A Study of Feature Construction for Text-based Forecasting of Time Series Variables

Yiren Wang*
University of Illinois at Urbana-Champaign
yiren@illinois.edu

Dominic Seyler*
University of Illinois at Urbana-Champaign
dseyler2@illinois.edu

Shubhra Kanti Karmaker Santu
University of Illinois at Urbana-Champaign
karmake2@illinois.edu

ChengXiang Zhai
University of Illinois at Urbana-Champaign
czhai@illinois.edu

ABSTRACT

Time series are ubiquitous in the world since they are used to measure various phenomena (e.g., temperature, spread of a virus, sales, etc.). Forecasting of time series is highly beneficial (and necessary) for optimizing decisions, yet is a very challenging problem; using only the historical values of the time series is often insufficient. In this paper, we study how to construct effective additional features based on related text data for time series forecasting. Besides the commonly used n-gram features, we propose a general strategy for constructing multiple topical features based on the topics discovered by a topic model. We evaluate feature effectiveness using a data set for predicting stock price changes where we constructed additional features from news text articles for stock market prediction. We found that: 1) Text-based features outperform time series-based features, suggesting the great promise of leveraging text data for improving time series forecasting. 2) Topic-based features are not very effective stand-alone, but they can further improve performance when added on top of n-gram features. 3) The best topic-based feature appears to be a long-term aggregation of topics over time with high weights on recent topics.

1 INTRODUCTION

Forecasting of time series is highly beneficial for optimizing decision making in many domains. For example, the forecasting of temperature enables humans to make more informed decisions when planning an outdoor event, while forecasting product sales is essential for optimizing business decisions. The general goal here is to predict the future values of a given target time series (e.g. sales in financial domain). How to solve this problem in a general way is an important challenge due to its broad impact on many applications.

Most existing forecasting approaches make use of previous values of the target or a related time series [7, 8, 12, 15, 16]. For instance, if gasoline sales rise after a consecutive negative price change, it is likely that sales will continue to grow. Even though future values of a time series tend to correlate with past values, this correlation may be weak and thus using only past values may not be effective.

In this paper we study how to leverage text data that is related to a time series variable for time series forecasting with a focus on general approaches to construct effective features for prediction. Text can be regarded as data generated by "human sensors" about our world. Thus, they often contain useful information that can potentially help predict future values of many time series variables. For example, sentiment in social media about some major events may be relevant for predicting the change of next day's stock market.

There has been previous work on using text data to predict stock prices in financial markets, mostly using word-based features [4, 5, 9, 13]; other work has aimed to optimize prediction for a particular application by using external knowledge [2, 3, 11], such as entities and relations, or sentiment [10, 14]. However, such features are generally domain dependent and would require significant amount of resources (e.g., manually labeled data for training). Thus so far, general features that can be automatically constructed for any text data have been restricted to word-based features.

Due to the complexity of natural languages word-based features are generally insufficient, since many words are ambiguous (polysemy) and some words with different surface forms may have the same or very similar meaning (synonymy). One way to address this problem is to use topics as features. A topic is represented as a probability distribution over words and thus can address polysemy and synonymy, since an ambiguous word would be "split" between multiple topics while synonyms would be allowed to match with each other. Such topics can be discovered in an unsupervised manner from any text data by using a topic model, such as Latent Dirichlet Allocation (LDA) [1]. However, it is unclear how effective features can be constructed using such topics for time series forecasting, which is the main question we study in this paper.

Specificaly, we propose and study a number of general strategies for constructing topical features for time series forecasting that can be applied to any application domain using two basic forecasting scenarios, i.e., prediction of significant change of a time series and prediction of the trend of change (if there is a significant change). Our experimental results allow us to make several findings: First, text-based features outperform time series-based features by a wide margin, suggesting the great promise of leveraging text data for improving time series forecasting. Second, topic-based features are not very effective stand-alone (due to lack of discriminativeness), but when added on top of n-gram features, they can further improve

* The two authors contributed equally to this work.

CIKM'17 , November 6–10, 2017, Singapore, Singapore

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133109

performance, suggesting the need for combining word-based n-gram features with topical features. Finally, the best topic-based feature appears to be a long-term aggregation of topics over time with high weights on recent topics.

2 PROBLEM FORMULATION

Our goal is finding the most effective textual features for time series variable forecasting. To this end, we present a general formulation of the problem, a general strategy for constructing data sets, and a general definition of multiple features that can be applied to any text in any language without requiring manual effort.

2.1 Problem Definition

The input of our forecasting problem consists of a time series $X = \{(x_1, t_1), ..., (x_N, t_N)\}$, and a set of text documents with time stamps $D = \{(d_1, t_1), ..., (d_N, t_N)\}$. The output is the binary categorization label $y \in \{-1, 1\}$ that represents the change of time series variable X. The change is defined as the relative improvement of the value of time series variable $\Delta_t = \frac{x_{t+1} - x_t}{x_t}$. We define two change thresholds Δ_a, Δ_b ($0 \leq \Delta_a \leq \Delta_b$), and formulate two different forecasting problems based on different definitions of label y:

• **Change Prediction Problem (CP):** y_{cp} is the label that marks whether there is significant change in time series variable.

$$y_{cp}^{(t)} = \begin{cases} 1, & \Delta_t \in (-\infty, -\Delta_b] \cup [\Delta_b, +\infty) \ \text{(significant)} \\ -1, & \Delta_t \in [-\Delta_a, \Delta_a] \ \text{(insignificant)} \end{cases} \quad (1)$$

• **Trend Prediction Problem (TP):** y_{tp} marks whether the direction of change is positive or negative.

$$y_{tp}^{(t)} = \begin{cases} 1, & \Delta_t \in [\Delta_b, +\infty) \ \text{(positive)} \\ -1, & \Delta_t \in (-\infty, -\Delta_b] \ \text{(negative)} \end{cases} \quad (2)$$

CP and TP provide two basic formulations underlying all the forecasting applications, and they can be naturally combined by performing CP first, and if a significant change is predicted, further performing TP to predict the trend of change.

2.2 Dataset Construction

The dataset (X, D, Y) consists of a time series X, a set of text documents with time stamps D that are relevant to the time series variable, and a sequence of binary labels $Y = \{y_1, ..., y_N\}$, where $y_t = (y_{cp}^{(t)}, y_{tp}^{(t)})(y_{cp}, y_{tp} \in \{-1, 1\})$. We set a date T to separate data into a training set ($train = \{(x_1, d_1, t_1; y_1), ..., (x_T, d_T, t_T; y_T)\}$), and a test set ($test = \{(x_{T+1}, d_{T+1}, t_{T+1}; y_{T+1}), ..., (x_N, d_N, t_N; y_N)\}$).

2.3 Feature Construction

We focus on extracting general, robust textual features that can be constructed without manual effort and applicable to all languages.

• **Word-based Features:** We extract n-grams $w(n)$ from the textual, preprocessed documents. We employ both raw count for n-grams ($f_c(n)$) and the TFIDF [6] score ($f_{tfidf}(n)$).

$$w(n) = (w_{k_1}, ..., w_{k_n}) \quad (3)$$

$$f_c(n) = count(w(n), d) \quad (4)$$

$$f_{tfidf}(n) = count(w(n), d) \times log\Big(\frac{1 + N_d}{1 + df(d, w(n))}\Big) \quad (5)$$

where $count(w(n), d)$ is the number of times w_n shows in document d, N_d is the total number of documents, $df(d, w(n))$ is the number of documents that contain n-gram $w(n)$.

• **Topic-based Features:** We also derive features based on topic modeling, where each document in the collection is modeled as a mixture of a pre-defined number of topics k. We use these document-topic distributions θ, and some variants that incorporate historical topic distributions as topic-based features.

a. **Topic Distribution (*TD*)**: θ_t, the document-topic distribution of document d_t.
b. **Topic Change (*Chg*)**: We take the difference in topic distributions θ of documents in the current and previous days, which identifies changes in topic distributions over time.

$$\theta_t^{\Delta} = \theta_t - \theta_{t-1} \quad (6)$$

c. **Topic History**: Apart from *Chg*, we introduce another way to capture the historical topic information by taking the weighted sum in a window of previous days.

$$\theta_t^{hist}(\alpha, L) = \sum_{i=1}^{L} \alpha^i \theta_{t-i} \quad (7)$$

where L is the window size, and α is the decaying factor over time, based on the motivation that the influence of previous topic distribution decays over time.

We introduce two ways to combine the TD and the topic history: by adding $\theta_t + \theta_t^{hist}(\alpha, L)$ ($Hist.add$), and by concatenation $[\theta_t; \theta_t^{hist}(\alpha, L)]$ ($Hist.cont$).

3 EXPERIMENTS

We evaluate the effectiveness of our general method for the problem of stock price forecasting, with a dataset consisting of stock prices of 53 companies over 7 years (October 2006 to November 2013), and a corpus of Reuters and Bloomberg news articles from [2].

3.1 Dataset

We collect news articles relevant to each company, generate labels and create the (X, D, Y) tuples as follows: We (1) first rank the list of NASDAQ companies according to the amount of market capitalization, which correlates with news coverage. Then query the company names and stock symbols of the 100 highest scoring companies against a Lucene [1] index. (2) set thresholds $\Delta_a = 0.5\%, \Delta_b = 1.0\%$ to get the labels, and perform undersampling to balance the classes. (3) remove all companies with a coverage of less than 100 days, which reduces the number to 53 companies (4) set the separation date $T =$ May 18^{th}, 2012 for CP and $T =$ July 7^{th}, 2012 for TP, to divide the training and test sets, which end up with 66% training instances and 34% test instances. Our final dataset has 26,278 instances for CP, 13,934 instances for TP and is available at http://bit.ly/2w12Ybp.

3.2 Machine Learning Framework

Once features are constructed, they serve as input to a machine learning framework, which enables us to compare features on their own and in combination.

[1] http://lucene.apache.org

• **Classifier:** Since our focus is to study features, we fix the classifier and use logistic regression (LR) to predict labels for future instances. We run 3-fold cross validation and perform grid search for parameters of the classifier, including penalization norm (l_1, l_2), regularization weight ($C \in [0.001, 0.01, 0.1, 1, 10, 100, 1000]$), and the solver (newton-cg, liblinear and lbfgs) [2].
• **Feature Selection:** We do feature selection for word-based features to avoid overfitting. For each feature candidate, we compute the statistical score that gives higher values to features that have higher correlation with one of the classes. We try both chi-square (X^2) and mutual information (MI) scores, and select $Top\text{-}k$ features.
• **Evaluation:** We measure model performances with classification accuracy, since we work with a balanced dataset.

3.3 Models

• **Vector Auto Regression (VAR):** We use VAR, an econometric model that captures the linear inter-dependencies among multiple time series. We predict x_{t+1} using the opening stock prices of up to $t = 20$ days as input. The label is then derived as in Equation 2.
• **Time Series-based Features:** We employ the percentage of change in opening prices of up to $t = 20$ previous days as features, and use LR for label prediction.
• **Text-based Features:** We use pure n-gram features with $n = 1, 2$ and feature selection. We run LDA[1] over the whole corpus and get topic distribution for each documents.

4 RESULTS AND DISCUSSION

• **Overall Comparisons:** Table 1 shows the results for TP[3] (we run cross-validations for the best hyperparameter combinations), from which we can draw the following conclusions:

a. The best performing model is the one combining all word-based features, topic-based features, and time series features[4].
b. Word-based features appear to be strongest signal for prediction. Adding topic-based features can enhance feature performance, while topic-based features alone are not as powerful as n-grams.
c. Time series-based features appear not to be a good standalone feature compared with topic-based[5] or word-based features[6]. They only work in combination with text or topic based features.
d. Performances with different hyperparameters for feature construction will be analyzed in detail in the following sections. In general, a small number of selected n-grams ($Top\text{-}k$=10, 20) and concatenated topic history ($Hist.cont$) turn out to be the most favorable settings. Topic change (Chg) and added topic history features ($Hist.add$) contribute little.

• **Time Series-based Features:** Table 2 shows the results for numeric time series features only. The first row is VAR with previous stock prices up to 20 days. The second row is LR with previous stock price changes as stand-alone features. We can see that (1) Adding more previous signals does not help VAR nor LR. (2) Neither price nor price changes on their own are good signals for forecasting the future price trend. Our interpretation is that time series data itself is too noisy to capture patterns and make accurate predictions, and suggests potential for extracting signals from related text data.

Table 1: Comparison of different models (TP)

Models	Performance
VAR	50.95
Price Change	50.71
n-grams ($n = 1$)	56.26
n-grams ($n = 2$)	56.20
n-grams ($n = 1$) + Price Change	56.39
n-grams ($n = 2$) + Price Change	56.23
Topic-based	52.73
Topics + n-grams ($n = 1$)	56.02
Topics + n-grams ($n = 2$)	56.00
Topics + n-grams + Price change	**56.44**

Table 2: Performances of time series-based models (TP)

Days	**1**	2	3	4	5	10	20
VAR	**50.95**	49.70	50.61	49.94	49.78	50.63	49.00
LR	**50.71**	50.48	50.54	50.19	49.22	48.98	49.07

Table 3: $Top\text{-}k$ ($k = 5$) selected n-grams using X^2 and MI (TP)

Unigram		Bigrams	
X^2	MI	X^2	MI
fell	percent	fell percent	share fell
myspace	apple	rose percent	rose percent
yahoo	company	drop percent	fell percent
rose	google	gain percent	gain point
burbank	has	fell point	new york

Table 4: Performances of word features (TP, x^2 / MI)

$Top\text{-}k$	$f_c(1)$	$f_{tfidf}(1)$	$f_c(2)$	$f_{tfidf}(2)$
10	54.14 / 51.28	55.81 / 51.34	54.66 / 51.28	54.61 / 51.34
20	54.72 / **55.11**	54.77 / 55.59	54.55 / 55.11	54.92 / 55.59
30	54.83 / 55.09	54.31 / .56.20	53.90 / 55.44	54.12 / **56.20**
40	54.48 / 54.87	54.53 / **56.26**	54.55 / 54.90	53.94 / 55.44
50	54.27 / 54.74	53.96 / 56.04	54.66 / 54.25	54.72 / 56.11
80	54.42 / 54.20	54.55 / 55.74	54.22 / **55.59**	54.38 / 55.55
100	54.68 / 53.83	54.16 / 55.59	53.96 / 53.68	54.38 / 55.26
1000	51.36 / 51.95	52.19 / 51.52	48.81 / 49.74	51.28 / 50.41

• **Word-based Features:** We manually inspect the highest ranked n-grams according to feature selection score (Table 3). It can be seen that selected unigrams are more related to price trends (e.g., fell, rose), or companies in concern (e.g, myspace, apple, yahoo). The results support our assumption that human interpretation in text data provides a good summary, which can be used as an external signal for time series prediction.

We then explore the effect of n-grams computation methods ($f_c(n)$, $f_{tfidf}(n)$), feature selection parameter $Top\text{-}k$ and scoring methods (chi-square, MI) for n-grams (Table 4). We observe that (1) A small number of features increase performance. When more than 100 features are employed, the method suffers from severe overfitting. (2) In general TFIDF representation of features outperform raw count representation. (3) Surprisingly, adding bigrams improves complexity yet does not improve performance (even drops slightly).

• **Topic-based Features:** We investigate how different variants and combinations of topic-based features impact the forecasting

[2]Implementations are based on scikit-learn (http://scikit-learn.org/).
[3]Results for CP are similar, we do not show them due to the lack of space.
[4]McNemar test on the best performing model and word-based features with $p = 0.51$
[5]McNemar test with p-value = 0.0531
[6]McNemar test with p-value < 0.00001

Table 5: Performances of topic-based features (CP / TP)

k	TD	Chg	TD + Chg	Hist.add	Hist.cont	TD + Chg + Hist.add	TD + Chg + Hist.cont
10	52.92 / 51.91	52.25 / 51.88	52.78 / 52.06	51.99 / 51.71	**53.38** / 52.10	52.28 / 52.58	53.15 / ***52.73***
15	**54.40** / 51.13	52.58 / **51.80**	53.96 / 51.49	54.15 / 51.28	54.33 / 51.62	54.10 / 51.60	54.07 /51.67
20	53.96 / 51.47	53.55 / 52.43	54.07 / 51.02	54.04 / 50.87	54.29 / 52.17	54.45 / 51.56	**54.49** / **51.73**
25	54.01 / 50.87	53.70 / 50.97	53.95 / 51.06	55.13 / 50.58	**54.80** / **52.36**	54.78 / 51.39	54.60 / 51.41
30	**55.21** / 51.34	53.06 / 51.19	54.09 / 51.52	54.98 / 51.80	54.98 / 51.86	54.53 / 51.62	54.65 / **52.04**
35	54.64 / 50.84	53.71 / 50.32	54.53 / 51.32	54.66 / 51.28	**54.83** / 51.86	54.48 / 51.36	54.43 / **52.06**
40	54.72 / 50.76	54.09 / 51.52	53.33 / 51.71	54.83 / 51.13	54.53 / 51.21	**54.57** / **51.97**	54.41 / 51.49
45	53.52 / 50.61	53.42 / 50.84	53.79 / 50.61	53.84 / 50.35	53.87 / 51.36	**54.26** / 51.06	53.95 / **52.32**
50	53.39 / 51.23	53.52 / 51.26	52.63 / 50.37	53.70 / 51.13	53.79 / **52.14**	**54.30** / 51.13	53.93 / 51.88

performances in change prediction (CP) and trend prediction (TP) respectively (Table 5).

a. A small number of topics (k between 10 and 30) performs better for TP, while k between 25 to 40 performs better for CP.
b. Topic Change (*Chg*) does not appear to be a very strong signal for either CP or TP. Particularly for CP, adding *Chg* hurts performance compared with TD.
c. For topic history features, combining history by concatenation ($Hist.cont$) generally outperforms combining by adding ($Hist.add$) in both CP and TP problems by a large margin. This is intuitive in that TD is an important signal and should not be "mixed" with other signals.
d. The combination of all features ($TD + Chg + Hist.cont$) performs best with an accuracy of 52.73% for TP, while $Hist.cont$ performs best with 54.98% for CP. It's evident that $Hist.cont$ is the most invaluable signal, and for trend prediction, TD and *Chg* can be good complementary signals.

Table 6 and 7 show the results for *Hist.cont* in CP and TP respectively. Each table investigates different settings for decaying factor α and window size L.

a. For CP, a larger decaying factory ($\alpha = 0.8, 0.9$) and medium window size ($L = 5, 10$) generally performs better.
b. For TP, there's no single fixed value for α or L that dramatically stands out. Interestingly, we can see that the $\alpha - L$ pairs on the diagonal of the table tend to perform better. Namely, the combinations of large α - small L and small α - large L are more favorable. This is consistent with the intuition that recent signals have more significant influence in trend forecasting.

Table 6: *Hist.cont* with different hyperparameters (CP)

α	$L = 1$	$L = 3$	$L = 5$	$L = 10$	$L = 20$
0.6	54.47	54.51	**54.56**	54.52	54.48
0.7	**54.68**	54.54	54.58	54.61	54.60
0.8	**54.90**	54.54	54.80	54.78	54.80
0.9	54.81	54.64	54.94	**54.83**	54.69
1.0	54.83	54.41	**54.98**	54.10	53.86

Table 7: *Hist.cont* with different hyperparameter values (TP)

α	$L = 1$	$L = 3$	$L = 5$	$L = 10$	$L = 20$
0.6	51.78	51.58	51.91	51.88	**51.95**
0.7	51.78	51.75	52.04	**52.36**	52.17
0.8	51.78	**52.23**	51.80	52.10	51.86
0.9	51.82	**51.88**	51.71	51.86	51.41
1.0	**52.10**	51.93	52.06	51.60	51.36

5 CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel general framework for time series forecasting problem with a general strategy for constructing datasets and general features that can be extracted without requiring manual work. We formulated the problem as two consecutive prediction tasks, namely change and trend prediction. We derived novel textual features and evaluated them on a stock prediction dataset, which is available for download. The experimental results show that (1) text-based features are more effective than time-series features, suggesting great promise of using text data to improve time series forecasting, (2) the proposed topic-based features can be combined with the word-based features to further improve accuracy, and (3) the best performance is achieved when word-based, topic-based and time series-based features are used in combination. Our work lays a good foundation for further study of this topic in multiple ways, including (1) the exploration of our features on different datasets (e.g., political news for election forecasting), (2) derive other topic-based features and explore how deep learning can be leveraged for this task, (3) test classifiers in addition to logistic regression (4) show how our features can be utilized in concrete applications, e.g. decision support for stock trading.

REFERENCES

[1] D. M. Blei et al. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan), 2003.
[2] X. Ding et al. Using structured events to predict stock price movement: An empirical investigation. In *EMNLP*, 2014.
[3] X. Ding et al. Deep learning for event-driven stock prediction. In *IJCAI*, 2015.
[4] S. Feuerriegel and R. Fehrer. Improving decision analytics with deep learning: the case of financial disclosures. In *ECIS*, 2016.
[5] M. Hagenau et al. Automated news reading: Stock price prediction based on financial news using context-capturing features. *Decision Supp. Syst.*, 55(3), 2013.
[6] K. S. JONES. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1), 1972.
[7] D. A. Kumar and S. Murugan. Performance analysis of indian stock market index using neural network time series model. In *PRIME*, 2013.
[8] C.-J. Lu et al. Financial time series forecasting using independent component analysis and support vector regression. *Decision Supp. Syst.*, 47(2), 2009.
[9] M.-A. Mittermayer and G. Knolmayer. *Text mining systems for market response to news: A survey.* 2006.
[10] T. H. Nguyen et al. Sentiment analysis on social media for stock movement prediction. *Expert Syst. Appl.*, 2015.
[11] T. Preis et al. Quantifying trading behavior in financial markets using google trends. *Scientific reports*, 3, 2013.
[12] M. M. Rahman, S. K. K. Santu, M. M. Islam, and K. Murase. Forecasting time series – a layered ensemble architecture. In *IJCNN*, pages 210–217. IEEE, 2014.
[13] Y. Shynkevich et al. Predicting stock price movements based on different categories of news articles. In *SSCI*, 2015.
[14] J. Si et al. Exploiting topic based twitter sentiment for stock prediction. In *ACL*, 2013.
[15] F. E. Tay and L. Cao. Application of support vector machines in financial time series forecasting. *Omega*, 29(4), 2001.
[16] Y. Zhang and L. Wu. Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network. *Expert Syst. Appl.*, 36(5), 2009.