Machine Learning Nanodegree
Capstone Proposal

Botao Deng

January 16, 2018

1 Domain Background

Sales forecasting is a key element in conducting business.[1] Good forecasting guarantees that sufficient products will be manufactured or delivered to customers on a timely basis, resulting in happier customers and fewer complaints. It also prepared the company with good management of inventory, avoiding both overstock and stock-out situations. If the company can predict demand and manage production more efficiently, it also have better control over the supply chain. This affords the company the opportunities to fully manage resources.

Time series forecasting is a known approach to solve the sales forecasting problem. Experience from the past is used to predict what will happen in the future. In the field of deep learning, there is one algorithm, LSTM, that is known to be exceptionally good at finding long-term relationship hidden in the data. And it has already been shown to be good at handling sale forecasting problem.[2] Compared with traditional recurrent neural network, it maintain a mechanism that hold on to the memory of its previous states, and during each stage, it selectively added information from its memory with an input gate and forget gate. An decision is made at the end.

At this project, we are going to build various recurrent neural networks, based on LSTM cell, to solve the sale forecasting problem introduced by Kaggle, "Corporación Favorita Grocery Sales Forecasting".

2 Problem Statement

This problem comes from one of the featured Kaggle challenge "Corporación Favorita Grocery Sales Forecasting", the following description comes from the kaggle challenge home page[1].

"Brick-and-mortar grocery stores are always in a delicate dance with purchasing and sales forecasting. Predict a little over, and grocers are stuck with overstocked, perishable goods. Guess a little under, and popular items quickly sell out, leaving money on the table and customers fuming.

The problem becomes more complex as retailers add new locations with unique needs, new products, ever transitioning seasonal tastes, and unpredictable product marketing. Corporación Favorita, a large Ecuadorian-based grocery retailer, knows this all too well. They operate hundreds of supermarkets, with over 200,000 different products on their shelves.

Corporación Favorita has challenged the Kaggle community to build a model that more accurately forecasts product sales. They currently rely on subjective forecasting methods with very little data to back them up and very little automation to execute plans. They're excited to see how machine learning could better ensure they please customers by having just enough of the right products at the right time."

To be specific, we are going to predict the short-term daily sales of products of Corporación Favorita from Aug.15, 2017 to Aug.31, 2017. And the model would be train on the sales information from 2013 to Aug.15,2017, and other relevant information such as oil price, product type, store type.

[1] https://www.kaggle.com/c/favorita-grocery-sales-forecasting

This problem is a regression forecasting problem, and it can be tackled using with time-series or non time-series model. For a time series model, it is designed to catch the casual dynamic relationship over time and infer the sales of products in the future. For a non time-series model, the sales, promotion, date information are treated as independent features, and the model would be designed to infer sales based on the universal information of a product.

3 Datasets and Inputs

The data is available on the competition website, and it has been split into training and testing set. The training set, contains transactions data from 2013 to 2017.8.15, about one hundred and twenty five million transaction records. While the testing set has the transaction records from 2017.8.15 to 2017.8.31. Our ultimate goal, is to learn from the history, and to predict the sales data of the testing set. But during training, we need to further split the training set into train, validation and test set. Because the original test set provided does not include target label, therefore it could not be used during the training phase.

The following file description comes from the Kaggle competition website.[2]:

- **train.csv:** Training data, which includes the target unit_sales by date, store_nbr, and item_nbr and a unique id to label rows. The onpromotion column tells whether that item_nbr was on promotion for a specified date and store_nbr.
- **test.csv:** Test data, with the date, store_nbr, item_nbr combinations that are to be predicted, along with the onpromotion information.
- **stores.csv:** Store metadata, including city, state, type, and cluster. Cluster is a grouping of similar stores.
- **items.csv:** Item metadata, including family, class, and perishable. Items marked as perishable have a score weight of 1.25; otherwise, the weight is 1.0.
- **transactions.csv:** The count of sales transactions for each date, store_nbr combination. Only included for the training data timeframe.
- **oil.csv:** Daily oil price. Includes values during both the train and test data timeframe. (Ecuador is an oil-dependent country and it's economical health is highly vulnerable to shocks in oil prices.)
- **holidays_events.csv:** Holidays and Events, with metadata.

In this project, we will utilize the store_nbr, item_nbr and date features as indexes to organize our dataset. And during training, we mainly use unit_sales and promotion, and other crafted features from daily unit_sales to construct our model.

4 Solution Statement

This problem is a supervised learning problem and the dataset provided could be reconstructed into a time-series format. To be specific, the original data are ordered by transaction id. Each transaction has its own unique id, and there are about one hundred and twenty five million rows, so there are about one hundred and twenty five transactions overall. Each transaction has its own store, date, etc.

Since the data could be treated as time-series data at its nature, at the first step I plan to reconstruct it into a new dataframe, where each row corresponds to a (store, item) pair, and each columns will be a date. By doing this, each row represents exactly the transaction record of one item at a specific store.

[2] https://www.kaggle.com/c/favorita-grocery-sales-forecasting/data

After we got this matrix, we could feed samples from the matrix batch by batch, along with other features,so that the LSTM could learn from the time series data of different (store, item) pair, and predict its sales in the future.

Another approach I'm going to take, is to do feature engineer of the raw time series data. Constructing from the sales data to various features such as "What's the total sales of last week", "what's the sales of the same day last week", "what's the last time this item get sold at this store", etc. And we feed those features of a (store, item) pair into LSTM at a sequential manner and let the LSTM to learn from these prepared features instead of the raw daily sales, which would be very noise when the time range is large.

5 Benchmark Model

There are two benchmark models I plan to use here.

- The first model used the first approach as discussed above. But it only used the sales data, without any catagorical data or features from any supplementary dataset. In this model, I randomly sampled time series from the whole data set, and feed the sampled time series into LSTM together. After training for 5 epochs on Kaggle's server on Jan.10, it reaches 0.553 Normalized Weighted Root Mean Squared Logarithmic Error, which was ranked around 1100 from 1632 teams.
- The second model comes from the public starter code[3]. In this model, unlike the previous one where we used raw daily sales data, it added features such as "mean sales of previous 3/7/14/... days", "number of promotion in the last 14/60/140 days", "same weekday's sales of previous/next week'. And the crafted features are fed into LSTM network. After training for 5 epochs, it reached 0.518 Normalized Weighted Root Mean Squared Logarithmic Error, and that ranked around 715 out of 1632 teams. We intend to add more intuitive and complexity features in our final model.

6 Evaluation Metrics

The evaluation metrics we uses in this problem is Normalized Weighted Root Mean Squared Logarithmic Error. (NWRMSLE)

$$E = \sqrt{\frac{\sum_i^n w_i \sum_{d=1}^{16} (log(y_{i,d}+1) - log(\tilde{y}_{i,d}+1)^2}{\sum_i w_i}} \tag{1}$$

Whereas n is the number of (store, item) pair, specifying the exact item being sold at a specific store. d referring to the date that the model will predict. There are 16 days in the test set, therefore d ranges from 0 to 16. The weight w_i is generated depending on whether the item is 'perishable' or not. This is a meta data information from the file 'items.csv'. $\tilde{y}$ is the predicted sales of a (store, item) pair, and y is the ground truth sale of that pair.

7 Project Design

7.1 Programming Language and Libraries

- **Python 3.**
- **scikit-learn**. Open source machine learning library for Python.
- **Keras.** Open source neural network library written in Python.
- **plotly.** Open source software for interactive data visualization.

[3] https://www.kaggle.com/senkin13/lstm-starter

- **pandas.** Open source software for data mining.
- **feather.** Open source software for fast saving and loading dataframe.

7.2 Environments

This project will use two environments:

- **Kaggle's kernel.** This is used for testing code on small samples. It has 4 CPUs 16GB RAM.
- **AWS EC2 p2.xlarge instance.** This is used for training model on the large dataset. It has 1 Tesla K80 GPU, whose RAM is 12GB. And it also has 2496 parallel processing cores, with 61 GB memory.

7.3 Workflow of project

- **Memory Optimization and fast data retrieval.** It's a relative large dataset with a hundred and twenty five million records. It would take forever to loaded it into memory, and it uses up memory fast. After that, we need to find a way to save those data so that we don't need to do memory optimization every time we run the scripts.
- **Data Exploration.** Visualized daily sales, monthly sales to get a rough intuition of the distribution of the data. And we also need to visualize the supplementary data such as daily oil price, different cluster of stores, etc.
- **Data Preprocessing.** This step we have two main parts 1) Manipulate the dataset into a time-series manner. 2) Encode categorical features.
- **Feature Engineering.** Crafted the features in a more concise manner, the original daily sales data is really noisy.
- **Building, training LSTM with raw daily sales data.** Adding other categorical features such as store type, store cluster, item type to our final model.
- **Building, training LSTM with engineered features.** Adding more complex and intuitive features to final model, such as "what's the mean/median/max/mean sales in the last 3/7/14/... days", "when is the last time/first time in the last 3/14/30/... days that this item has sales", "When is the last/first time that this item has promostion", etc.
- **Tuning LSTM.** Explore different architectures of LSTM, tuning hyper parameters until it reaches optimal performance.

References

[1] Beheshti-Kashi, S., Karimi, H. R., Thoben, K.-D., Lütjen, M., and Teucke, M. A survey on retail sales forecasting and prediction in fashion markets. *Systems Science & Control Engineering 3*, 1 (2015), 154–161.

[2] Tyrpáková, N. Deep neural networks for sales forecasting.