App Download Forecasting: An Evolutionary Hierarchical Competition Approach

Yingzi Wang^{⋆†}, Nicholas Jing Yuan[‡], Yu Sun[‡], Chuan Qin[⋆], Xing Xie[†]

*University of Science and Technology of China

†Microsoft Research

#Microsoft Corporation

‡University of Melbourne

yingzi@mail.ustc.edu.cn, nicholas.yuan@microsoft.com, yusun.aldrich@gmail.com chuanqin0426@gmail.com, xingx@microsoft.com

Abstract

Product sales forecasting enables comprehensive understanding of products' future development, making it of particular interest for companies to improve their business, for investors to measure the values of firms, and for users to capture the trends of a market. Recent studies show that the complex competition interactions among products directly influence products' future development. However, most existing approaches fail to model the evolutionary competition among products and lack the capability to organically reflect multi-level competition analysis in sales forecasting. To address these problems, we propose the Evolutionary Hierarchical Competition Model (EHCM), which effectively considers the time-evolving multi-level competition among products. The EHCM model systematically integrates hierarchical competition analysis with multi-scale time series forecasting. Extensive experiments using a real-world app download dataset show that EHCM outperforms state-of-theart methods in various forecasting granularities.

1 Introduction

Predicting and understanding products' future sales is an essential task for companies to follow users' satisfaction level and improve business revenue. It also enables investors to estimate a company's value and allows users to follow the trends of a market.

Usually, there is intense competition in a product market. Similar to the competition in the natural world (among animals and plants), which plays an irreplaceable role in natural selection, the competition in a product market also greatly affects the development and sales of products. As illustrated by Figure 1, there is often a three-level hierarchy in a typical market, where the top-level is users, bottom-level is products (here we use apps as an example), and in-between is various categories of user demands, e.g., the demands for having food, staying healthy, and playing games. Normally, the overall amount of these categories of demands from users is relatively stable. When users' engagement in one category of demand increases, their time spent on other demand categories will decrease, which leads to the competition among different

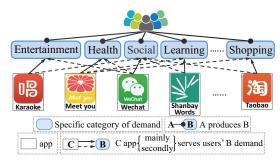


Figure 1: A three-level hierarhy in the app market

categories of demands (in the middle level in Figure 1). In a lower level, each product normally serves several categories of user demands [Radosavljevic et al., 2016]. For example, as shown in Figure 1, Wechat is mainly used for social activities, while its step count function caters to users' health demands. These products compete with each other by providing engaging functions to cater to various categories of demands, and this forms the competition among products (in the bottom level in Figure 1). Therefore, there exists a two-level competition in a typical product market.

It is challenging to model such hierarchical competition among products. Traditional sales forecasting methods, such as ARIMA [Brockwell and Davis, 2016], Recurrent Neural Networks [Chandra and Zhang, 2012], do not consider such hierarchical competition and often only analyze the sales time series of a single product. There are also several sales prediction methods using side information, e.g., reviews [Yu et al., 2010] or spatial information [Garber et al., 2004], to improve accuracy. The side information, however, cannot effectively model the competition pattern among products. Although a few models try to use the interactions among items, such as MLDS [Rogers et al., 2013] and Trimine [Matsubara et al., 2012], they also lack the capabilities to describe the hierarchical competition patterns.

Therefore, we propose an Evolutionary Hierarchical Competition Model (EHCM) to model the time-evolving hierarchical competition among products and to provide more accurate product sales forecasting. Specifically, EHCM considers the competition patterns at two levels: the product level and demand category level, and exploits a linear dynamic system to model the evolutionary competition and interactions among the two competition levels. For the first time, EHCM

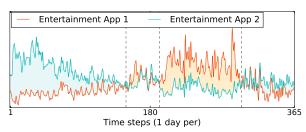


Figure 2: Competition relationship between two apps

organically integrates the hierarchical competition analysis with time series analysis for forecasting. Since the app market is a typical product market with a large amount of products and users (where the sales of an app is the download amount), in this paper we investigate EHCM with the specific application of app download amount forecasting. In summary, this paper offers the following main contributions:

- We propose a novel evolutionary hierarchical competition relationship among products, and investigate a two-level competition hierarchy (product level and demand category level) for sales forecasting.
- We systematically integrates evolutionary hierarchical competition analysis with multi-scale time series forecasting, which can effectively detect long-term competition patterns among products.
- We use a real-world app download data for over 20 thousands apps in the Chinese market to evaluate our model. The results shows that EHCM significantly ourperforms the baseline methods.

2 Background for Hierarchical Competition

In biology, species co-existing in the same environment will compete with each other for resources, and the ones that are less competitive will gradually die out [Begon et al., 2005; Sahney et al., 2010]. Competition in the natural world is hierarchical. The food resources that each species relies on establish the relations among different species, forming the food chain [Briand and Cohen, 1987]. The position of a species in a food chain represents its trophic level. Typically, trophic level 1 is the producers (or autotrophs in biology) who absorb nutrients and water from the natural world and manufacture organism using photosynthesis. Trophic level 2 is the primary consumers (or herbivores) who consume producers as food. Species occupying the same trophic level will compete for food resources obtained from the lower trophic level (or natural resources, like water, soil, and sunshine)

Similar to the competition pattern among species, products in a free market also compete with others vigorously for survival [Johnson *et al.*, 2013]. Users, the fundamental assets in a market, are similar to natural resources in the environment. Users have various categories of demands in their daily life, e.g., demands for food, entertainment, and health. For the sake of analysis convenience, assuming that the total demands of a number of users are stable (at least within a certain period), each category of demand competes with others for more user "resources", which is analogous to producers' competition for resources in biology. For example, when

users' total demands for games increase (e.g., when games like *Pokemon Go* rapidly become popular), their demands for reading or exercising will decrease simultaneously. Various products in the market compete with others for these *categories of demands*, which are their "food", by providing various functions to cater to these demands. Such hierarchical competition relations are illustrated in Figure 1.

3 Method

3.1 Problem Definition

Assume there are M demand categories of users for app services: $C = \{c_1, c_2, ..., c_M\}$, such as entertainment, social, etc. Different apps can cater to users' different demands. Assume that there are N apps in an app system, represented as $A = \{a_1, a_2, ..., a_N\}$, each of which is at the service of at least one demand category (more than one category for most apps). The collected download amounts of these apps form the download matrix $X \in R^{N \times T}$, where T is the number of time steps. The download amount of app a_n in time t is $X_{t,n}$.

The problem is defined as forecasting the amount of download of each app. Specifically, given the download amount of all the N apps in the past T time steps, our goal is to forecast the download amount of each app a_n in future \hat{T} time steps, which is represented as $\hat{X}_n \in R^{1 \times \hat{T}}$.

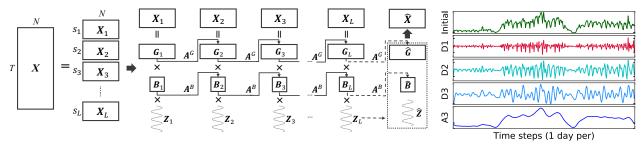
3.2 Model Formulation

Model Overview

To solve this problem, we propose a forecasting model which systematically integrates evolutionary hierarchical competition patterns with time series analysis. Specifically, we first split time series matrix \boldsymbol{X} into several stable segments, within each of which the competition among apps is relatively steady and the trends of number of downloads are almost unchanged. Then, we exploit an evolutionary matrix trifactorization method to explore the hierarchical competition in each segment, obtaining the app service factor, demand category competition factor, and user demand time series factor, and we investigate the evolution of the first two competition factors. Next, we adapt wavelet transform method to decompose user demand time series into different scales and apply a multi-scale autoregression algorithm for forecasting.

Hierarchical Competition in EHCM

In a constant environment, the system approaches a stable equilibrium after a period of competition among different species [Huisman and Weissing, 1999]. We assume that the competition among apps will also lead to a temporary stabilization over a period of time. During that time, the download amount of each app has a stable fluctuation pattern. For example, Figure 2 shows the download amounts of two apps over one year in the "Entertainment" category on the iOS platform. We can see that the competition relation between these two apps can be roughly segmented into 4 sections (illustrated by dotted lines). In each segment, the fluctuations of the two sequences have stable patterns, e.g, in Segment 1, download amount of app 1 has a steady increasing trend, while the download amount of app 2 continuously decreases at a stable speed as well.



(a) Illustration of hierarchical competition evolution

(b) 3-level wavelet transform of an app download time series

Figure 3: Illustration of hierarchical competition evolution and a 3-level non-decimated wavelet transform

According to the stabilization in different continuous time periods, we can divide the T time steps into L stabilization segments. In each segment, the competition among apps approaches to a stable equilibrium, and the download trends of these apps have minor changes. The way we find these stabilization segments is to find the segments that minimize the intra-segment variance of download matrix X [Himberg $et\ al.$, 2001; Fu, 2011]. When we segment X into L sub-segments $\mathcal{X} = \{X^1, X^2, ..., X^L\}$, the total intra-segment variance of the L matrices is computed as $\Omega(\mathcal{X}) = \frac{1}{T} \sum_{l=1}^L \sum_{t=1}^{s_l} \|X_t^l - \mu_l\|_F^2$, where $\|\cdot\|_F$ is the Frobenius norm of a matrix, $\mu_l \in R^{N\times 1}$ is the mean vector of segment X^l , and $S = \{s_1, s_2, ..., s_L\}$ is segment length list $(\sum_{l=1}^L s_l = T)$. Here we adopt the Global Iterative Replacement (GIR) method [Himberg $et\ al.$, 2001] to segment matrix X into L sub-segments. The main idea is that we first randomly divide matrix X into L segments, and then iteratively merge adjacent segments and search for a new division position that minimizes $\Omega(X)$.

As mentioned before, the competition in a market is hierarchical. To obtain a compact competition model and hence retain effective prediction performance, we assume that in each stabilization segment l, the bulk of interactions among apps and number of downloads can be represented hierarchically by three levels of factors: app service factor, demand category competition factor, and user demand time series factor:

$$\mathbf{X}^l \approx \mathbf{G}^l \mathbf{B}^l \mathbf{Z}^l \text{ for all } l = 1, 2, ..., L,$$
 (1)

where

- $G^l \in R^{N \times M}$ is the app service factor for segment l. We assume that an app can cater to users' more than one demand categories. $G^l_{n,m}$ represents the demand amount that app a_n can serve for category c_m in segment l. App service factor is time-evolving among the L segments, representing the app competition evolution.
- $\mathbf{B}^l \in R^{M \times M}$ is the demand category competition factor for segment l. In \mathbf{B}^l we obtain M latent demand categories, and each demand category dynamically competes with others for limited user "resources". $\mathbf{B}^l_{m_1,m_2}$ denotes the demand proportion that category c_{m_1} takes from c_{m_2} in segment l. Demand competition factor is also time-evolving, reflecting the demand category competition evolution.
- $\mathbf{Z}^l \in R^{M \times s_l}$ is the user demand time series factor for M categories in segment l. We assume that user demands

can be quantified and will influence app download quantity with the joint leverage of the app service and demand category competition factors.

We can obtain these factors and matrices through matrix tri-factorization [Zhuang *et al.*, 2013], which can be obtained by optimizing the following objective function:

$$G^{l}, B^{l}, Z^{l}_{\{l=1,...,L\}} = \min_{G^{l}, B^{l}, Z^{l}} ||X^{l} - G^{l}B^{l}Z^{l}||_{F}^{2}.$$
 (2)

Competition Evolution

Next, we consider the competition evolution among the L stabilization segments. From the above analysis, we can see that each app competes with each other for limited demand resources. To enable efficient evolution-modeling, we assume that the app competition transferring from segment l to l+1 can be represented as linear transfer matrix \mathbf{A}_G . To further simplify this dynamic system, we propose a method in which the common structures of all the demand transfers are represented as the same transfer matrix \mathbf{A}_G . Here, given \mathbf{G}^l , l=1,...,L-1, we can generate \mathbf{G}^{l+1} according to the conditional distribution:

$$G^{l+1}|G^l \sim \mathcal{N}(A_G \cdot G^l, Q),$$
 (3)

where Q is the conditional covariance shared by all G^l . Similarly, we can generate demand category competition factor B^{l+1} according to the conditional distribution:

$$\mathbf{B}^{l+1}|\mathbf{B}^{l} \sim \mathcal{N}(\mathbf{A}_{B} \cdot \mathbf{B}^{l}, \mathbf{R}), \tag{4}$$

where A_B is the transfer matrix which describes the dynamics of evolving demand competition factors $B^1, ..., B^L$, and R is the conditional covariance shared by all B^l . Although the assumed linear transformation might cause information loss for the true competition progression, it provides a valid approximation which reduces the model complexity and can be computed deterministically and efficiently.

In a nutshell, our goal now is to estimate the parameters $\{G^l, B^l, Z^l, A_G, A_B\}$, for l=1,...,L that minimize the objective function:

$$\Phi = \sum_{l=1}^{L} \| \mathbf{X}^{l} - \mathbf{G}^{l} \mathbf{B}^{l} \mathbf{Z}^{l} \|_{F}^{2} + \sum_{l=1}^{L-1} (\| \mathbf{G}^{l+1} - \mathbf{A}_{G} \mathbf{G}^{l} \|_{F}^{2} + \| \mathbf{B}^{l+1} - \mathbf{A}_{B} \mathbf{B}^{l} \|_{F}^{2}).$$
(5)

User Demand Forecasting

We then consider the demand time series evolution. Different from the above two types of competition factors, time

series matrices $\mathbf{Z}^l, l=1,...,L$, contain temporal information themselves. To obtain the complete time series information, we assemble the L time series of each demand category into a whole sequence, obtaining a new time series matrix $\mathbf{Z} \in R^{M \times T}$. We aim to predict the future demands of the M time series from \mathbf{Z} .

In time series forecasting, a significant challenge is that: learning too much noise may cause overfitting while neglecting fast dynamic components may cause underfitting [Soltani, 2002]. To tackle this problem, we decompose a series into different scales and capture the dynamic components at each scale to avoid both overfitting and underfitting. Here we adopt the Wavelet Transform method [Percival and Walden, 2006] to decompose a time series $\mathbf{Z}_i \in$ $R^{1\times T}$ into multi-scale components $\{W_1^i,...,W_J^i\}$, where the decomposition level is J-1. Unlike traditional discrete wavelet transform, which picks up every second data point in a sequence at each level, we apply a non-decimated transform. This means that all the series obtained from the decomposition have the same size as the initial series, i.e., $W_i^i \in$ $R^{1\times T}$ for each j. Figure 3(b) shows a 3-level non-decimated decomposition of an app download time series, where A represents approximation and D represents detail. The original signal passes through a low-pass and a high-pass filter to get one high-frequency component "D 1" and low-frequency component "A 1", and then using the same method the sequence "A 1" is futher decomposed into "D 2" and "A 2", and so forth. Since "A n" is the sum of "D n + 1" and "A n + 1", the original time series \mathbf{Z}_i can be represented by J-1 detail sequences and 1 approximation sequence.

After the wavelet transform, we forecast the future value for each detail series and the approximation series. We apply the multi-scale autoregressive method [Renaud *et al.*, 2005; Ernst *et al.*, 2007] to perform such forecasts. Specifically, for each time series \mathbf{Z}_i , we have:

$$\mathbf{Z}_{i,t+\hat{t}} \approx \sum_{j=1}^{J-1} \sum_{k=1}^{K} \mathbf{Y}_{j,k}^{i,\hat{t}} \mathbf{W}_{j,t-1-2^{j}(k-1)}^{i} + \sum_{k=1}^{K} \mathbf{Y}_{J,k}^{i,\hat{t}} \mathbf{W}_{J,t-1-2^{J}(k-1)}^{i},$$

where $(2+2^J(K-1)) \le t \le T$ and $(\mathbf{W}_1^i,...,\mathbf{W}_J^i)$ are the wavelet transform results of time series $\mathbf{Z}_i, i=1,...,M$, in which $(\mathbf{W}_1^i,...,\mathbf{W}_{J-1}^i)$ are the detail series and \mathbf{W}_J^i is the approximation series. K is a constant (e.g., 2). After we obtain the parameters $\mathbf{Y}^{i,\hat{t}}$ for each $i \in (1,...,M)$ and each $\hat{t} \in (1,...,\hat{T})$, we can forecast the future \hat{T} values of each time series \mathbf{Z}_i .

3.3 Optimization

We first estimate parameters $\{G^l, B^l, Z^l, A_G, A_B\}$ for l = 1, ..., L in Equation (5) with stochastic gradient descent (SGD), where the main task is to obtain the gradient of each parameter. We compute their gradients as follows:

• Gradient of G^l :

$$\begin{split} \frac{\partial \mathbf{\Phi}}{\partial \mathbf{G}^l} &= -2(\mathbf{X}^l - \mathbf{G}^l \mathbf{B}^l \mathbf{Z}^l)(\mathbf{B}^l \mathbf{Z}^l)^\top + 2\lambda(\mathbf{G}^l - \mathbf{G}^{l-1} \mathbf{A}_G) \\ &+ 2\beta(\mathbf{G}^{l+1} - \mathbf{G}^l \mathbf{A}_G)(-\mathbf{A}_G)^\top, \end{split}$$

where λ equals 1 for l=2,...,L and equals 0 otherwise, β

equals 1 for l = 1, ..., (L - 1) and equals 0 otherwise.

• Gradient of B^l :

$$\frac{\partial \mathbf{\Phi}}{\partial \mathbf{B}^{l}} = -2(\mathbf{G}^{l})^{\top} (\mathbf{X}^{l} - \mathbf{G}^{l} \mathbf{B}^{l} \mathbf{Z}^{l}) (\mathbf{Z}^{l})^{\top} + 2\lambda (\mathbf{B}^{l} - \mathbf{B}^{l-1} \mathbf{A}_{B}) + 2\beta (\mathbf{B}^{l+1} - \mathbf{B}^{l} \mathbf{A}_{B}) (-\mathbf{A}_{B})^{\top},$$

where λ and β have the same definitions as above.

• Gradient of \mathbf{Z}^l :

$$\frac{\partial \mathbf{\Phi}}{\partial \mathbf{R}^l} = -2(\mathbf{G}^l \mathbf{B}^l)^\top (\mathbf{X}^l - \mathbf{G}^l \mathbf{B}^l \mathbf{Z}^l),$$

• Gradient of A_G :

$$\frac{\partial \mathbf{\Phi}}{\partial \mathbf{A}_G} = \sum_{l=1}^{L-1} -2(\mathbf{G}^{l+1} - \mathbf{A}_G \mathbf{G}^l)(\mathbf{G}^l)^\top,$$

• Gradient of A_B has the same form with A_G .

Next we estimate parameters $\mathbf{Y}^{i,\hat{t}}$ with the least square method. For each time series \mathbf{Z}_i , i=1,...,M, and each $\hat{t} \in (1,2,...,\hat{T})$, we can rewrite Equation (6) as $\mathcal{Z} = \mathcal{W}\mathcal{Y} + \epsilon$, where $\mathcal{Z} \in R^{\bar{T} \times 1}$, $(\bar{T} = (T - 2^J(K - 1) - 1))$, $\mathcal{W} \in R^{\bar{T} \times JK}$, $\mathcal{Y} \in R^{JK \times 1}$, $\epsilon \in R^{\bar{T} \times 1}$ is the error vector. The approximation of \mathcal{Y} (represented as $\tilde{\mathcal{Y}}$) is given by $\tilde{\mathcal{Y}} = (\mathcal{W}^{\top}\mathcal{W})^{-1}\mathcal{W}^{\top}\mathcal{Z}$.

With all the learned parameters, we can obtain future app service factor \hat{G} through Equation (3), obtain future demand competition factor \hat{B} through Equation (4), and obtain demand time series matrix \hat{Z} through Equation (6). Then, we apply Equation (1) to forecast download amount of each app.

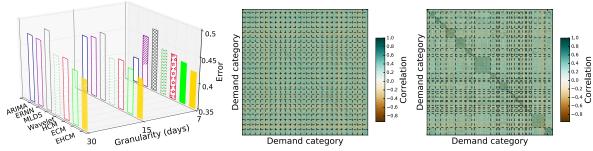
4 Experiment

4.1 Set-up

Dataset

From a commercial app analysis platform, we collect the download amount data for over 20 thousand apps on the iOS platform from Jul. 10th 2015 to Jan. 9th 2017 (550 days) in the Chinese market. The dataset consists of the daily download amount and the category of each app. On the iOS platform, the apps are classified into 25 categories: Entertainment, Games, Books, Medical, etc. Considering each day as one time step and filtering the apps that have zero download for 2 continuous months during the last 250 days, we obtain a download matrix $X^{''} \in R^{20,781 \times 550}$ for 20,781 apps. We apply a moving window with a 365-day length and a 37-day moving step on $X^{''}$ and obtain 6 sub-matrices $X^{'} \in R^{20,781 \times 365}$.

Since app developers or app investors may be interested in the future prospect of marketing in different granularities, we forecast the future app download amount in one week, half a month, and a month, respectively. We divide each sub-matrix \mathbf{X}' into a training matrix and testing matrix at different positions for such settings. For example, when we need to evaluate the forecasting accuracy of half a month in the future, we divide each \mathbf{X}' into $\mathbf{X} \in R^{20,781 \times 350}$ as the training data and $\hat{\mathbf{X}} \in R^{20,781 \times 15}$ as testing data.



(a) Average results at 3 different granularities (b) App service correlation among 25 (c) App service correlation among 25 default categories categories learned in the experiment

Figure 4: Average error results and app service correlation comparison

Error		Segment amount L					
		7	11	15	19	23	27
M	5	0.465	0.451	0.441	0.434	0.449	0.459
	10	0.459	0.449	0.440	0.428	0.433	0.447
	15	0.457	0.449	0.430	0.427	0.431	0.436
	20	0.450	0.443	0.427	0.426	0.426	0.428
	25	0.457	0.448	0.436	0.442	0.429	0.434
	30	0.463	0.452	0.431	0.437	0.427	0.432

Table 1: Errors of EHCM when forecasting length is 15 days

Comparing Methods

The methods we compare with EHCM include:

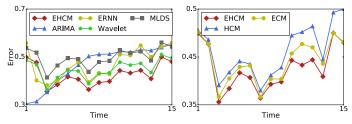
- ARIMA. ARIMA [Brockwell and Davis, 2016] is an effective single time series analysis algorithm. Here we apply the automatic ARIMA approach proposed in [Hyndman et al., 2007].
- ERNN. Elman Recurrent Neutal Networks (ERNN) [Chandra and Zhang, 2012] is a widely used three-layer recurrent neural network, which is capable of processing sequentially ordered stimuli.
- Wavelet. Wavelet Transform Prediction (Wavelet) [Renaud *et al.*, 2005] is a wavelet-based multi-scale autoregressive prediction method introduced in Section (3.2).
- MLDS. Multilinear Dynamical System (MLDS)[Rogers *et al.*, 2013] models a time series tensor as a multilinear projection from certain latent tensors. Here we set the dimension of the input tensor as two.

We also add two more compared methods in our experiment: HCM and ECM. HCM is the EHCM model without temporal evolution, in which we set the segment amount $L{=}1$ to investigate the importance of modeling evolution in app competition. ECM is the EHCM model only having a one-level competition, in which we only apply a traditional nonnegtive matrix factorization method (cf. Section 3.2) to investigate the effect of hierarchical competition.

Metric

The metric we use to evaluate the app download forecasting error is Relative Euclidean Distance [Rogers *et al.*, 2013]:

$$RED_{(t)} = \sqrt{(\sum_{n=1}^{N} (\hat{X}_{n,t} - \tilde{X}_{n,t})^2)/(\sum_{n=1}^{N} \hat{X}_{n,t}^2)},$$



(a) Compare with other methods

(b) Compare with sub-models

Figure 5: Daily forecasting results comparison

where \hat{X} and \tilde{X} are the app download matrix and forecasting matrix, respectively.

4.2 Results

We first present the results of EHCM compared with various methods. Figure 4(a) shows the average RED results of 6 test windows in three different forecasting granularities: 7 days (one week), 15 days (half a month), and 30 days (one month). We can observe that errors increase when the forecasting period becomes longer (from 7 days to 30 days) for all the models. When the granularity is 15 days and 30 days, EHCM has the best performance compared with other methods. When the granularity is 7 days, EHCM outperforms all the other methods except ARIMA. To investigate the underlying reason, we plot Figure 5, which shows the daily results (15 days) of one test window. We can see that ARIMA has the lowest error in the first 3 days, while EHCM has dominantly low errors afterwards. This shows that EHCM has a big advantage for long-term forecasting than short-term forecasting. ARIMA converges quickly to the mean of test data, as its error converges to 0.5 as shown in Figure 5(a). Moreover, Figure 5(a) also shows that ERNN's performance gradually becomes worse when the forecasting period is longer, which is also reflected in Figure 4(a). We can also oberve that HCM has worse performance than ECM, which demonstrates that capturing the competition evolution is more crucial than hierarchical patterns in forecasting, especially when forecasting period is longer (Figure 5(b)).

Next, we evaluate the performances of EHCM with different parameters. We vary the demand category amount M from 5 to 30, and vary the stabilization segment amount L from 7 to 27. Table 1 presents the average results for this experiment when the forecasting period is 15 days (where training period is 350 days). We can see that EHCM has the best performance when $M{=}20$ and $L{=}19$ or 23. When we fix L,

EHCM has the lowest error when $M{=}20$. This indicates that classifying all the apps into 20 categories according to the demand they cater to should be sufficient for the 25 categories used in the App Store. Moreover, we can also observe that EHCM perfoms better when $L{=}19$ or 23 for a fixed M. This indicates that EHCM have a better performance when the average length of each segment (18.5 or 15.2 days for $L{=}19$ or 23 respectively) is close to the test period (15 days).

The competition evolution pattern is crucial in EHCM, since it detects the time-varying competition relationships at both the app level and demand category level. Figure 6 shows the evolution of demand category competition matrix \mathbf{B}^l along one year when M=20 and L=23, where we plot 4 matrices: \mathbf{B}^1 , \mathbf{B}^8 , \mathbf{B}^{15} , and \mathbf{B}^{23} . We normalize the elements in these matrices into the range of [0, 1], making it easier for observation. These figures show that the competition among the 20 latent categories is evolving during the year, and the highlights of the diagonal lines represent each demand category inheriting most of its own demands during competition. To illustrate the latent demand categories more clearly, we show representative apps within several categories (here we assume app a_n mainly serves demand category m if $G_{n,m}$ is the largest value in row G_n averaged over all segments. The competitiveness of latent categories 13 to 20, mainly containing readers, English learning, English dictionary, weather, urban environment, stock-market, and exercise related apps, has a noteworthy decrease, while the advantage of latent category 8, consisting of apps on financial management, hospital check-in, flea market etc., fluctuates throughout the year.

In the App Store, apps are classified into 25 categories, e.g., Social, Health, Travel, etc. Empirically, apps in the same category will satisfy users' similar demand categories. To validate this intuition, we exploit the Pearson's Correlation (PC) among these apps' service patterns. Specifically, we compute the PC between each two apps in matrix G^{l} , which is the app service factor, and show the correlation in Figure 4(b). We add dotted lines to divide apps into different categories. We can observe that there are no apparent correlations among apps in each category. This means that the classification for these apps in the App Store is not rational enough if only considering how they cater to user demands. When we assume that app a_n belongs to category m if $\boldsymbol{G}_{n,m}^l$ has the largest value in row G_n^l , we can compute the correlations among these apps, which is shown in Figure 4(c). The obvious highlights along the diagonal clearly show a higher app service correlation with their categories defined in this approach.

5 Related Work

5.1 Time Series Evolution

The investigation of time series evolution is a fundamental task of weather forecasting, market analysis, user mobility prediction, and so on [Berrocal *et al.*, 2010; Kahn, 2010; Sadilek and Krumm, 2012]. Some state-of-the-art time series analysis methods, like ARIMA [Brockwell and Davis, 2016], Markov chain [Liao *et al.*, 2012], Recurrent Neural Networks [Chandra and Zhang, 2012], and Wavelet Transform based prediction [Renaud *et al.*, 2005], are widely applied in evolution analysis. All of these methods only focus

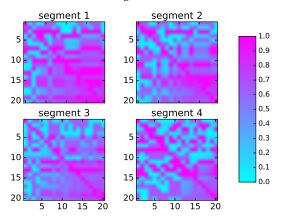


Figure 6: Demand competition evolution

on the analysis of a single series, neglecting the interactions among a group of series.

To resolve this disadvantage and explore the connections among a group of time series, [Dunlavy et al., 2011; Matsubara et al., 2012] combine matrix factorization with time series analysis for time-evolving datasets. They firstly decompose a matrix into latent factors and then apply time series forecasting methods on the temporal factor. [Rogers et al., 2013] present a multilinear dynamical system, which decomposes the tensor in each time step and utilizes the evolution-analysis of core tensor to predict future values. Nevertheless, these methods only consider one-level relationships among time series, ignoring the hierarchical interactions.

5.2 Product Forecasting

Products are the lifeblood of business and markets. Forecasting the sales of products helps to understand customer satisfaction and control product inventory, thus continuously improving the business in question. Therefore, product forecasting is always a crucial task in business management. [Yu et al., 2010] propose a regression model for sales prediction, considering the quality factors of online reviews of products. [Wu and Brynjolfsson, 2014] utilize a seasonal autoregressive method to model the relationship between queries submitted to Google's search engine and housing market indicators to predict housing prices and sales. For new product sales forecasting, a spatial divergence approach is applied in [Garber et al., 2004], taking into account both sales amount and spatial information of existing products. However, these methods neglect competition relationship among products.

6 Conclusion

We have proposed EHCM, a systematic method to integrate hierarchical competition evolution analysis with time series forecasting, and applied an evolutionary matrix trifactorization method to exploit hierarchical competition interactions among products. We have also explored the temporal patterns through a multi-scale wavelet prediction approach. Extensive experiments using a real-world dataset validate that EHCM outperforms state-of-the-art approaches in various forecasting granularities. Although EHCM is based on a dataset of app download amounts, we believe that the hierarchical competition evolution method can be easily generalized to other product sales forecasting applications.

References

- [Begon *et al.*, 2005] Michael Begon, Colin R Townsend, and John Lander Harper. *From individuals to ecosystems*. Blackwell Publishers Hoboken, 2005.
- [Berrocal et al., 2010] Veronica J Berrocal, Adrian E Raftery, Tilmann Gneiting, and Richard C Steed. Probabilistic weather forecasting for winter road maintenance. *Journal of the American Statistical Association*, 105(490):522–537, 2010.
- [Briand and Cohen, 1987] Frederic Briand and Joel E Cohen. Environmental correlates of food chain length. *Science*, 238(4829):956–961, 1987.
- [Brockwell and Davis, 2016] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. springer, 2016.
- [Chandra and Zhang, 2012] Rohitash Chandra and Mengjie Zhang. Cooperative coevolution of elman recurrent neural networks for chaotic time series prediction. *Neurocomputing*, 86:116–123, 2012.
- [Dunlavy *et al.*, 2011] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *TKDD*, 5(2):10, 2011.
- [Ernst et al., 2007] Floris Ernst, Alexander Schlaefer, and Achim Schweikard. Prediction of respiratory motion with wavelet-based multiscale autoregression. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 668–675. Springer, 2007.
- [Fu, 2011] Tak-chung Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181, 2011.
- [Garber *et al.*, 2004] Tal Garber, Jacob Goldenberg, Barak Libai, and Eitan Muller. From density to destiny: Using spatial dimension of sales data for early prediction of new product success. *Marketing Science*, 23(3):419–428, 2004.
- [Himberg *et al.*, 2001] Johan Himberg, Kalle Korpiaho, Heikki Mannila, Johanna Tikanmaki, and Hannu TT Toivonen. Time series segmentation for context recognition in mobile devices. In *Data Mining*, 2001. ICDM 2001, pages 203–210. IEEE, 2001.
- [Huisman and Weissing, 1999] Jef Huisman and Franz J Weissing. Biodiversity of plankton by species oscillations and chaos. *Nature*, 402(6760):407–410, 1999.
- [Hyndman *et al.*, 2007] Rob J Hyndman, Yeasmin Khandakar, et al. Automatic time series for forecasting: the forecast package for r. Technical report, Monash University, Department of Econometrics and Business Statistics, 2007.
- [Johnson *et al.*, 2013] Dominic DP Johnson, Michael E Price, and Mark Van Vugt. Darwin's invisible hand: Market competition, evolution and the firm. *Journal of Economic Behavior & Organization*, 90:S128–S140, 2013.
- [Kahn, 2010] Kenneth B Kahn. *New-Product Forecasting*. Wiley Online Library, 2010.

- [Levine, 1976] Stephen H Levine. Competitive interactions in ecosystems. *The American Naturalist*, 110(976):903–910, 1976.
- [Liao *et al.*, 2012] Zhung-Xun Liao, Po-Ruey Lei, Tsu-Jou Shen, Shou-Chung Li, and Wen-Chih Peng. Mining temporal profiles of mobile applications for usage prediction. In *Data Mining Workshops (ICDMW)*, 2012 IEEE, pages 890–893. IEEE, 2012.
- [Matsubara *et al.*, 2012] Yasuko Matsubara, Yasushi Sakurai, Christos Faloutsos, Tomoharu Iwata, and Masatoshi Yoshikawa. Fast mining and forecasting of complex timestamped events. In *Proceedings of the 18th ACM SIGKDD*, pages 271–279. ACM, 2012.
- [Percival and Walden, 2006] Donald B Percival and Andrew T Walden. *Wavelet methods for time series analysis*, volume 4. Cambridge university press, 2006.
- [Radosavljevic et al., 2016] Vladan Radosavljevic, Mihajlo Grbovic, Nemanja Djuric, Narayan Bhamidipati, Daneo Zhang, Jack Wang, Jiankai Dang, Haiying Huang, Ananth Nagarajan, and Peiji Chen. Smartphone app categorization for interest targeting in advertising marketplace. In Proceedings of the 25th International Conference Companion on WWW, pages 93–94, 2016.
- [Renaud *et al.*, 2005] Olivier Renaud, J-L Starck, and Fionn Murtagh. Wavelet-based combined signal filtering and prediction. *IEEE Transactions on Systems, Man, and Cybernetics*, Part B (Cybernetics), 35(6):1241–1251, 2005.
- [Rogers *et al.*, 2013] Mark Rogers, Lei Li, and Stuart J Russell. Multilinear dynamical systems for tensor time series. In *Advances in Neural Information Processing Systems*, pages 2634–2642, 2013.
- [Sadilek and Krumm, 2012] Adam Sadilek and John Krumm. Far out: Predicting long-term human mobility. In *AAAI*, 2012.
- [Sahney *et al.*, 2010] Sarda Sahney, Michael J Benton, and Paul A Ferry. Links between global taxonomic diversity, ecological diversity and the expansion of vertebrates on land. *Biology letters*, 6(4):544–547, 2010.
- [Soltani, 2002] Skander Soltani. On the use of the wavelet decomposition for time series prediction. *Neurocomputing*, 48(1):267–277, 2002.
- [Wu and Brynjolfsson, 2014] Lynn Wu and Erik Brynjolfsson. The future of prediction: How google searches fore-shadow housing prices and sales. In *Economic analysis of the digital economy*, pages 89–118. 2014.
- [Yu et al., 2010] Xiaohui Yu, Yang Liu, Xiangji Huang, and Aijun An. A quality-aware model for sales prediction using reviews. In *Proceedings of the 19th international conference on World wide web*, pages 1217–1218. ACM, 2010.
- [Zhuang et al., 2013] Fuzhen Zhuang, Ping Luo, Changying Du, Qing He, and Zhongzhi Shi. Triplex transfer learning: exploiting both shared and distinct concepts for text classification. In *Proceedings of the sixth ACM ICWSM*, pages 425–434. ACM, 2013.