A Sales Prediction Method Based on LSTM with Hyper-Parameter Search

Yun Dai1,* and Jinghao Huang2

1 Hitachi(China) R&D Corporation Guangzhou Branch, Guangzhou, China
2 School of Mathematical Science, South China Normal University, Guangzhou, China
* Corresponding author: ydai@hitachi.cn

Abstract. Sales forecast is a significant topic in business operation, which generally formulated as a time-series regression problem. Although there are many research results in this field, we still face some challenges in real scenes, such as data with high-sparsity, users may have a preference in prediction results, and systems need a single model with high performance. In this paper, a method is proposed to address the above challenges. We present a long short-time memory (LSTM) model with a special loss function and use the hyper-parameter search for accuracy optimization. To illustrate the performance, we employ them on the open dataset, Kaggle Rossman sales data. The experiment results show that compare with a series of machine learning models using the AutoML (Auto Machine Learning) tool, the proposed method significantly increased the performance of prediction on sparse data. Besides, it can reasonably overestimate or underestimate sales forecasts based on user preferences that meet the actual business demands.

1. Introduction
In business operations, sales forecast is an important issue, which will affect various links such as merchants' replenishment, promotion and production.

1.1 Related work
The existing sales forecast methods focus on three categories: statistical ways, traditional machine learning, and deep learning.

Companies mainly use statistical ways, such as the expert inquiry method [1] and time-series related algorithms. The expert inquiry method completely relies on human experience and the accuracy is not stable enough. The time-series algorithms including auto-regression [1], exponential smoothing method [1], and ARIMA model [2], which are using historical sales data to construct the model. These methods cannot make full use of related factors, for example, price, holiday, etc. so that hard to guarantee prediction accuracy in complex changes.

Various traditional machine learning (ML) techniques, such as random forest [3], support vector machine[4], XGBoost [5] are introduced into the field of sales forecast. The above methods comprehensively use factors related to sales and improve the prediction accuracy. However, these models cannot directly process time-series data, nor can they well extract the hidden rules of the data.

More recently, deep learning techniques, such as CNN [6] and RNN [7] have also shown to be competitive in this domain. And, LSTM [8] is superior to other methods in prediction accuracy.

Meanwhile, some scholars assemble these three types of models and obtain impressive results [9].
1.2 Problems in real scenes
Back to our challenges in sales prediction, we will face three problems: sparse data, user preference, a single model with good performance.

Sparse data appears as:
(1). Among all the products, only a small part belongs to the best-selling with sales records every day. Most products only have sales records in certain periods and none in other periods.
(2). The smaller the division dimension of the products, the more obvious the sparseness of the data. But we need smaller granularity data because it is accurate for decision-making of daily operation.

Hence, with the change of market strategy, users may have preferences for prediction results. For example, when the warehousing cost is less than the out-of-stock cost, appropriate overestimate the prediction is beneficial for promptly restock.

And then, in a real prediction system, although the ensemble model is effective, it is not easy to maintain. A single model with good performance is more valuable.

However, few experts and scholars consider these limits during the sales forecast.

1.3 Proposed method
We present an LSTM model with a special loss function, and the hyper-parameter search is used for optimization. Then, set six types of machine learning models with AutoML as the baseline, compare the performance of two different LSTM with or without the hyper-parameter search. We also sample a set of sparse data from Kaggle Rossman open dataset to verify the performance of different models. The details of the proposed method are explained in the following sections.

2. Models Structure
First, we would like to introduce the data structure of the model. The data will be divided into a training set and a test set according to a certain time node. And then, use the time sliding window to construct a sample that fits the LSTM network. Let the sales at time $T$ be $Y_i$, and the related variables at time $T$ are $X_i = (X_{i,1}, X_{i,2}, ..., X_{i,n}, Y_i)$, where $X_{i,1}, X_{i,2}, ..., X_{i,n}$ are features related to sales, such as information of product, store, external, etc. The sample can be expressed as $X_1, X_2, ..., X_n \rightarrow (Y_{n+1}, Y_{n+2}, ..., Y_{n+K})$, that is to say, use the first $N$ moments to predict the future sales at $K$ moments.

The details of the three models are described below.

2.1 AutoML
In response to this regression problem, we selected five machine learning baseline models and assemble them into a stacking model.

KNN is a representative model for metric learning. The theoretical basis is that the more similar samples have similar properties. It finds $K$ nearest neighbours of the sample to be predicted and uses the average value of these neighbours as the sales forecast result.

Random forest, CatBoost, XGBoost, and Lightgbm models are all representative solutions for regression problems, and they achieve prediction by constructing one or more decision trees that have a very impressive performance in many data competitions.

Because the above model cannot achieve multiple outputs like the neural network model, we adopt multiple models, for example, the first model predicts future sales in the first week, and the second model predicts sales in the second week, and so on.

Hence, feature processing and hyper-parameters are critical to the performance of the model. Usually, these two steps need to manually adjusted, but with the development of AutoML, they can already be done automatically. Some scholars used this method and got a better effect than manual adjusted[10]. In this paper, the AutoGluon tool developed by Amazon[11] will be used for data conversion, feature processing, and optimal hyper-parameters search, which helps machine learning models find the hidden connection of the data.

2.2 LSTM
LSTM is an extension of RNN that can learn long-term dependencies in the sequence. The basic unit is composed of the input gate, forget gate, and output gate, and specific process is shown in figure 1:

![Figure 1. Structure of LSTM network unit][8]

We design a new loss function. Because the loss function commonly used in LSTM has equal weight to each prediction, but according to users' preference, appropriately underestimating or overestimating predicted values can better meet business strategies. For example, if the predicted value is small, the shop could not restock in time which causing customers to complain. If the predicted value is large, it increases the storage cost. When storage cost is less than out-of-stock cost, appropriately increasing the predicted value is a benefit to the whole business operation. The loss function shows as:

\[
\text{loss}(y_{\text{truth}}, y_{\text{pred}}) = \begin{cases} 
  w (y_{\text{truth}} - y_{\text{pred}})^2, & y_{\text{truth}} > y_{\text{pred}} \\
  (y_{\text{truth}} - y_{\text{pred}})^2, & y_{\text{truth}} \leq y_{\text{pred}}
\end{cases}
\]

The loss function can adjust by weight w, the greater the w, the predicted result is overestimated, but it still based on the law of ordinary predicted value. Then, since the sales for merchandise cannot less than 0, we choose ReLU \((x) = \max(x, 0)\) as activation function. The Adam optimization method is used for faster convergence speed.

### 2.3 LSTM + Hyper-parameter search

Hyper-parameter search is an effective method for deep learning automatically adjust parameters in LSTM [12]. Hyper-parameters include the number of cyclic layer units and dropouts, etc. If we use this method to all the data, it is unbearable in time, so we draw on the idea of the greedy strategy and proposes the following methods:

a. Determine all hyper-parameters in the network.
b. Sample a small size data for grid search, and obtain the optimal N sets of hyper-parameters.
c. Put the obtained N sets of hyper-parameters into the full dataset, and train to get the optimal model.

This method sample a representative training dataset uses a large-scale search in a small data set and then uses the full amount of data for fine-tuning, which helps balance the cost of accuracy and time.

### 3. Simulation Result and Evaluation

#### 3.1 Dataset

We conduct experiments on the dataset, Kaggle Rossmann dataset, which consisting of sales and store information from 2013 to 2015 including 1115 chain stores. For addressing the challenges, we extracted a sparse dataset. The record of the whole of 2013 is set as a training set, and the data from January 1, 2014, to June 30, 2014, is set as a test set. To meet actual needs, we will make weekly and monthly forecasts. The statistical information of datasets is detailed in Table 1.

| Dataset             | Sample  | Store | Raw feature |
|---------------------|---------|-------|-------------|
| Kaggle Rossmann     | 1017209 | 1115  | 27          |
| Extracted data      | 456147  | 500   | 27          |
3.2 Data Processing
Features are constructed based on raw data, as shown in Table 2, and the week/store id and month id/store id are used as the primary key to aggregate and construct the training sample. Simultaneously, the sales is logarithmically transformed into $\hat{y} = \log(y + 1)$, $y$ and $\hat{y}$ are the sales before and after the transformation, respectively. The categorical variables such as store type are transformed by one-hot encoding. Then, adjust the sample for data balance. We define a sample whose sales is not zero within a period is a positive sample, otherwise, it is a negative sample. For example, set the next four weeks as the prediction target, if stores' sales are zero in all four weeks, the sample will be considered as a negative sample, others are positive samples. We set the positive and negative samples according to 1:1 to ensure the balance of the dataset.

### Table 2. Features used in the experiment

| Category            | Features                                                                 |
|---------------------|--------------------------------------------------------------------------|
| Sales information   | Weekly/monthly: total sales volume, average / max / min traffic volume, number of days with weekly sales not 0 |
| Store information   | Whether to promote this week, number of store closures this week          |
| External information| Weekly/monthly: the number of working days/weekends/public holidays/special promotional days |

3.3 Models’ setup
For AutoML models, we will use the AutoGluon library for training. For LSTM models, Keras will be used, and the main experimental parameters of forecast models are shown in Table 3. We also try the hyper-parameter search to automatically set the parameters. The model uses historical data for the first 8 weeks to predict sales for the next 4 weeks. The monthly forecast uses historical data from the first 5 months to forecast sales in the next 3 months.

### Table 3. Main experimental parameters of the LSTM model

| Parameters                                   | Values | Parameters     | Values |
|----------------------------------------------|--------|----------------|--------|
| Number of LSTM units                        | 256    | Weight of loss function | 5      |
| Number of neurons in the fully connected layer | 64     | Learning rate  | 0.001  |
| Number of neurons in the output layer       | 4      | Batch size     | 32     |
|                                              |        | Epoch          | 15     |

3.4 Model Evaluation
To evaluate the prediction accuracy, the RMSPE (Root Mean Square Percentage Error) and MAPE (Mean Absolute Percentage Error) are computed against the testing data, which are calculated as follows:

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i}, \quad \text{RMSPE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}$$

where $n$ is the number of samples, $y_i$ is the real sales of sample $i$. When $y_i$ is zero, the denominator will be set as $y_i + 1$. $\hat{y}_i$ is the predicted sales of sample $i$.

3.5 Experiment result
After carrying out the models on the dataset, the results of the weekly prediction are shown in Table 4. We can conclude that:

1. In AutoML models, the prediction effect of the KNN and random forest is the worst. Relatively, CatBoost \ XGBoost \ LightGBM has significantly improved the performance, but the stacking ensemble of all models does not contribute to better results.

2. Compared with AutoML models, the LSTM model has better performance on prediction. The best MAPE and RMSPE appeared in the first-week prediction, reaching 0.043 and 0.056, respectively.

3. Compared with the ordinary LSTM model, the prediction results of the first, third, and fourth weeks have significantly improved after the addition of hyperparameter search. The best RMSE and RMSPE
results appear in the first-week prediction, which has MAPE lift 79% and RMSPE lift 75% than ordinary LSTM model.

**Table 4. Result of weekly predictions of different models**

| Models      | First week | Second week | Third week | Fourth week |
|-------------|------------|-------------|------------|-------------|
|             | MAPE | RMSPE | MAPE | RMSPE | MAPE | RMSPE | MAPE | RMSPE |
| KNN         | 3.281 | 4.252 | 3.624 | 4.812 | 3.823 | 4.568 | 2.912 | 3.253 |
| Random Forest | 4.235 | 3.950 | 3.598 | 4.132 | 3.722 | 4.232 | 3.974 | 4.135 |
| AutoML      |          |          |          |          |          |          |          |          |
| CatBoost    | 0.392 | 0.473 | 0.741 | 0.987 | 0.823 | 0.971 | 0.412 | 0.435 |
| XGBoost     | 0.353 | 0.413 | 0.212 | 0.133 | 0.498 | 0.593 | 0.653 | 0.857 |
| LightBGM    | 0.217 | 0.438 | 0.587 | 0.633 | 0.227 | 0.396 | 0.261 | 0.356 |
| Stacking    | 0.302 | 0.343 | 0.498 | 0.689 | 0.579 | 0.987 | 0.210 | 0.341 |
| LSTM        | 0.043 | 0.056 | 0.088 | 0.178 | 0.184 | 0.133 | 0.594 | 0.727 |
| LSTM+ Hyper-Parameter Search | **0.024** | **0.032** | **0.125** | **0.183** | **0.035** | **0.087** | **0.327** | **0.382** |

We extracted the predicted values of four stores in the same week to verify the influence of the loss function weight on the predicted values. Table 5 shows the corresponding results. We find that the predicted value increases as w increases. When w=5, the result is close to the true value which is the prediction without preference. When w=3 and w=7, it is the prediction of underestimation and overestimation, respectively. Users can adjust the weight according to the actual demand.

**Table 5. The influence of different weights on the prediction result**

| Samples | w=3 | w=5 | w=7 | True value |
|---------|-----|-----|-----|------------|
| 1       | 0   | 1   | 5   | 0          |
| 2       | 3   | 6   | 9   | 7          |
| 3       | 3364 | 3687 | 4953 | 3847      |
| 4       | 7913 | 8397 | 10156 | 8301      |

To verify the prediction ability of the LSTM model with hyper-parameter search at different time granularities, we conducted a monthly sales volume prediction, the results are shown in Table 6. We found that the error value in the weekly forecast is smaller, and the closer the time, the more accurate the forecast, but this trend is not obvious in the monthly forecast. Overall, the model has achieved good results in different time granularities.

**Table 6. Different period prediction result using LSTM + hyper-parameter search**

| Model              | Period       | MAPE | RMSPE | Period       | MAPE | RMSPE |
|--------------------|--------------|------|-------|--------------|------|-------|
| LSTM+ Hyper-        | First week   | 0.043 | 0.056 | First month  | 0.321 | 0.343 |
| parameter search    | Second week  | 0.088 | 0.178 | Second month | 0.273 | 0.332 |
|                    | Third week   | 0.184 | 0.133 | Third month  | 0.351 | 0.438 |
|                    | Fourth week  | 0.594 | 0.727 |              |      |       |

4. Discussion
For model effects, besides the difference in RMSPE and MAPE of three models, we find the diversity of results predicted by the AutoML is lower than that of the LSTM with or without hyper-parameter search, that is to say, the output of AutoML is concentrated on only a few values, and LSTM has a stronger fitting ability. We use standard deviation to characterize the degree of dispersion of the data. The results recorded in Table 7 also confirm our observations.
The possible reason for the above is that the structure of traditional machine learning is relatively simple, which unable to fit complex changes. Besides, they cannot directly handle time-series data and only can build multiple models and predict one week by one week, resulting in the loss of information.

In terms of the loss function, adjust $w$ can appropriately overestimate or underestimate the basis of ordinary prediction which meets customers’ needs.

For the prediction granularity, the LSTM with hyper-parameter search can handle different time granularities with relatively high accuracy. One thing worth mentioning is that in the experiment of this article, the smallest predicted dimension is sales per store/week. However, in an actual scenario, much smaller granularity prediction is common, such as predicting per SKU (Stock Keep Unit)/store/time. As the forecast granularity becomes smaller, the sparsity of data will increase. We used an internal dataset to make a fine-grained sales prediction, in which the dimension is per SKU/store/day, and the result still shows good performance.

5. Conclusion
In this paper, the LSTM model with a special loss function and hyper-parameter search is proposed for overcoming challenges in the real-scene sales forecast. By comparing with traditional methods, we find that the proposed model is superior to others, which provide reliable predictions.

The flaw of this model is that it cannot solve the cold start problem of new product prediction. How to modify the model so that to apply new products is our direction in the future.

References
[1] Chang P C, Wang Y W. Fuzzy Delphi and back-propagation model for sales forecasting in PCB industry[J]. Expert systems with applications, 2006, 30(4): 715-726.
[2] Xia M, Wong W K. A seasonal discrete grey forecasting model for fashion retailing[J]. Knowledge-Based Systems, 2014, 57: 119-126.
[3] Vairagade N, Logofatu D, Leon F, et al. Demand Forecasting Using Random Forest and Artificial Neural Network for Supply Chain Management[C]//International Conference on Computational Collective Intelligence. Springer, Cham, 2019: 328-339.
[4] Zhu Y, Zhao Y, Zhang J, et al. Spring onion seed demand forecasting using a hybrid Holt-Winters and support vector machine model[J]. PloS one, 2019, 14(7).
[5] Ji S, Wang X, Zhao W, et al. An Application of a Three-Stage XGBoost-Based Model to Sales Forecasting of a Cross-Border E-Commerce Enterprise[J]. Mathematical Problems in Engineering, 2019, 2019.
[6] Livieris I E, Kiriazidou N, Kanavos A, et al. Employing Constrained Neural Networks for Forecasting New Product’s Sales Increase[C]//IFIP International Conference on Artificial Intelligence Applications and Innovations. Springer, Cham, 2019: 161-172.
[7] Hewamalage H, Bergmeir C, Bandara K. Recurrent neural networks for time series forecasting: Current status and future directions[J]. arXiv preprint arXiv:1909.00590, 2019.
[8] Bandara K, Shi P, Bergmeir C, et al. Sales demand forecast in e-commerce using a long short-term memory neural network methodology[C]//International Conference on Neural Information Processing. Springer, Cham, 2019: 462-474.
[9] Pemathilake R G H, Karunathilake S P, Shamal J L A J, et al. Sales forecasting based on autoregressive integrated moving average and recurrent neural network hybrid model[C]//2018
14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). IEEE, 2018: 27-33.

[10] He Y, Lin J, Liu Z, et al. Amc: Automl for model compression and acceleration on mobile devices[C]//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 784-800.

[11] Erickson N, Mueller J, Shirkov A, et al. AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data[J]. arXiv preprint arXiv:2003.06505, 2020.

[12] Bergstra J, Bengio Y. Random search for hyper-parameter optimization[J]. Journal of machine learning research, 2012, 13(Feb): 281-305.