A Generalizable Sales Forecasting Pipeline with Embedding and Residual Networks

Alexia Wenxin Xu
alexiaxu@stanford.edu

Abstract

Many industries rely on precise sales forecasting to manage production and avoid unforeseen cash flow problems. While statistical models like moving average are reasonably effective, they require heavy feature engineerings, which make those fine-tuned models not generalizable among different industries. In this project, we hope to develop a feature-engineering free pipeline to predict unit sales, such that the pipeline could be used in different fields. To explicit the concept, we trained embedded features and residual networks to predict unit sales in grocery stores. We improved the performance significantly comparing to the baseline.

1. Introduction

Companies in various industries rely on sales forecasting to predict achievable sales revenue, efficiently allocate resources and plan for future growth. Most forecasting systems in-service now are based on statistical learning methods, such as moving average, and regression. Both methods have the advantage of easy to understand and implement, and are reasonably effective in most cases. In addition, moving average is able to capture the long-term trends, which makes it the first choice to predict time series data. However, all these methods require heavy and delicate feature engineering. It requires human knowledge to identify all possibly related features and find out the best way to incorporate them into the models. Moreover, these models are not likely to generalize between industries. Thus it will be beneficial to build a generalizable pipeline for sales forecasting that does not require task-specific feature engineering.

Among all industries, grocery stores are probably the most eager to have such a model since they bear the brunt of imprecise predictions. Overestimation of the sales makes them stuck with overstocked, perishable goods, and underestimation leaves money on the table and customers fuming. In this project, we will utilize the real noisy incomplete grocery sales record to explore the modeling process.

In order build an end-to-end forecasting pipeline, neural networks seem to be the best candidates. The progress in deep learning research has dramatically improved the state-of-art in speech recognition, computer vision and many other domains such as genomics. Neural networks have not been widely adopted in finance, since its performance has not beat other models' in large datasets with significant noise. Nevertheless, we believe this is a broad way that worth more study and exploration.

In this project, we propose a generalizable pipeline to predict unit sales. The input to our algorithm is the information of an item whose sales people want to predict (including store index, item index, date, promotion information, oil price, etc.). We then use a residual neural network to output a predicted unit sales.

2. Related Work

There has not been many machine learning papers studying similar problems, but predicting sales and recommendations of e-commerce has been a hot topic. In e-commerce, companies and customers interact through images and texts, which provides a platform for research on natural language processing(NLP) and compute vision. One impressive work was to predict sales from the language of product descriptions on Ratuken [8]. However, in lack of text information as features, we have to take inspiration from existing algorithms that have not been applied to this problem.

Recurrent neural networks(RNNs) are the most intuitive choice to deal with sequential data [6]. Long Short Term Memory networks (LSTMs) [4] and Gated Recurrent Units(GRUs) contributed to the progress of machine translation, text summarization and other NLP tasks. However, it could be difficult for RNNs to deal with long sequences.

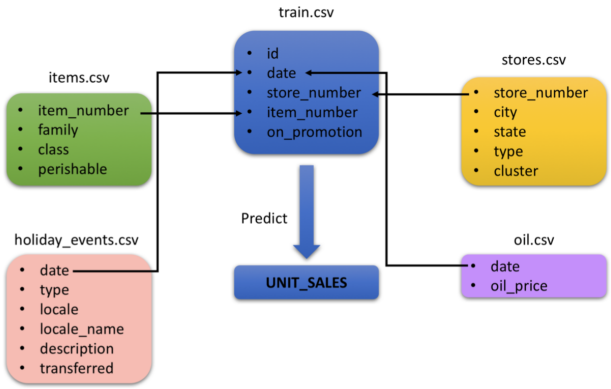

Figure 1. The features of the dataset

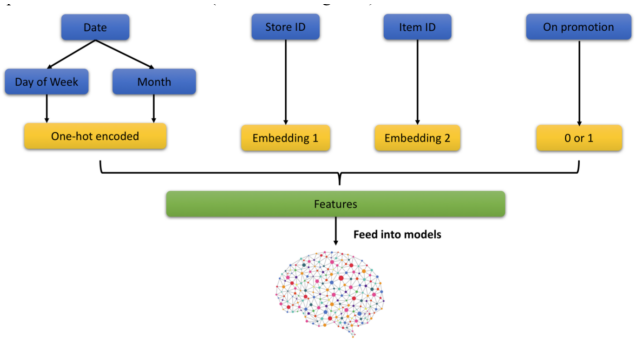

Figure 2. Preprocessing of the dataset

Residual nets(ResNet) is one of the most powerful neural net architectures in computer vision [2] [3] [9]. Although convolutional neutral networks(CNNs) are designed for images, we took the inspiration that residuals conquer the vanishing gradient problems and designed a 2D variance of it. In coping with categorical features, we were inspired by the idea of word embedding [7] and other feature embedding papers in computer vision [11] [10]. We trained embedding matrices for categorical features to capture the similarity between stores and items. We also added month, day and day of the week as features.

3. Dataset and Features

3.1. Dataset and Features

Our data come from the Kaggle competition: Corporacion Favorita Grocery Sales Forcasting [1].Our task is to predict the unit sales amount of an item in a particular store given the store ids, item ids, date and promotion information as the input. The features of the dataset is shown in Figure 1. There are 125,497,040 training examples in total, ranging from 01/01/2013 to 08/15/2017. There are 3,370,464 text examples, ranging from 08/16/2017 to 08/31/2017. Since this project aims to exploring a pipeline for sales forecasting, while the labels of the test set are not available, we split the data from 08/01/2017 to 08/16/2017 as validation set. We will evaluate the performance of our model in predicting future sales on validation set. There are 55 stores and 4100 items in the dataset.

3.2. Preprocessing

In analysis of the dataset, we noticed that store ids and item ids are categorical features, which are typically expanded to one-hot vectors in regressions. However, there are 55 unique stores and 4100 unique items, the size of dataset will explode if we employ the one-hot encoding, given the 5 Gb size of the original training set. To handle the situation, we took inspiration from word embedding in natural language processing. We treated the store ids and the items ids as indices in two vocabularies, and trained a vector representation for each index (as shown in Figure 2).

4. Methods

4.1. Baseline

Before diving into the neural networks, we trained two linear regression models as our baseline, with and without the embedded features respectively. The model trained with embedded features performs much better than the one without, which indicates that the embedding were able to abstract the category features. The result encouraged us to continue using the embedding method.

$$\hat{y} = \theta^\top \mathbf{x}$$

4.2. A Simple Neural Network

At first, we decided to build a embedding + neural network based pipeline, and we trained neural networks with 4, 8, 12 layers. We used rectified linear units(ReLUs) as the activation. ReLU helps prevent vanishing gradient when the value of the node becomes very large. The forward propagation of each layer is shown below:

$$z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]}$$
$$a^{[l]} = ReLU(z^{[l]})$$

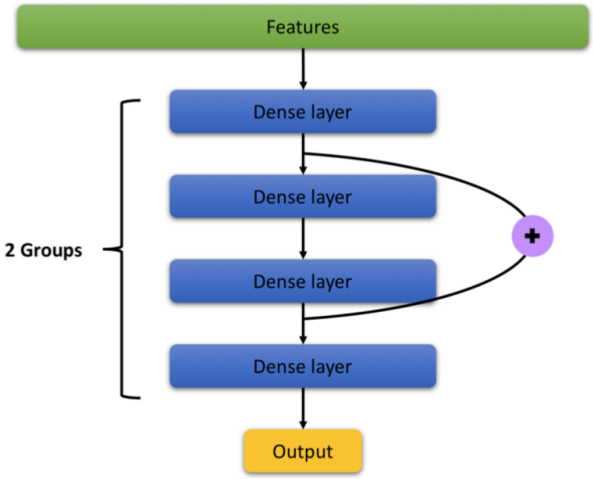

Figure 3. A simple ResNet architecture

4.3. A Residual Network

In training the simple neural network, we found our model was not able to overfit the dataset. According to the lecture on ML advice, we need a model with larger hypothesis space. However, as the neural networks get deeper, training become more and more difficult and I have to adopt very small learning rate at the beginning. Thus we took inspiration from the Residual Deep Networks [2], and built a variant of residual nets that works for 2D data (as shown in Figure 3).

4.4. Teacher Forcing: to Capture Sequential Info

Our ResNet architecture improved the performance of the model dramatically. However, it is not perfect since it cannot capture the sequential information. We have to build a model that functions like RNN but are easier to train. At this time, NLP provides us with an idea again: we found a concept called teacher forcing [5]. Teacher forcing uses the output from a prior time step as input to quickly and efficiently training recurrent neural networks. Inspired by this concept, we concatenated the historical unit sales of last fifteen days into the embedded features. This modified feature improved our dev loss to the stage comparable to the first place of the competition.

5. Experiments and Results

We subset the data from 04/01/2017 to 07/31/2017 and from 07/01/2016 to 08/31/2016 as the training data – 17,894,521 examples in total. We use the data from 08/01/2017 to 08/15/2017 as the dev set – 1,570,968 examples in total. The idea behind the split is that sales record from two years ago may not be a significant indicator for the current sales. However, many goods are seasonal, thus the records from the same season of last year will in informational.

For all experiments, we trained the model for 100,000 iterations with batch size 1024. Loss was recorded and printed out every 1000 iterations. We used SGD optimizer with annealing learning rate, and the initial learning rate was 0.01 for all the experiments.

Embedding matrices were trained together with the network, in an end-to-end manner. There are 55 stores and 4100 items in total. The embedding size of stores is 60, and the embedding size of items was 300. To repeat our best result, we also trained embedding for month, day and day of week, with size 12, 31 and 7 obviously.

5.1. Loss function

We adopted the Normalized Weighted Root Mean Squared Logarithmic Error (NWRMSLE) as our loss function (as shown below).

$$J = \sqrt{\frac{\sum_{i=1}^{m} w_i \times (log(\hat{y}_i + 1) - log(y + 1))^2}{\sum_{i=1}^{m} w_i}}$$

where $w_i = 1.25$ if if an item is perishable, otherwise $w_i = 1$ There are two reasons why we want to choose this loss function. a)This is the loss function used by the competition. It will be more convenient for us to compare our results with others. b) The sales of different items could vary in magnitude, taking log of the sales would help with both training and numerical stability.

5.2. Simple Neural Networks

The simple neural networks takes the features in Figure 1 directly and output the predicted sales. We tried 4, 8 and 12 layers with a hidden unit size of 256. ReLU was used as the nonlinearity. The training curve of the model is shown in Figure 4. The lowest average training loss is about 0.52 and dev loss 0.58

5.3. Residual Nets and Teacher Forcing

We tried residual networks with 2 and 3 residual blocks, i.e 8 and 12 layers. The hidden size was 256 for all layers. We trained the residual nets both with and without the "teacher forcing": previous 15 days' sales as features. The training curve of the model without teacher forcing is shown in Figure 5. The lowest average training loss is about 0.51 and dev loss 0.53. The training curve of the model with historical sales as features is shown in Figure 6. The lowest average training loss is 0.47 and the lowest average dev loss is 0.48.

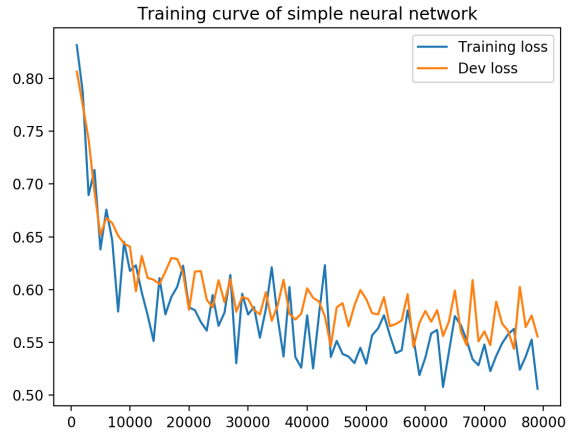

Figure 4. The training curve of the simple neural nets

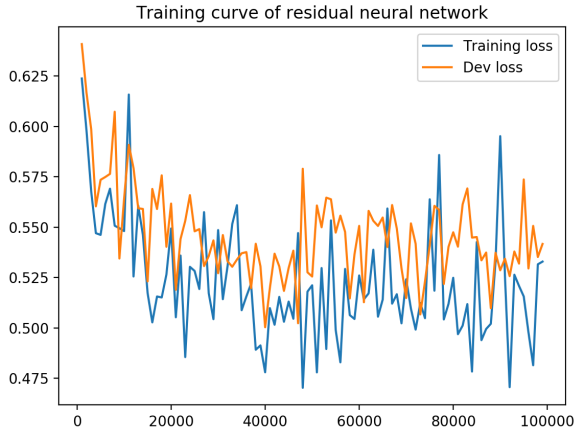

Figure 5. The training curve of the residual neural nets

6. Discussion

6.1. Analysis of Results

To show our results of training, a comparison of those models are shown in Table 1. We could recognize that both the neural nets, the residual nets and the historical features have improved the model from the baseline. However, based on the ML-advice lecture, a good model should be able to fit the dataset. It seems that the noise in our dataset is probably too big, since changing the number of layers from 4 to 12 and changing the hidden unit size from 256 to 512 did not help further decrease the training loss. There are more other analysis we can do for the model, such as looking at the 2D projection of the

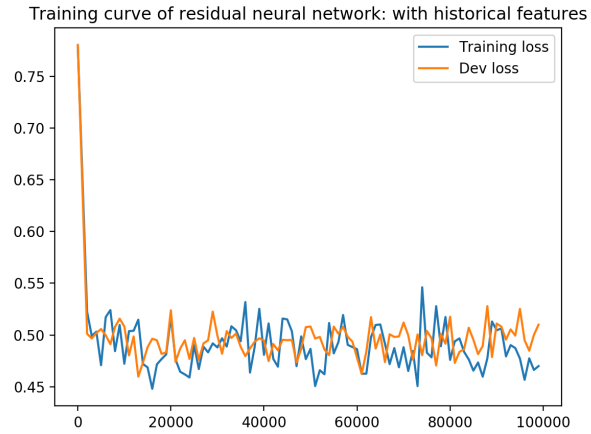

Figure 6. The training curve of the residual neural nets with historical features

Model	Training loss	Dev loss
Linear regression	0.81	0.81
Simple neural Nets	0.52	0.58
Residual nets	0.51	0.53
With historical	0.47	0.48

Table 1. Evaluation on the In-Car test data

embedding of items and stores. One may ask questions like are pen's and pencil's embedding closer to pen's and banana's? Are the embedding of stores in similar locations closer to each other? These are all ways to evaluate the model qualitatively. I am very pitiful that I don't have time for these analysis, but I will continue working on the project.

6.2. Optimizers and Learning Rate

We have tried Adam, Adagrad, momentum and RMSprop optimizers and learning rate from 0.1 to 0.0001. It turns out that SGD with annealing learning rate (decay by 0.2 when reaches plateau) works the best. Adam learns faster but was not able to obtain a better training loss.

6.3. Objective of the Project

My favorite part of the project is about the concept: to build a pipeline to get rid of the painful feature engineering. No matter what industry are you from, just upload the data, let the model deal with the categorical features and embeddings, train until some point, and then store the weights and use it for future sales forecasting. As I mentioned before, using deep learning to financial like

data is still under exploring, but I believe it is a great direction. ra

References

[1] Corporacin Favorita Grocery Sales Forecasting. https://www.kaggle.com/c/favorita-grocery-sales-forecasting, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[4] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[5] A. M. Lamb, A. G. A. P. GOYAL, Y. Zhang, S. Zhang, A. C. Courville, and Y. Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pages 4601–4609, 2016.

[6] Z. C. Lipton, J. Berkowitz, and C. Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[7] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[8] R. Pryzant, Y.-j. Chung, and D. Jurafsky. Predicting sales from the language of product descriptions. 2017.

[9] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995. IEEE, 2017.

[10] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):40–51, 2007.

[11] L. Zhang, Q. Zhang, L. Zhang, D. Tao, X. Huang, and B. Du. Ensemble manifold regularized sparse low-rank approximation for multiview feature embedding. *Pattern Recognition*, 48(10):3102–3112, 2015.