affect the final results, a more robust method could be run each model several times to take the average of them. Because of the time limit, we didn't perform this test in our report. However, the scale of MSE is similar as we observed in the test run, therefore we can assume the model performance can be measured by the Testing set MSE.

All Buyout Funds				
	LSTM	Encoder-Decoder LSTM	CNN-LSTM	ConvLSTM
Univariate	0.003433	0.003014	0.001763	0.003214
Multivariate(20)	0.002658	0.002565	0.003666	**<u>0.001238</u>**
Multivariate(PCA)	0.002613	0.001973	0.001621	0.002994
Previous Result	0.008121		85% ↓	

Real Estate Funds				
	LSTM	Encoder-Decoder LSTM	CNN-LSTM	ConvLSTM
Univariate	0.001338	0.001116	**<u>0.000202</u>**	0.000415
Multivariate(20)	0.000439	0.001293	0.000482	0.000636
Multivariate(PCA)	0.000329	0.001510	0.000235	0.000437

Previous Result	0.002393	92% ↓

Venture Capital Funds				
	LSTM	Encoder-Decoder LSTM	CNN-LSTM	ConvLSTM
Univariate	0.001125	0.000710	0.004908	**<u>0.000146</u>**
Multivariate(20)	0.001189	0.001246	0.000665	0.000291
Multivariate(PCA)	0.000712	0.001206	0.000770	0.000536
Previous Result	0.006540		98%↓	

Multi-Horizon Prediction

Used Multivariate ConvLSTM model for Multi-Horizon prediction model, and was able to get a Train set MSE of 0.00033705793340350486 and Test set MSE is 0.002652866528976238. This may seem significantly higher than the other models we have tested, however the multi-horizon prediction has more uncertainty in its predictions as the time horizon increases as well as the fact that it is counting more points when calculating the MSE so therefore it is expected to be higher. There is still a lot of value in being able to relatively accurately predict up to two years in advance rather than only one year.

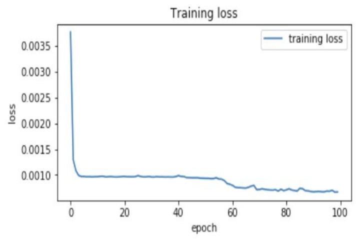
Visualization of Loss Function and Actual/Predict in Train & Test Set

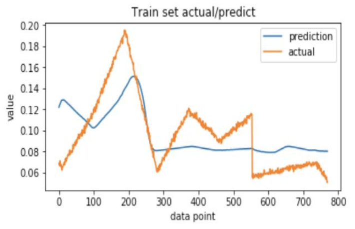
Left: Training Loss
Mid: Training Set Actual Value/Prediction
Right: Testing Set Actual Value/Prediction

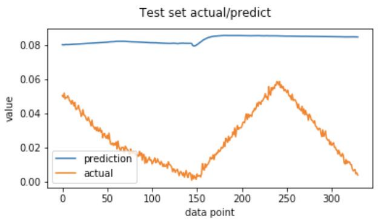
All Buyout Fund

1. Model: Baseline LSTM

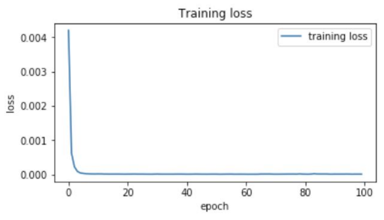
   Input: Univariate
   Train set MSE is 0.0006549997107659761
   Test set MSE is 0.0034333645650489486

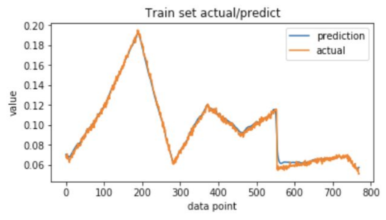


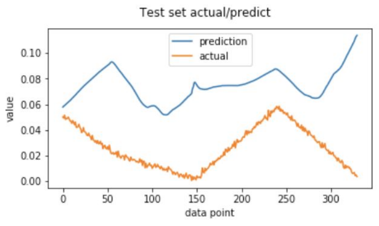




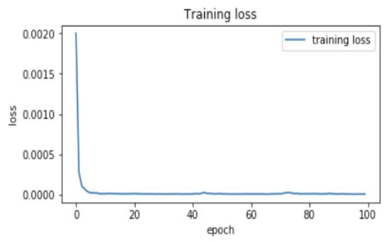
   Input: Multivariate (All-factor)
   Train set MSE is 8.177399627701792e-06
   Test set MSE is 0.0026589672355913784

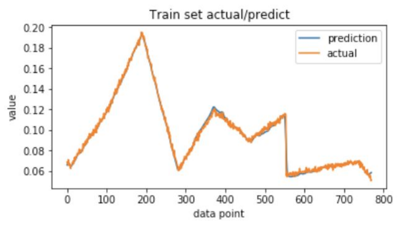


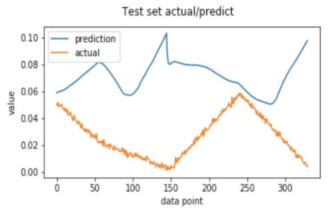




   Input: PCA Selected Features
   Train set MSE is 6.4391721091126745e-06
   Test set MSE is 0.0026137282807406894





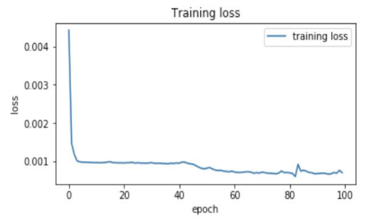


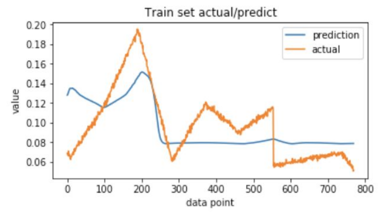
2. Model: Encoder-Decoder LSTM

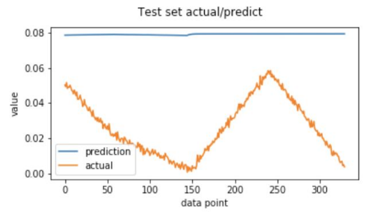
Input: Univariate

Train set MSE is 0.0006748829093227509

Test set MSE is 0.0030147464299767482



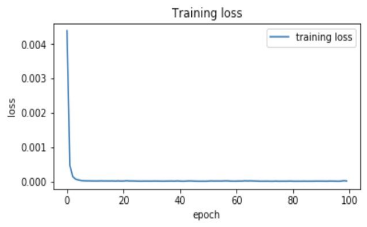


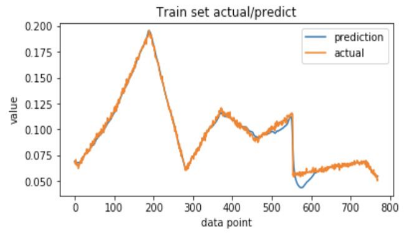


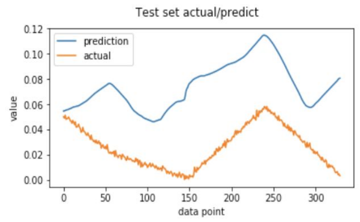
Input: Multivariate (All-factor)

Train set MSE is 1.3852581075816028e-05

Test set MSE is 0.0025655700343377307



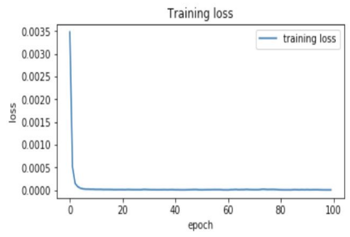


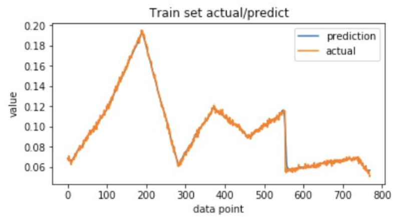


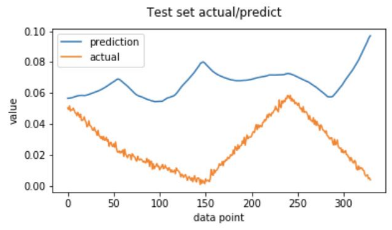
Input: PCA Selected Features

Train set MSE is 6.1654821744355575e-06

Test set MSE is 0.001973766235568837





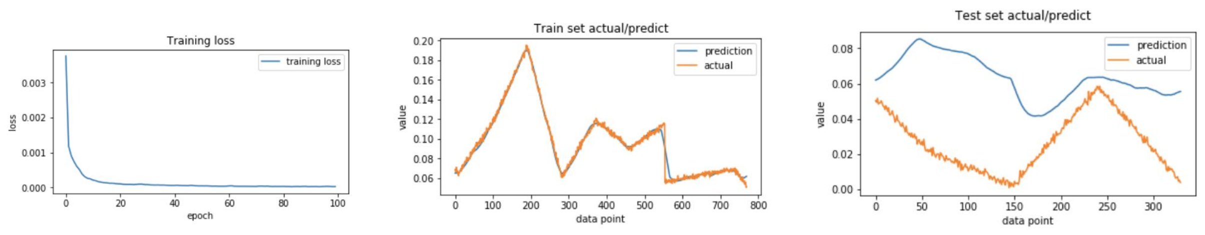


3. Model: CNN-LSTM

Input: Univariate

Train set MSE is 1.6415126130313443e-05

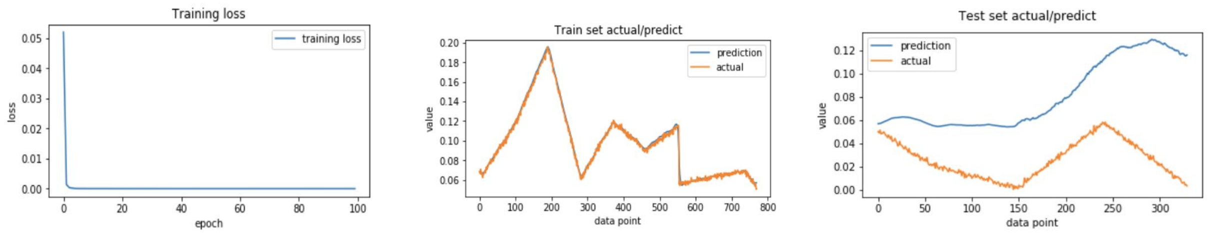
Test set MSE is 0.0017634873721841956



Input: Multivariate (All-factor)

Train set MSE is 7.218706969019849e-06

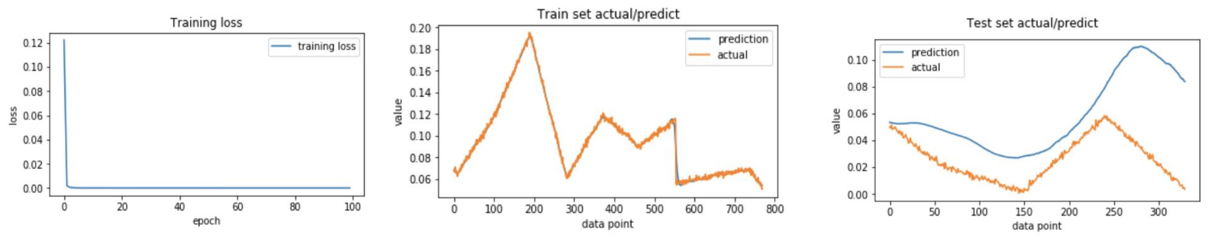
Test set MSE is 0.0036667311295287895



Input: PCA Selected Features

Train set MSE is 6.2321488141824164e-06

Test set MSE is 0.0016216223675102684

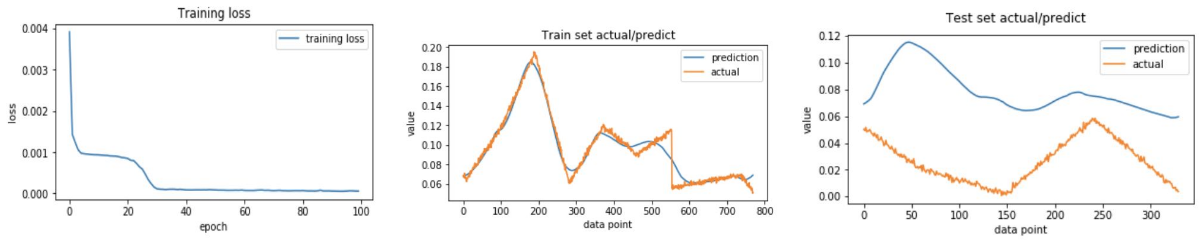


4. Model: ConvLSTM

Input: Univariate

Train set MSE is 4.974198232438426e-05

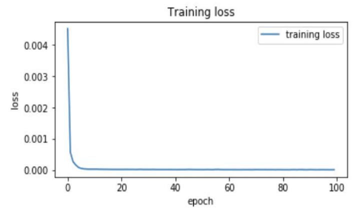
Test set MSE is 0.003214774548321893

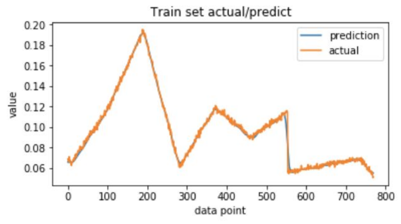


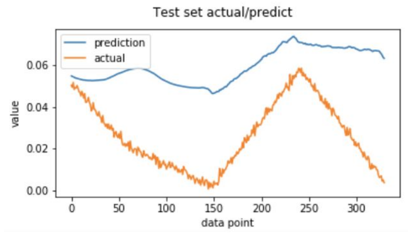
Input: Multivariate (All-factor)

Train set MSE is 7.924394474608798e-06

Test set MSE is 0.0012380390212529586



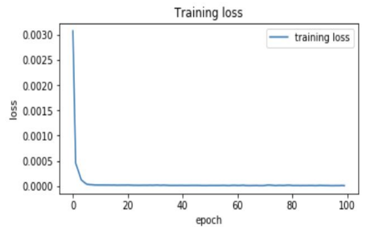


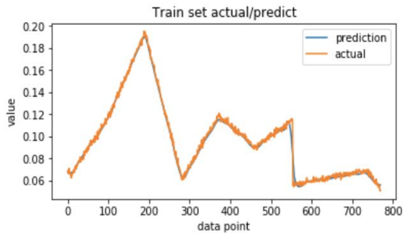


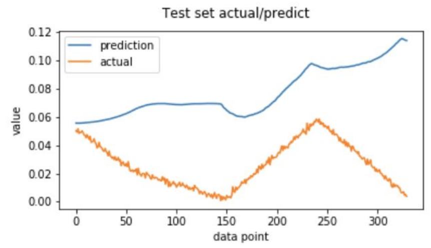
Input: PCA Selected Features

Train set MSE is 9.100457667517129e-06

Test set MSE is 0.0029949832754286062



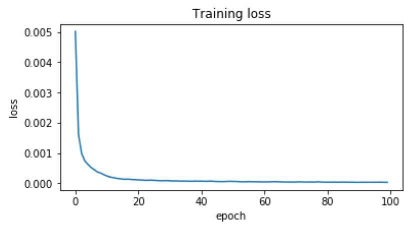


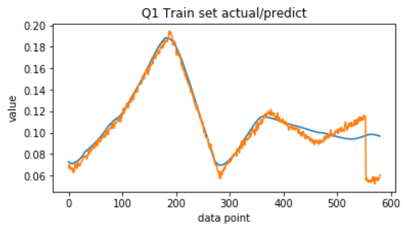


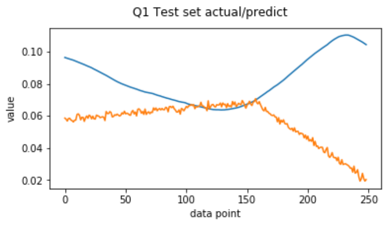
5. Model: Best Multi-horizon Model

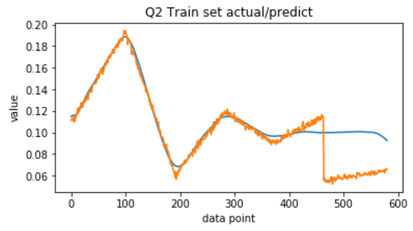
Train set MSE is 0.00033705793340350486

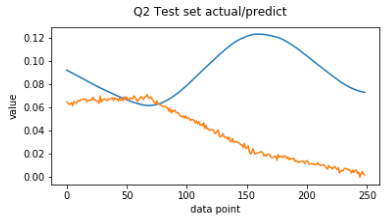
Test set MSE is 0.002652866528976238

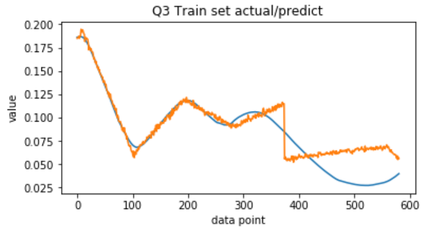


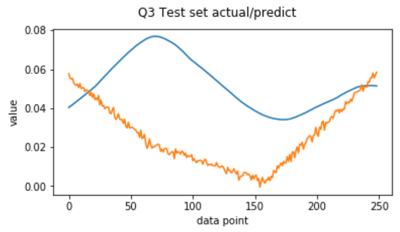


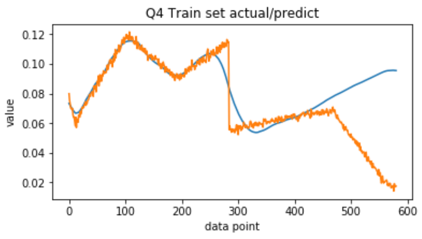


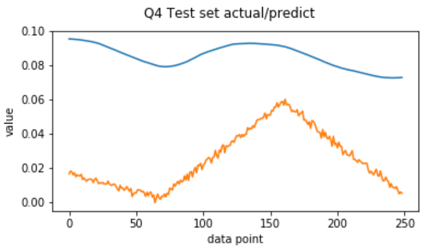












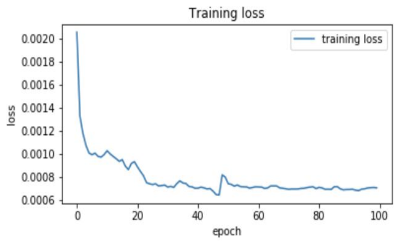
Real Estate Fund

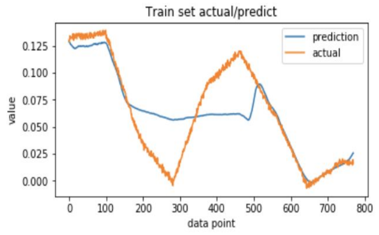
1. Model: Baseline LSTM

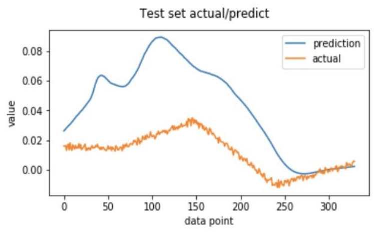
Input: Univariate

Train set MSE is 0.0006773761847043598

Test set MSE is 0.0013380424088606198



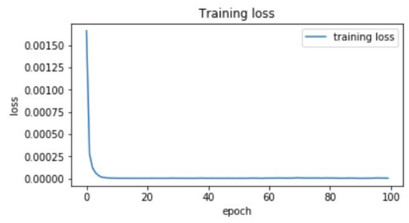


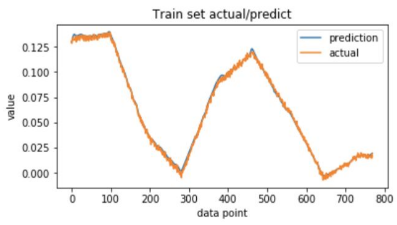


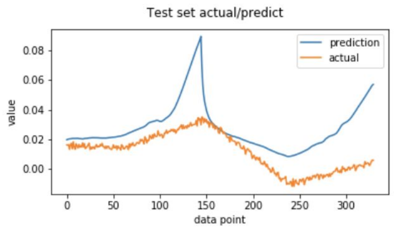
Input: Multivariate (All-factor)

Train set MSE is 5.149862976147644e-06

Test set MSE is 0.00043983321127729 83



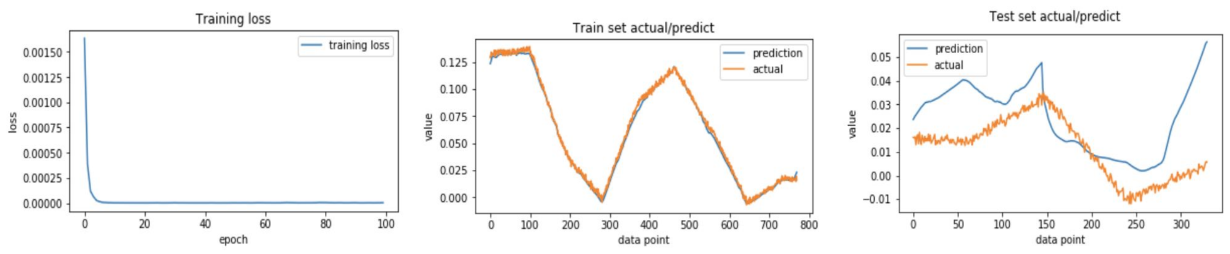




Input: PCA Selected Features

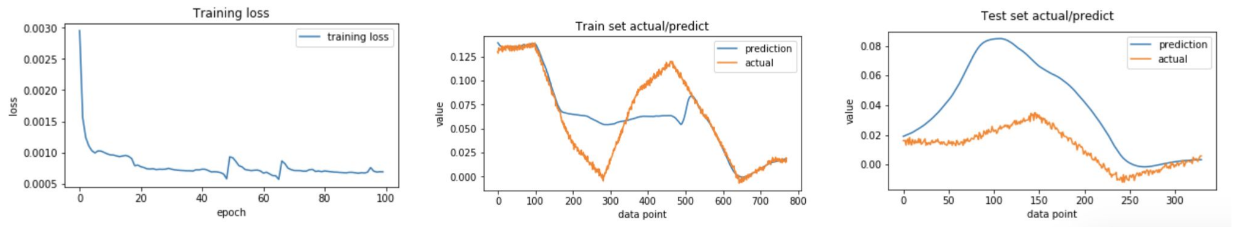
Train set MSE is 8.636933969532945e-06

Test set MSE is 0.0003298860952482105

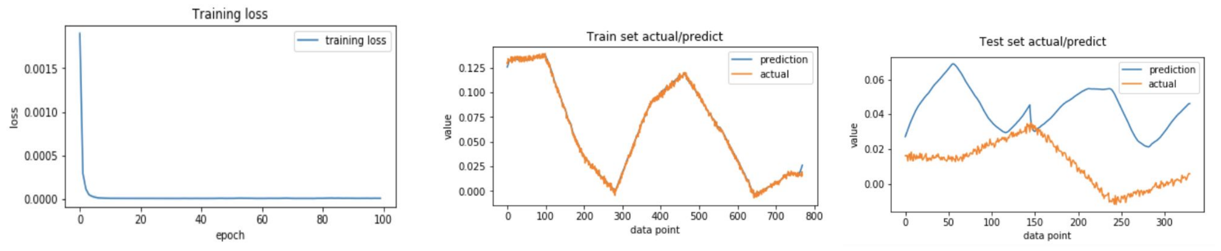


2. Model: Encoder-Decoder LSTM

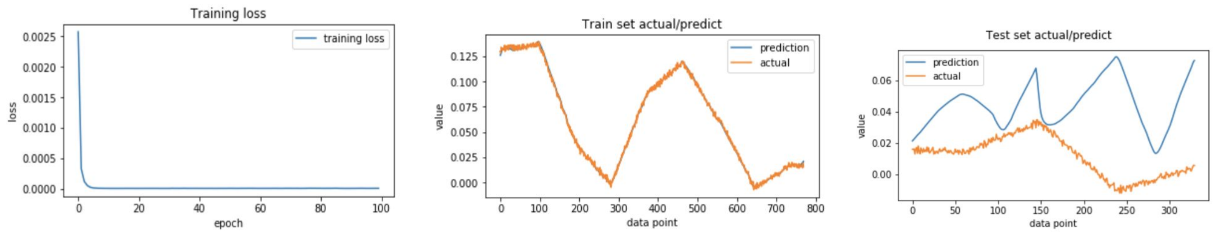
Input: Univariate
Train set MSE is 0.0006504954578443078
Test set MSE is 0.0011163749376717216



Input: Multivariate (All-factor)
Train set MSE is 3.5614861768792964e-06
Test set MSE is 0.0012939465736106603



Input: PCA Selected Features
Train set MSE is 3.1999504813966872e-06
Test set MSE is 0.001510234629211043

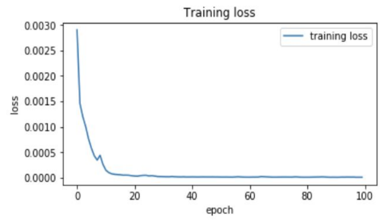


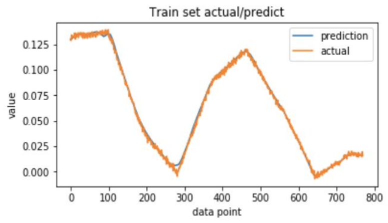
3. Model: CNN-LSTM

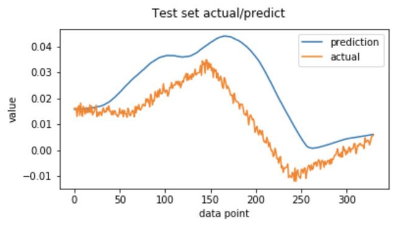
Input: Univariate

Train set MSE is 5.684404996077497e-06

Test set MSE is 0.00020296780220297294



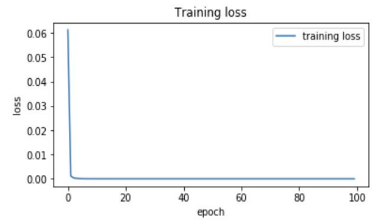


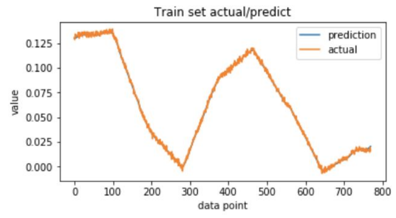


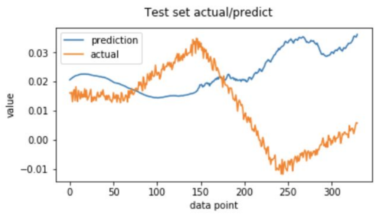
Input: Multivariate (All-factor)

Train set MSE is 2.923847751081172e-06

Test set MSE is 0.00048293549424407344



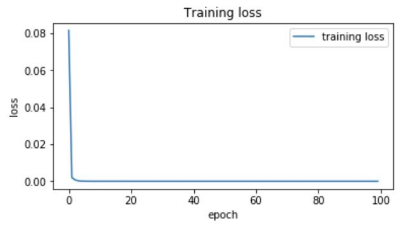


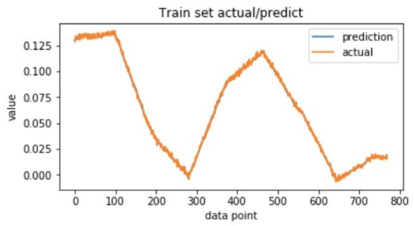


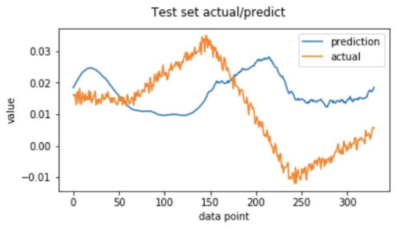
Input: PCA Selected Features

Train set MSE is 2.8420242919978478e-06

Test set MSE is 0.00023551123035649477





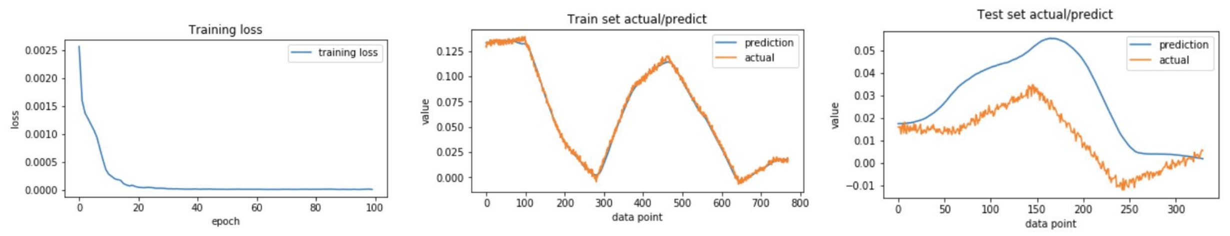


4. Model: ConvLSTM

Input: Univariate

Train set MSE is 5.721947279469896e-06

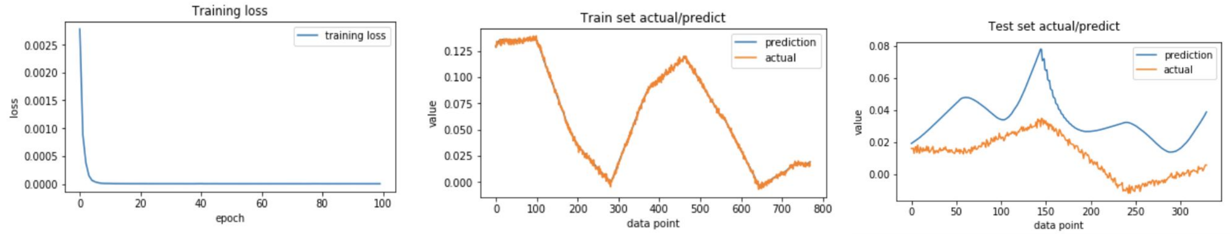
Test set MSE is 0.0004155101480850257



Input: Multivariate (All-factor)

Train set MSE is 2.9017021030301465e-06

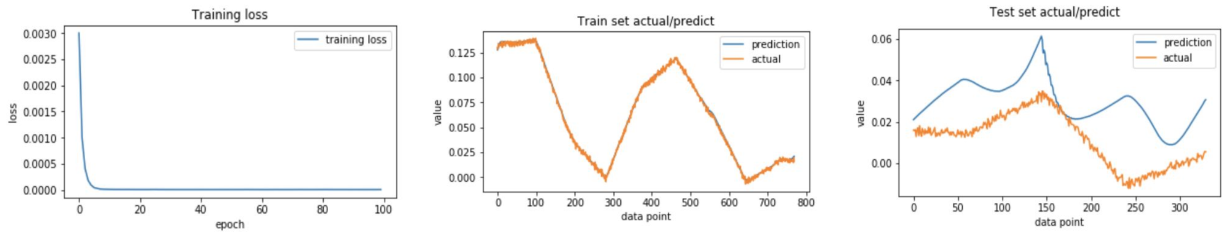
Test set MSE is 0.0006362623433158646



Input: PCA Selected Features

Train set MSE is 3.598598715032704e-06

Test set MSE is 0.00043796406433010027



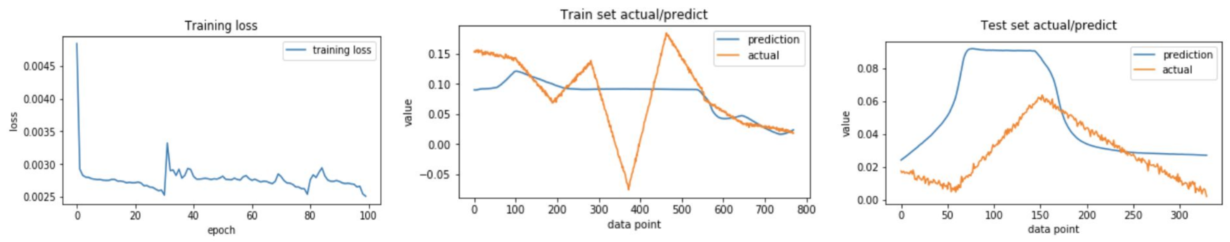
Venture Capital Fund

1. Model: Baseline LSTM

Input: Univariate

Train set MSE is 0.00242864089637308

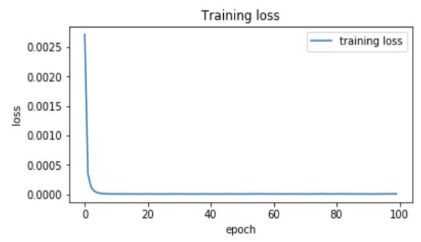
Test set MSE is 0.0011251215212489155

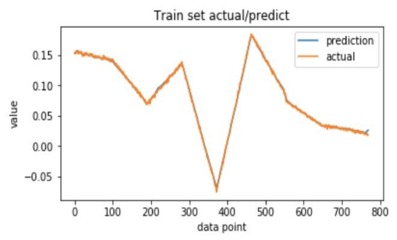


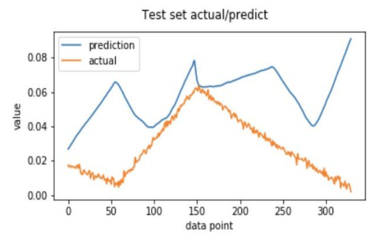
Input: Multivariate (All-factor)

Train set MSE is 3.783381623114348e-06

Test set MSE is 0.001189963432621182



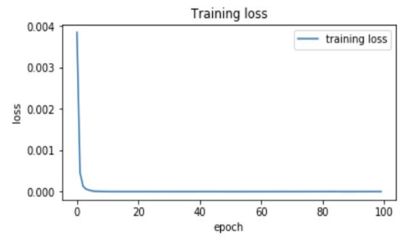


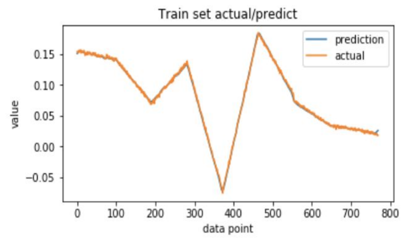


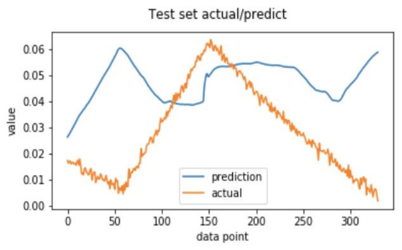
Input: PCA Selected Features

Train set MSE is 4.531771853487727e-06

Test set MSE is 0.0007127502499383113





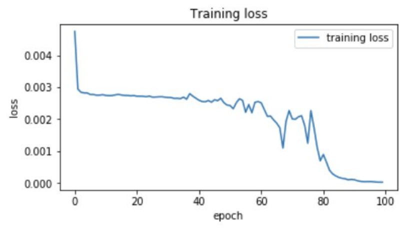


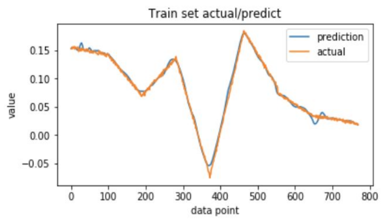
2. Model: Encoder-Decoder LSTM

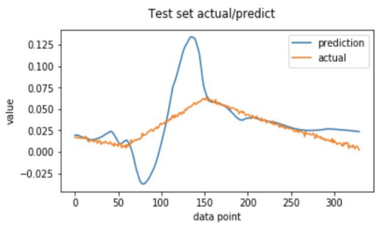
Input: Univariate

Train set MSE is 2.2558564342148055e-05

Test set MSE is 0.0007105117416587866



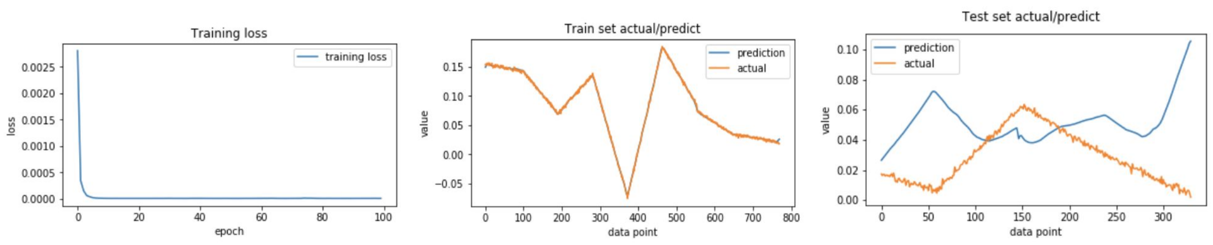




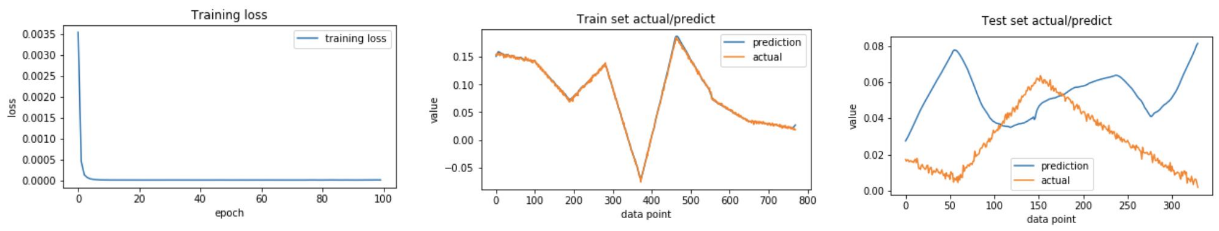
Input: Multivariate (All-factor)

Train set MSE is 3.0357990846902254e-06

Test set MSE is 0.0012467722024268053

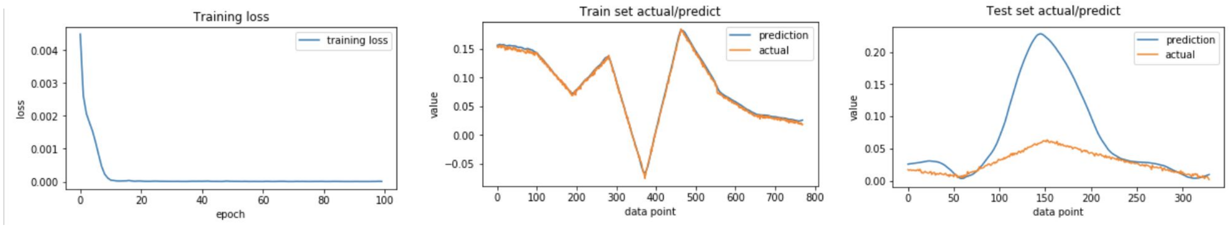


Input: PCA Selected Features
Train set MSE is 4.981166732821954e-06
Test set MSE is 0.0012064668374373471

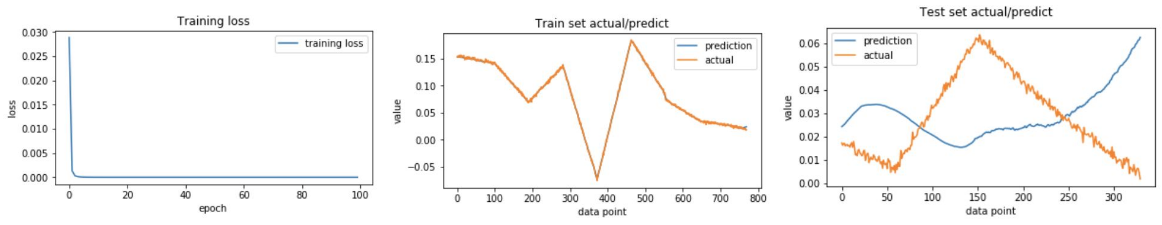


3. Model: CNN-LSTM

Input: Univariate
Train set MSE is 1.1960764657986165e-05
Test set MSE is 0.004908687091169263

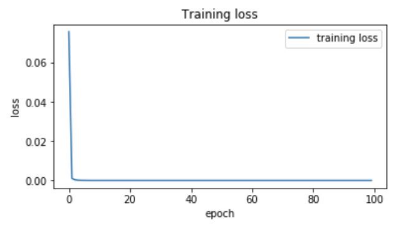


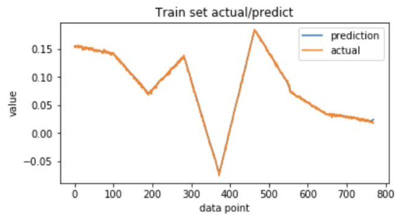
Input: Multivariate (All-factor)
Train set MSE is 2.779537999607492e-06
Test set MSE is 0.0006654913606920099

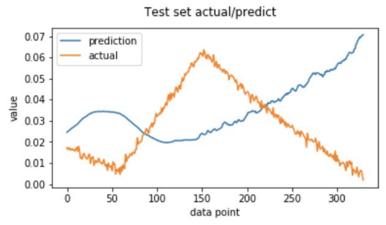


Input: PCA Selected Features
Train set MSE is 3.2687673613994777e-06

Test set MSE is 0.0007700962210716858





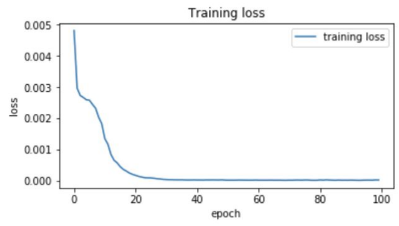


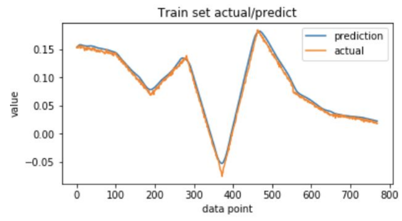
4. Model: ConvLSTM

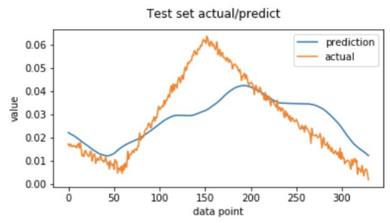
Input: Univariate

Train set MSE is 3.775532829770266e-05

Test set MSE is 0.00014653712921160433



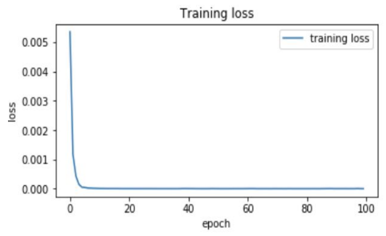


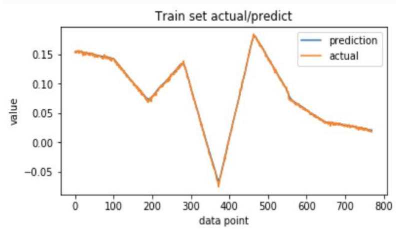


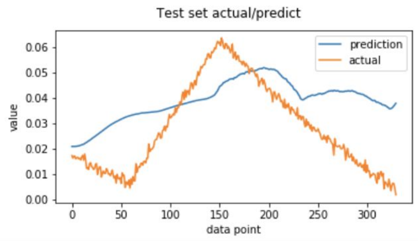
Input: Multivariate (All-factor)

Train set MSE is 4.4709348825411295e-06

Test set MSE is 0.00029197641718038236



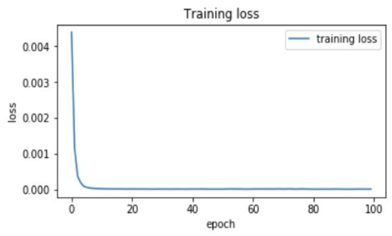


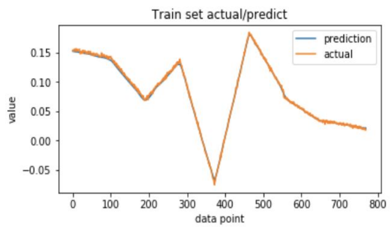


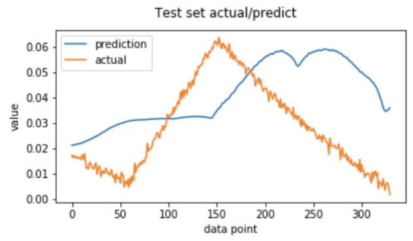
Input: PCA Selected Features

Train set MSE is 6.302606849085836e-06

Test set MSE is 0.0005364992663839686







Conclusion and Future Improvements

Our main aim was to ameliorate the modeling that was done by the previous group by building on their LSTM model for one year in advanced prediction of contributions and distributions of buyout PE funds. We have built on their model by adding different architectures into the LSTM model layers in order to improve the fitting of the model, taking into account different macro factors into our model as features to include, performing PCA on those factors using the PCA components as features, analyzing models and training different model on industry specific fund data, and finally increasing the prediction horizon of the previous model. By performing all of these tasks, along with tuning hyperparameters, we were able to significantly decrease our model train and test MSE, which we continued to use as our evaluation metric, as well as increase the prediction capabilities of the model.

There are many directions which one could take to further this project. One would be to replace the classical stochastic bridge interpolation with something more robust like a variance gamma brownian bridge or even a deep Conditional Generative Adversarial Network. You could also use ensemble methods to combine different LSTM models to achieve a better less overfit model. Moreover, you could try and get more contribution distribution data for all the kinds of PE funds in order to have a larger training set for each model. We contacted the service of Preqin and were notified that the fund specific historical performance data might be obtained through special requests and fees might be applied. If this work is intended to go deeper, a sufficient and robust dataset is required. Furthermore you could also experiment with replacing the LSTM model with another time series model, such as an auto regressive one, to make a different benchmark for model performance. You could even feed the result from the autoregressive model as another feature in the LSTM model and see how that affects the performance.

Reference

1. Macro factors:http://data.eastmoney.com/cjsj/foreign_0_9.html
2. A Gentle Introduction to LSTM Autoencoder:https://machinelearningmastery.com/lstm-autoencoders/
3. Unsupervised Learning of Video Representations using LSTMs, Nitish Srivastava, Elman Mansimov, Ruslan Salakhutdinov https://arxiv.org/abs/1502.04681
4. Unsupervised Pre-training of a Deep LSTM-based Stacked Autoencoder for Multivariate Time Series Forecasting Problems, Alaa Sagheer & Mostafa Kotb https://www.nature.com/articles/s41598-019-55320-6
5. Deep Air Quality Forecasting Using Hybrid Deep Learning Framework, Shengdong Du, Tianrui Li, Yan Yang, Shi-Jinn Horng https://arxiv.org/abs/1812.04783
6. Long-term Recurrent Convolutional Networks for Visual Recognition and Description, Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, Trevor Darrel https://arxiv.org/abs/1411.4389
7. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting, Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, Wang-chun Woo https://arxiv.org/abs/1506.04214v1
8. An introduction to ConvLSTM, Alexandre Xavier https://medium.com/neuronio/an-introduction-to-convlstm-55c9025563a7
9. How to Use Weight Decay to Reduce Overfitting of Neural Network in Keras https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/
10. A Gentle Introduction to Backpropagation Through Time https://machinelearningmastery.com/gentle-introduction-backpropagation-time/
11. How to Avoid Exploding Gradients with Gradient Clipping https://machinelearningmastery.com/how-to-avoid-exploding-gradients-in-neural-networks-with-gradient-clipping/