IEOR 4742 Deep Learning
Final Report

Time Series Forecast 2

Bowen Zhang, Chun Yat Yeung
bz2399, cy2623

Abstract

Alternative data are data collected from non-traditional sources and are used by investment firms to gain competitive edge. Examples of alternative features of mutual funds include administrators, auditors, custodians, distributors and managers, to name but a few. In this project, we combined the time-series data and alternative data to predict the future performance of mutual funds and two prediction frameworks are proposed: (1) `AltFNN`, a simple feedforward network (FNN) that merges time-series and alternative data, and (2) `LSTM-AltReg/FNN`, a separate architecture with two sub-networks – Long Short-Term Memory (LSTM) for time-series and regression/feedforward network for alternative data. The performance of the two frameworks and their variations are compared. It is observed that, while predictions using historical excess returns alone are not satisfactory, alternative data add predictive power, showing improvements in terms of mean-squared errors.

1 Introduction

Alternative data abound in the financial world, especially in securities analysis. For example, hedge funds are drawing on real-time satellite imagery to predict stock prices. Nowadays, geolocation, social media, mobile app usage, credit card payments, and even weather can be the inputs for financial model development. Alternative data provide additional, hidden information on mutual funds, which our performance models can use, along with time-series data, to produce more accurate predictions. Time series analysis is critical for estimating the value of fragmented, missing, and future data sources, with the purpose to understand the structure and pattern of time-series data. It is common in modeling key performance indicators. However, time-series data and cross-sectional alternative data usually have different dimensions. Thus, the primary goal of our work is to create an algorithm that combines them to achieve better accuracy in prediction.

In this project, we created prediction models that combine time-series and alternative data to forecast the three-year forward excess returns of mutual funds. As the baselines, we used the Long Short-Term (`LSTM`) model and Feedforward Neural Network (`FNN`) model catered to the time-series data. For the alternative dataset, pre-processing, labeling and encoding were applied. We sought additional improvement through collaboration with Embedding 2 group. In subsequent sections, we gave a statistical analysis on the time-series dataset, proposed our treatment of alternative data, experimented with different model architectures, and finally concluded by outlining prospective work. Notably, we implemented two different forecast frameworks that take care of alternative data: (1) **`AltFNN`**, a simple feedforward network that merges time-series and alternative data, and (2) **`LSTM-AltReg/FNN`**, a separate architecture with two sub-networks – Long Short-Term Memory for time-series and regression/feedforward network for alternative data. Network parameters and prediction results are described in detail.

2 Dataset

The two main datasets used in this project are:

1. `MF_LargeCap_ExcessReturn_3Y.parquet`,
2. `opt_people.parquet`.

2.1 Time Series Data

`MF_LargeCap_ExcessReturn_3Y.parquet` contains rolling 3-year backward excess returns of 1330 large-cap mutual funds from `2003-01-01` to `2021-08-31`, managed by asset management companies such as Fidelity and BlackRock. For example,

the excess returns of the first 50 funds evolve as follows. As expected, the series are centered around zero, indicating that funds sometimes outperform market but sometimes do not. It is seen that excess returns are typically contained within $[-0.2, 0.2]$. Note that because of the backward rolling window, excess returns on two close dates are highly correlated, and one should not naively feed consecutive dates into networks.

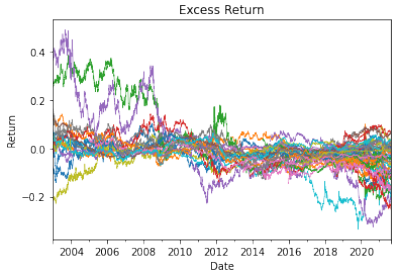

Figure 1: Excess return series

2.2 Single-Fund Analysis

Without loss of generality, we picked a particular fund, named `B42`, and assessed its performance relative to other mutual funds, in order to gain intuition for the excess return time series.

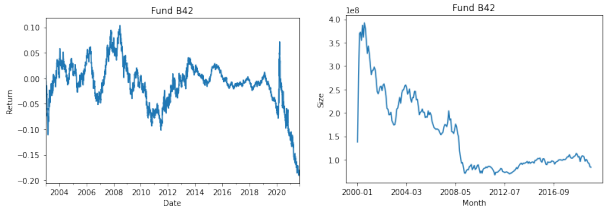

Figure 2: Excess return and fund size of `B42`

Just to make sense of the data, we see that `B42` does not perform so well lately around 2020, hence investors withdraw from their accounts and fund size drops as a result, from 4e8 in 2001 to 1e8 in 2020. As mutual funds are carrying out similar strategies, for instance, long-short in particular sectors/industries, geographic concentrations etc., one can expect that some mutual funds may be replicated well with other funds, exhibiting a high correlation between their excess returns. To see this, we performed a regression of `B42` returns versus other funds that exist for a comparable history. Specifically, we chose `2012-05-13` to `2020-04-07` as our timeframe, between which over 4000 days exist, and within this period, 866 funds are filtered out, including `B45`, `B209` and so on. Regression is applied to `B42` daily (rolling) returns, with returns of the other 866 funds as features. To simplify the model, recursively we picked features that exhibited "sufficiently high" t-statistics, at a threshold of 2, and concluded that `B42` fund may be replicated well with just 43 other funds, yet achieving an adjusted R-squared of 95.5%. Of course, one may desire other model simplifications, e.g. Lasso/Ridge regularization – we leave this for future research. The finding indicates that in the space of mutual funds, some are redundant and may be reproduced by a small combination by other funds, echoing with the two-fund theorem in an efficient market: mean-variance investors seek diversification between only two efficient funds, i.e. linear combination of two efficient funds produce all other efficient funds. In some sense, the finding reflects the degree of market efficiency in US equity market. Below we contrast the returns data generated by the replicating portfolio of 43 funds, called `predicted`, and `B42` itself, called `actual`, and a high degree of linearity is observed, despite a small portion of outliers.

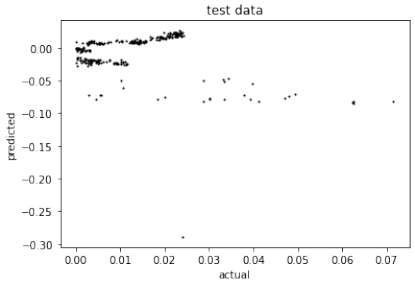

Figure 3: Replicating performance of `B42`

Now, a cross-sectional analysis of `B42` is completed, via regression with excess returns of other funds day by day. One may question: does historical time series of the fund predict future movements? Specifically, does previous returns snapshot of the entire fund space provide information on how the fund will perform today? Results indicate affirmative – following model simplification procedure described above, this time with a t-statistics threshold of 0.6, `t`-return of `B42` is found to correlate with `t-1`-returns of 49 other funds, giving an adjusted R-squared of 94.2%. What about a wider historical window? A one-day window may not suffice, and one would want to feed a wider window into the model, in scale of months. Long Short-Term Memory comes into play. Here, we choose to feed a 3-month backward historical window of `B42` and its associated 49 other funds into a LSTM single-layer 200-node network, with its prediction results as follows. A linearity is observed, which suggests LSTM can be of use in predicting from historical data. But to what extent? – Here, the caveat is that excess returns between consecutive days are highly correlated as they refer to approximately the same historical window, so we should not be over-confident in LSTM's power because naively using `t-1`-return of `B42` as its forecast value at `t` may do just as well. With this in mind, below we manipulated the time-series `parquet` file, ensuring data fed into models are spaced sufficiently far so that correlation effects die out.

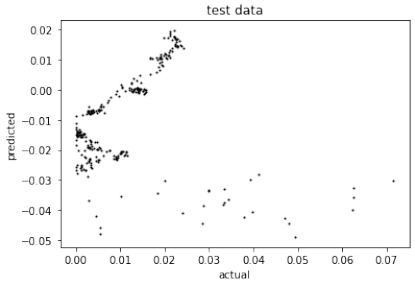

Figure 4: LSTM performance for `B42` using historical snapshots

2.3 Alternative Data

`opt_people.parquet` contains alternative fund information with 12 features: 'Administrator', 'Advisor', 'Auditor', 'Auditor Fee', 'Custodian', 'Distributor', 'Manager History', 'Manager of Managers', 'Manager Ownership Level', 'Subadvised', 'Subadvisor' and 'Superannuation'.

2.3.1 One-Hot Encoding

All features except `Auditor Fee` are categorical. The categorical features are converted to a numerical format via one-hot encoding – for each feature, a binary vector, which contains a single `1` and all `0`'s, represents the state for each individual fund. As an example, `Auditor` feature contains 18 values, such as `Deloitte & Touche LLP`, and each fund is mapped to a binary vector, as shown below:

`Auditor`	Funds		
Deloitte & Touche LLP	1	$\cdots$	0
Ernst & Young LLP	0	$\cdots$	0
$\vdots$	$\vdots$	$\ddots$	$\vdots$
Boyle CPA, LLC	0	$\cdots$	1

Table 1. One-hot encoding on `Auditor`

2.3.2 Embedding

A problem one will immediately raise is that the feature space will be huge. There are so many values of `Administrator` and `Auditor` existing in the dataset, and carrying them around and feeding them into models are both computationally and memory-inefficient. To reduce dimension of the huge categorical feature space, we sought collaboration with Embedding 2 group, making use of their pre-processing and embedding techniques. The dimension-reduced categorical data are ready to pass into forecast models.

Now, one will ask: do alternative data *alone* predict excess returns well? Intuitively, the answer is of course negative – a fund administrator remains static over a long period of time but fund performance constantly evolves, dependent on fund manager's control and market conditions but independent of how it is administrated or audited. To confirm this, we utilized (1) regression and (2) FNN to forecast the *mean* excess returns of mutual funds from embedded categorical data (of lower dimension), with the performance on the test dataset illustrated below. As expected, linearity is weak and forecast results are noisy. While we did *not* use the embedded categorical data as inputs to subsequent models, we note that embedding improves model efficiency as a low-dimensional representation is learnt, but risks loss of information. Is embedding effective, or even necessary, for sparse categorical data? – Again, we leave this for future research.

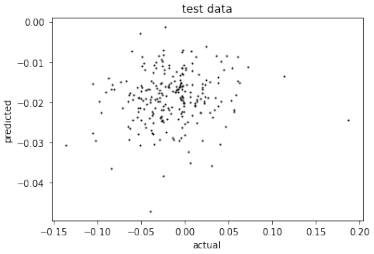

Figure 5: Regression on embedded alt. data

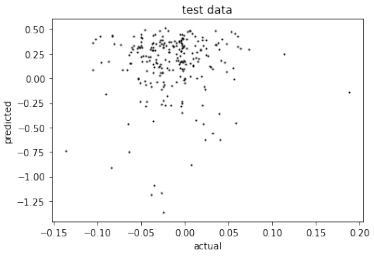

Figure 6: FNN on embedded alt. data

3 Baseline Models

3.1 Data Labeling

Data labeling is the process of adding target attributes to training data and labeling it so that the machine learning model can learn the predictions it is supposed to make. This process is one of the stages of preparing data for *supervised* machine learning.

Since not every mutual fund exists for the entire period spanning `2000` to `2021`, there is a large number of missing values in the time-series `parquet` dataset. `N/A` values are removed, and we note that all (non-`N/A`) numeric values are consecutive for each fund.

A performance measure window of three years is used, meaning we view every three years of excess returns as a cycle. Given this, we remove funds with an excess return history of less than three years.

The dataset is split into a training and

testing set, with the splitting threshold on `2020-06-30`. This follows the conventions of previous work, as in `dev_prediction.ipynb`. Specifically, the sampling parameters applied to `MF_LargeCap_ExcessReturn_3Y.parquet` to generate our training and testing set are as follows:

1. `feat_window = 90`
2. `pm_window = 3`
3. `lb_window = int(3 * pm_window * 365.25) + 1`
4. `sample_window = 30`
5. `test_start_date = '2020-06-30'`

Readers may want to refer to functions `prepare_data()` and `timeseries_dataset()` for its implementation details. Roughly, historical excess returns for each fund are chopped into disjoint timeframes and concatenated into a data matrix.

3.2 `LSTM` for Time-Series

Long Short Term Memory (LSTM) network is a recurrent neural network capable of learning sequenctial dependencies in sequence prediction problems. Unlike standard feedforward neural networks, LSTM has feedback connections, which process not only individual data points but also the entire data sequences. We use a single-layer 200-node LSTM model on time-series alone, which serve as one of our baseline models, against which our later alternative data models are benchmarked.

Let us recall several definitions: (1) Training loss is the error on the training dataset; (2) Validation loss is the error after running the validation dataset through the trained network, usually immediately following an epoch. Here, we adopt log-scale for the losses. As the number of epochs increase, training loss is observed to consistently drop, as with most machine learning training scenarios, while validation loss fluctuates and does not exhibit decreasing trend. While epochs we used are already very small (training is stopped at 10 epochs), it already shows signs of overfitting, seeing that validation loss saturates. Potentially this means the model does not learn much from the data, but only memorizes. This echoes with the known fact that financial time series contain high noise-to-signal ratios.

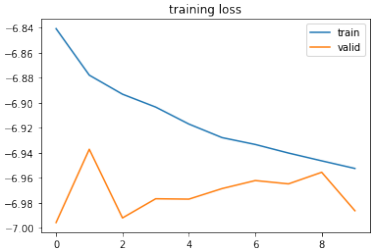

Figure 7: Training loss of LSTM for time-series

While predictions on the training dataset look impressive, the results on the testing dataset show otherwise. There appears to be a slight linear trend, but predictions are capped around 0.025 when actual values take extremely large values at 0.2. In terms of root-mean-sqaured error (RMSE), for the training dataset, RMSE is 0.0306 while for the testing dataset, RMSE is 0.0702, a double. This indicates the model performs much worse out-of-sample, i.e. overfitting. While model architecture is not our focus here (we are sticking to the simplest single-layer 200-node LSTM), future research may look into how number of nodes and layers impact the out-of-sample performance.

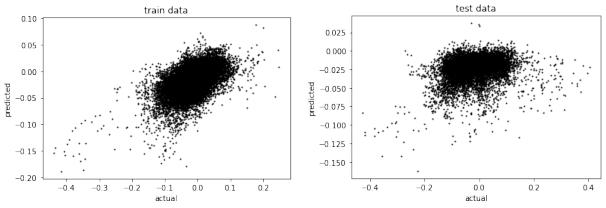

Figure 8: Predictions of LSTM for time-series

3.3 `FNN` for Time-Series

Feedforward Neural Network (FNN) is an artificial neural network where the connections between nodes do not form a cycle, with the goal to approximate some function f^* by defining a map $y = f(x;\theta)$ and learning parameters θ that result in the best function approximation. Learning process is achieved via gradient descent techniques (and many of its variations) and training loss is typified by mean-squared error. As with the LSTM network, here we adopt a 200-node single layer.

As the previous LSTM model, training loss steadily drops but validation loss fluctuates, again indicating signs of overfitting, i.e. the model performs well and appears to memorize the training data, but makes poor predictions out-of-sample.

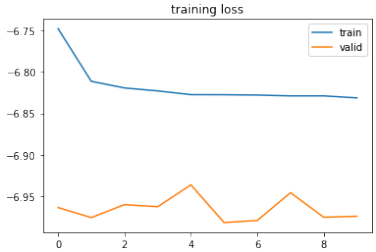

Figure 9: Training loss of FNN for time-series

Making predictions with merely time-series data, prediction results are not satisfactory, as observed from the testing dataset plot, where linearity is nowhere to be seen. Regarding RMSE, for the training dataset, RMSE is 0.0325 while for the testing dataset, RMSE is 0.0690, again double.

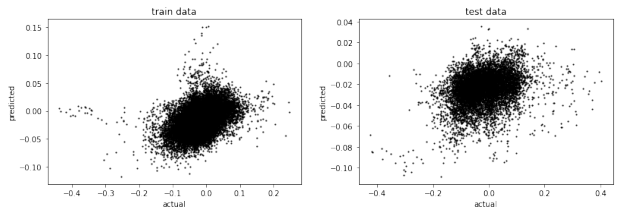

Figure 10: Predictions of FNN for time-series

3.4 Comparison

Feedforward neural networks are primarily used for supervised learning in cases where the data is neither sequential nor time-dependent. It is expected that FNN does not yield an advantage compared to LSTM, due to its incapability to capture covariance structure hidden in the data. In our results, both FNN and LSTM achieve similar out-of-sample RMSE, at approximately 0.07, which is at a scale comparable to the excess return data themselves. Therefore, naively using the two simplest networks for predictions is discouraged. LSTM does not demonstrate a significant advantage, if not performing even wose than FNN. Nonetheless, LSTM offers greater flexibility in terms of model achitecture, and one could easily insert densely connected FNN layers into LSTM. We suggest that future research may investigate whether combinations of the two yield better prediction results.

4 Alt. Data Models

Two models are proposed as follows, in an attempt to combine time-series and alternative data: (1) a simple FNN that merges time-series and alternative data, and (2) a separate architecture with two sub-networks – LSTM for time-series and regression/feedforward network for alternative data. Respectively, we denote them **`AltFNN`** and **`LSTM-AltReg/FNN`**. Compared to our baselines, both achieve better prediction results, either graphically or via mean-squared error. Below, we discuss their associated rationales and the prediction outcomes. Regarding network architectures, all LSTM/FNNs in this section contain a 200-node single layer, and our aim here is to shape the prediction frameworks, rather than optimal network structures.

4.1 **`AltFNN`** Model

`AltFNN` concatenates the alternative features to time-series data, and feeds them altogether into an FNN network. More explicitly, the data construction goes as follows: for each mutual fund, (1) split the historical excess return time series into disjoint timeframes of d_1 days, according to parameters specified in **Data Labeling**; (2) attach to each timeframe *identical* one-hot encoded categorical features of the fund, of dimension d_2. As they are concatenated, the input matrix into FNN carries a dimension of $N \times (d_1 + d_2)$. In our case, $d_1 = 37$ days and $d_2 = 3325$ is the sum of the numbers of existing values in all categorical features, including `Administrator`, `Auditor` and so on, and finally $N = 47052$. We split training and testing set by a ratio of $0.7 : 0.3$. Reader may refer to function `timeseries_alt_dataset()` for the concatenation procedure.

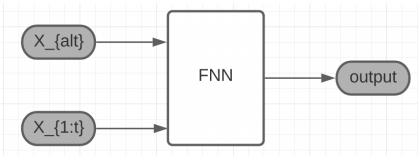

Figure 11: `AltFNN` model schematics

Implementing the model, training loss drops significantly initially but the decrease is less obvious ensuing epoch 1. On the contrary, validation loss remains steady over all epochs. Overfitting might be less serious in this case, compared to our training of LSTM and FNN on time-series, since both training and validation loss saturate when the training process terminates.

Figure 12: Training loss of `AltFNN`

Referencing our baselines, `AltFNN` performs significantly better: when making predictions on the testing dataset, an obvious linearity is seen, and the RMSEs for the training and testing dataset are respectively 0.0255 and 0.0606. Recall that our baselines achieved an RMSE of 0.07 on the testing dataset, and the inclusion of alternative data reduces it by 0.01. This is not unexpected because alternative data provide additional information on the fund identity. For example, suppose fund manager A historically produces several funds that outperform the market. Since the fund managers information is supplemented into `AltFNN`, the network knows to lift up the forecast by a certain amount once it encounters a binary `1` in our one-hot categorical inputs.

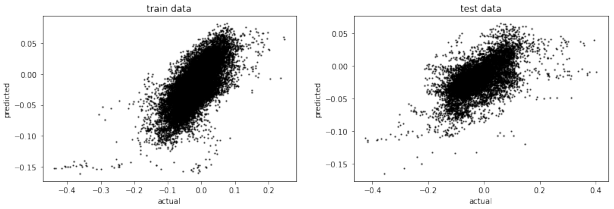

Figure 13: Predictions of `AltFNN`

4.2 `LSTM-AltReg` Model

`LSTM-AltReg` extends our baseline LSTM model, by appending to the network a regression predictor y_1 for *correcting* the LSTM forecasts y_2, via a linear weight adjustment $w \in [0, 1]$, with the final output given by $(1 - w)y_1 + wy_2$. Here, we treat time-series and alternative data on equal footing, for ease of implementation. Should user desire time-series data carry a more important role, a constraint on w may be applied, for instance, $w \in [0.5, 1]$. For the regression, the *mean* historical excess returns are regressed against the one-hot encoded categorical features, of dimension $d = 3325$. Each fund, carrying its idiosyncratic alternative features, therefore gets mapped to a single *mean* excess return. Note that such sparse regression can be dangerous, since the number of samples N, in our case, number of mutual funds, is comparable to the number of one-hot features d. A consequence is that t-statistics of the regression coefficients may be small, and forecasts can carry large error margins. Future research may look into oversampling techniques, such as SMOTE, to artificially produce more samples to improve statistical significance of coefficients. Regarding the last combining component of `LSTM-AltReg`, we resort to the simplest solution – linear combination of y_1 and y_2. While the results demonstrated by this method are already very promising, we suggest future work looks into possibilities of replacing it with FNN or more sophisticated structures. Below, we illustrate the schematics of `LSTM-AltReg`.

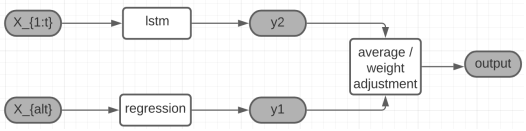

Figure 14: `LSTM-AltReg` model schematics

We sought the optimal combination of y_1 and y_2 based on the training dataset, by minimizing the loss $L = 1/2N \sum(\hat{y} - y)^2$, where $\hat{y} = (1 - w)y_1 + wy_2$ is the averaged output and y is the training label. From the training dataset, we obtained $w = 0.22$ (as shown on the right), and for comparison, we also looked at $w = 0.50$, i.e. a simple average of y_1 and y_2 (as shown on the left). Unexpected as it is, w is calculated to be < 0.5, meaning alternative data carries more weight than time-series. This runs counter to our intuition – alternative data provide information only on the mean, or expected, future returns, while time-series offer historical trends, so one should expect time-series to be more informative in terms of predictions. This appears to echo with the Markovian nature of efficient markets – given the current state, one can basically forget about the entire history to make predictions. We suggest that future research delves into the reasons that $w < 0.5$. For the statistics, $w = 0.50$ gives an RMSE of 0.0601 while $w = 0.22$ gives an RMSE of 0.0591, only a slight improvement over the former.

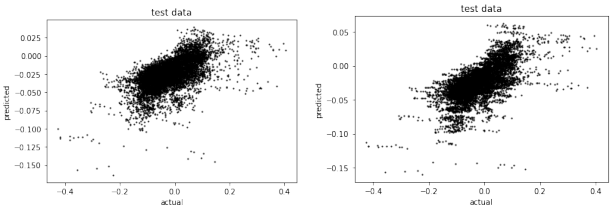

Figure 15: Predictions of `LSTM-AltReg`: $w = 0.50$ and $w = 0.22$

4.3 `LSTM-AltFNN` Model

Now, we slightly modify `LSTM-AltReg` – for alternative data, we replace the regression predictor with an `AltFNN`, our first model, hence the name **`LSTM-AltFNN`**. Similarly, we combine the two forecasts y_1 and y_2 by a linear weight w. Note that we have a duplication of time-series data here, in order to address the previous observation that time-series carry less important weight than alternative data.

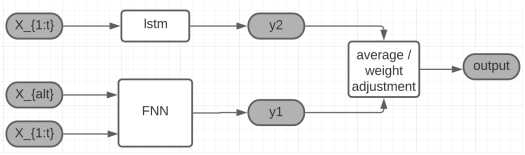

Figure 16: `LSTM-AltFNN` model schematics

Prediction results exhibit a clean linearity, meaning the model predictions correlate with the actual values. For the statistics, training dataset gives an RMSE of 0.0230 while the testing dataset gives an RMSE of 0.0575. This model returns the lowest out-of-sample RMSE among all models attempted, which nicely concludes our project.

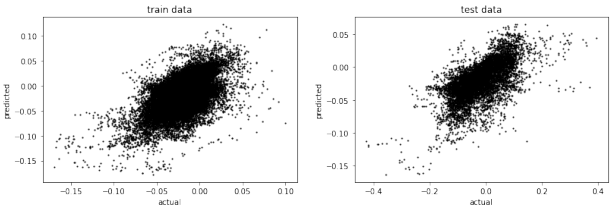

Figure 17: Predictions of `LSTM-AltFNN`

5 Conclusion

In this project, we combined the time-series data and alternative data to predict the future performance of mutual funds and proposed two prediction frameworks: (1) `AltFNN` and (2) `LSTM-AltReg/FNN`. Both frameworks outperform out baselines: `LSTM/FNN` applied to time-series data alone. In particular, **`LSTM-AltFNN`** gives the most remarkable performance, returning an out-of-sample RMSE of 0.0575, lowest among all models. We believe that alternative data do offer additional values that investment managers can seize on, and are therefore a nice supplement to the traditional technical forecasts based on time-series, numeric data alone. Our project is in no way stopped here. Below, we discuss potential research directions and extensions to our models future groups can delve into.

6 Discussion

Alternative data are new perspectives to interpret financial performances. While this project comes to the conclusion that **`LSTM-AltFNN`** effectively merges time-series and alternative data, outperforming our time-series model baselines, several interesting (side) research directions arise in the meantime. Below, we provide directions that future research groups can look into:

1. (Fund-specific models) In our training of the fund performance forecast models, we assume that funds are homogeneous and they are fed into the same model – by first chopping their historical excess returns into appropriate timeframes and then feeding into either LSTM/FNN predictor. This assumption follows previous group's practice, and is understandable given that we interpret funds are drawn from a universal sample space. But models trained this way may not be able to capture idiosyncratic features of individual funds – will separate models work better? Can we first categorize, via clustering, the funds into groups, and train models on those groups?
2. (Embedded alternative data) We trained our alt. data models with one-hot encoding, resulting in a large number of features, which is both computationally and memory-inefficient. Will predictions based on the embedded counterparts work better? Or will encoding sparse matrix (in our case, sparse one-hot binary matrix) cause loss of information, hence hindering performance?
3. (Modifying `LSTM-AltReg/FNN`) Our work focuses on proposing frameworks, rather than optimal model architectures, so throughout the whole research, we stick with the simplest models, for example, single-layer 200-node LSTM for time series, simple OLS regression for one-hot alt. data. Will sophisticated, optimized model architectures significantly improve our model forecasts? Besides, regression on one-hot alt. data is statistically dangerous, since samples do not outnumber features, creating insignificant, unstable coefficients. Will over-sampling techniques, such as SMOTE, described in Hirsa's paper, help?

7 Statement of Contributions

All works in this project, including data analysis, model development, report writing and result presentation, are equally split between us. We thank Professor Hirsa, Advisor Malhotra from Ask2.ai and teaching assistants Miao, Sikun and Federico for their valuable advice and assistance over the semester.

References

[1] Goodfellow et al., *Deep Learning*, MIT Press, 2016.

[2] Hirsa et al., *Predicting Status of Pre and Post M&A Deals Using Machine Learning and Deep Learning Techniques*, 2021.