Frontmatter

Editors

SIGAPP FY’15 Quarterly Report
S. Shin

A Message from the Editor
S. Shin

Selected Research Articles

Towards Statistical Modeling and Machine Learning Based Energy Usage Forecasting in Smart Grid
W. Yu, D. An, D. Griffith, Q. Yang, and G. Xu

Super Resolution Face Image Based on Locally Linear Embedding and Local Correlation
V. Nguyen, C. Hung, and X. Ma

Post Ranking in a Blogosphere: Algorithms and Evaluation
W. Hwang and S. Kim

Efficient Hibernation Resuming with Classification-based Prefetching Scheme for Embedded Computing Systems
C. Ho, S. Cheng, Y. Chang, Y. Chang, S. Hong, and C. Chang
SIGAPP FY’15 Quarterly Report

January 2015 – March 2015
Sung Shin

Mission

To further the interests of the computing professionals engaged in the development of new computing applications and to transfer the capabilities of computing technology to new problem domains.

Officers

<table>
<thead>
<tr>
<th>Role</th>
<th>Name</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chair</td>
<td>Sung Shin</td>
<td>South Dakota State University, USA</td>
</tr>
<tr>
<td>Vice Chair</td>
<td>Jiman Hong</td>
<td>Soongsil University, South Korea</td>
</tr>
<tr>
<td>Secretary</td>
<td>Michael Schumacher</td>
<td>University of Applied Sciences Western Switzerland, Switzerland</td>
</tr>
<tr>
<td>Treasurer</td>
<td>Tei-Wei Kuo</td>
<td>National Taiwan University, Taiwan</td>
</tr>
<tr>
<td>Webmaster</td>
<td>Hisham Haddad</td>
<td>Kennesaw State University, USA</td>
</tr>
<tr>
<td>Program Coordinator</td>
<td>Irene Frawley</td>
<td>ACM HQ, USA</td>
</tr>
</tbody>
</table>

Notice to Contributing Authors

By submitting your article for distribution in this Special Interest Group publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance by the editor
- to digitize and post your article in the electronic version of this publication
- to include the article in the ACM Digital Library and in any Digital Library related services
- to allow users to make a personal copy of the article for noncommercial, educational or research purposes

However, as a contributing author, you retain copyright to your article and ACM will refer requests for republication directly to you.
A Message from the Editor

Greetings to all and welcome to the spring issue of Applied Computing Review! This issue includes four selected papers presented at the 2014 ACM Research in Adaptive and Convergent Systems (RACS) held in Baltimore, United States. The purpose of the conference was to provide a professional forum to share novel ideas in modern computing systems, and I would like to show my gratitude to the authors for contributing the advanced methods in their research area.

ACR is available to everyone who is interested in the modern applied computing research trends. Our goal is to provide you with a platform for sharing innovative thoughts among professionals in various fields of applied computing. We are working with the ACM SIG Governing Board to further expand SIGAPP by increasing membership and developing a new journal on applied computing in the near future.

Lastly, I would gladly remind you once again that the 30th SAC (Symposium on Applied Computing) will be held in Salamanca, Spain, next month. The committee members and track chairs have worked diligently to prepare for this monumental year. I am confident that the conference will be a great success so please join and celebrate it all together. Your continuous cooperation and support will be appreciated. May all your dreams and wishes come true this year.

Sincerely,

Sung Shin
Editor in Chief & Chair of ACM SIGAPP

ACM FCRC

The 2015 Federated Computing Research Conference (FCRC) will be held in Portland, Oregon, in June 2015. It facilitates communication among researchers in various fields in computer science and computer engineering. More details about ACM FCRC can be found at http://fcrc.acm.org/.

Next Issue

The planned release for the next issue of ACR is June 2015.
Towards Statistical Modeling and Machine Learning Based Energy Usage Forecasting in Smart Grid

Wei Yu*, Dou An§, David Griffith†, Qingyu Yang§, and Guobin Xu*

* Towson University
Towson, MD 21252
wyu@towson.edu
gxu2@students.towson.edu

§ Xi’an Jiaotong University
Xi’an, Shaanxi, China
adkaka.an@gmail.com
yangqingyu@mail.xjtu.edu.cn

† National Institute of Standards and Technology
Gaithersburg, MD 20899
david.griffith@nist.gov

ABSTRACT
Developing effective energy resource management strategies in the smart grid is challenging due to the entities on both the demand and supply sides experiencing numerous fluctuations. In this paper, we address the issue of quantifying uncertainties on the energy demand side. Specifically, we first develop approaches using statistical modeling analysis to derive a statistical distribution of energy usage. We then utilize several machine learning based approaches such as the Support Vector Machines (SVM) and neural networks to carry out accurate forecasting on energy usage. We perform extensive experiments of our proposed approaches using a real-world meter reading data set. Our experimental data shows that the statistical distribution of meter reading data can be largely approximated with a Gaussian distribution and the two SVM-based machine learning approaches to achieve a high accuracy of forecasting energy usage. Extensions to other smart grid applications (e.g., forecasting energy generation, determining optimal demand response, and anomaly detection of malicious energy usage) are discussed as well.

Categories and Subject Descriptors
C.4 [Performance of Systems]: Modeling Techniques

General Terms
Measurement, Performance

Keywords
Statistical Modeling Analysis, Energy Usage Forecasting, Machine Learning, Real-world Meter Reading Data, Smart Grid

1. INTRODUCTION
With recent developments in sensing, information, and communication technologies, the smart grid becomes a proposing system that makes the power grid more efficient, reliable, and secure. To efficiently deliver energy resources in the smart grid, an energy resource management strategy needs to be developed to balance the energy demand and supply [28]. Nonetheless, developing effective energy resource management schemes is challenging due to the entities on both the demand and supply sides experiencing numerous fluctuations. For example, on the supply side, fluctuations could come from distributed renewable energy resources due to solar irradiance, wind speed, etc. On the demand side, numerous effects, including natural disasters, plug-in vehicles, personal habits of using energy, weather and temperature, etc., could make it difficult to predict energy usage.

To address these issues, in this paper, we develop techniques to effectively manage energy resources and usage in order to adapt to fluctuations. Particularly, to balance energy demand and supply, we develop effective techniques to accurately model and forecast the amount of energy generation and demand over time. Therefore, the issue of quantifying fluctuations on the energy demand side can be addressed. It is worth noting that the techniques developed in this paper can be applied to the energy generation side as well. We also conduct the modeling analysis to derive a statistical model of energy usage and develop several machine learning based approaches to perform accurate forecasting of energy usage. The extensions to areas, including forecasting energy generation, determining optimal demand response, and anomaly detection of malicious energy usage, are discussed as well.

To summarize, the key contributions of this paper are as follows:

• First, using the real-world meter reading data set from Stanford University that consists of meter readings from houses over 200 days as described in [18], we study the statistical distribution of real-world meter reading data using non-parametric tests, including the Shapiro-Wilk test [31] and the Quantile-Quantile plot normality test [9]. The experimental data shows that the distribution of meter reading data can be approximated with a Gaussian distribution.

• Second, we develop machine learning based approaches to conduct accurate energy usage forecasting. Partic-
ularly, we consider the standard Radial Basis Function (RBF) based SVM, the Least Squares (LS) based SVM, and the Backward Propagation Neural Network (BPNN). In addition, we conduct extensive experiments using the aforementioned real-world meter reading data set to validate the effectiveness of these approaches. The experimental data shows that the two SVM-based approaches achieve a higher prediction accuracy than the BPNN based approach.

- Third, the techniques that we developed in this paper can be expanded to other areas as well, including the modeling and forecasting of energy generation, the optimal demand response, and anomaly detection of malicious energy usage. Using the prediction of wind speed as an example, the use of the SVM machine learning based approach can be used to effectively conduct the forecasting on the distributed energy resources in the energy supply side. In addition, the developed statistical modeling and forecasting results can be applied to derive the upper and lower bounds of energy usage and determine optimal demand response as well as anomaly detection of malicious energy usage.

The remainder of this paper is organized as following: The literature review is conducted in Section 2. The problem of balancing the energy demand supply and the developed approaches to perform the statistical modeling and forecasting of energy usage are presented in Section 3. The experimental results using real-world meter reading data set to validate the effectiveness of the developed approaches are shown in Section 4. The extensions of the work to other areas (e.g., forecasting energy generation, determining optimal demand response, and performing anomaly detection of malicious energy usage) are presented in Section 5. Finally, the conclusion is drawn in Section 6.

2. RELATED WORK

A number of research efforts have been conducted to improve energy transmission and distribution efficiency [6, 10, 25, 5, 20, 11]. For example, Guan et al. [10] proposed minimizing the overall cost of electricity and natural gas for a building operation. Chen et al. [5] proposed an optimal demand response scheme that could match electricity supply and shape electricity demand accordingly in both competitive and oligopolistic markets.

The challenges associated with the forecasting and demand response associated with energy usage were also discussed in [23]. Broadly speaking, energy usage forecasting can be categorized into short-term, medium-term, and long-term forecasting. For example, Hong et al. [13] adopted a multiple linear regression mechanism for conducting short-term forecasting, which provides an interpretability of the behavior of the electricity usage in the service territory. A semi-parametric additive model proposed by Fan et al. in [8] used a regression mechanism and investigated the nonlinear relationships between energy usage data and variables in the short-term time period. In addition, a human-machine co-construct intelligence framework was proposed in [14] to determine the horizon year load for a long term load forecasting.

Machine learning methods such as SVM and neural networks have been used in carrying out forecasting [2, 32, 37, 35, 1, 19, 15, 29]. For example, Shi et al. [32] developed a SVM-based model for one-day-ahead power output forecasting using the characteristics of weather classification.

Different from the existing research efforts, using the real-world meter reading data set [18], non-parametric tests were used to investigate the statistical distribution of energy usage. To the best of our knowledge, our paper is one of the first to validate that the statistical distribution of meter reading data can be largely approximated with a Gaussian distribution. In addition, two SVM and neural network based approaches were used to systematically perform the energy usage forecasting and the effectiveness of these machine learning approaches was systematically evaluated and compared. The findings from the paper can be extended to other areas, including the energy generation forecasting, the optimal demand response, and anomaly detection of malicious energy usage.

3. OUR APPROACHES

In this section, we first present an overview of the problem and our proposed approaches. We then describe the real-world data set and develop the non-parametric test based approaches to carry out statistical modeling. Finally, we discuss machine learning based approaches to perform energy usage forecasting.

3.1 Overview

In the smart grid, the electric power from generators can be delivered through the power grid to large geographical areas. High efficiency in power production and energy utilization can be realized through monitoring and control of power transmission and distribution processes. How to manage both bulk and distributed energy resources and the consumption levels of consumers to balance energy supply and demand is important. Nonetheless, developing effective management techniques to balance energy supply and demand is a challenging task because both sides experience various fluctuations.

To address this issue, we developed a statistical analysis and model of energy usage in this paper. We also developed machine learning based approaches to conduct accurate forecasting of energy usage. For the statistical modeling, we use two types of non-parametric test approaches to derive the distribution of energy usage based on real-world meter reading data. For forecasting energy usage, we developed several machine learning based approaches to conduct accurate energy usage forecasting. Energy providers can use these techniques to schedule energy generation and to make energy transmission and distribution efficient.

3.2 Real-world Energy Usage Data Set

We now introduce the real-world data set from Stanford University, which consists of meter readings from houses over 200 days (between February 2010 and October 2010) [18]. In this data set, weather information (e.g., mean temperature) for each 24 hour period is taken from archival data at Weather Underground website. We use meter readings and weather information for 283 houses in our experiments in Section 4.
An example of meter reading is shown in Table 1. From the table, each house is assigned an ID. The meter reading data for energy usage is measured hourly. The fields contained in the data set are shown in Table 2, which consists of the house ID, time, energy usage, the maximum, mean, and minimum value of temperature, and maximum and mean value of wind speeds. The house size (i.e., the area) is included as well. As an example, the information shown in Table 3 is the data associated with house 1001 that is for a rented townhouse, built in 2004, with 92.90 – 139.35 sq. meters. In Table 4, we show an example of meter readings for energy usage and weather information at 2 p.m. from days 100 to 102 for house 1001. On day 100, the energy usage is 2.20 kilowatt hours (KWhs) and the mean values of temperature and wind speed are 50 Fahrenheit degrees (F) and 20.92 Km Per Hour (KmPH), respectively.

### Table 1. Data Range and Time Scale

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID of Houses</td>
<td>1-283</td>
</tr>
<tr>
<td>Time Interval</td>
<td>Hourly</td>
</tr>
<tr>
<td>Time Span</td>
<td>Approximately 200 days</td>
</tr>
<tr>
<td>Number of Data Points</td>
<td>Approximately 4800 (one per hour)</td>
</tr>
</tbody>
</table>

### Table 2. Data Fields

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Temp</td>
<td>Day-of-Year</td>
</tr>
<tr>
<td>Mean Temp</td>
<td>Hour</td>
</tr>
<tr>
<td>Min Temp</td>
<td>Electricity Consumption</td>
</tr>
<tr>
<td>Max Wind Speed</td>
<td>Hour</td>
</tr>
<tr>
<td>Mean Wind Speed</td>
<td>Hour</td>
</tr>
<tr>
<td>Min Wind Speed</td>
<td>Hour</td>
</tr>
</tbody>
</table>

### Table 3. Sample of House Information

<table>
<thead>
<tr>
<th>ID</th>
<th>Building, Year, Rent</th>
<th>Const.</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1001</td>
<td>Townhouse, duplex</td>
<td>Rent</td>
<td>92.90-139.35 sq. meters</td>
</tr>
<tr>
<td>1002</td>
<td>Single Family Detached House</td>
<td>Own</td>
<td>185.81-232.37 sq. meters</td>
</tr>
</tbody>
</table>

### 3.3 Statistical Model of Energy Usage

To establish a statistical model of energy usage, we develop two non-parametric test based approaches to derive the statistical distribution of energy usage based on the aforementioned real-world meter reading data. We use a non-parametric test to carry out the analysis of the energy usage data. For a set of one-dimensional data, common non-parametric test approaches include the Shapiro-Wilk test [31] and the Kolmogorov-Smirnov (K-S) test [12]. It is worth noting that because the K-S test demands the pre-knowledge of the distribution of the sample data, the test result will not be credible if the population’s Cumulative Distribution Function (CDF) is estimated from the sample data. It is worth noting that the predetermined CDF of the meter data is not known, so we consider the Shapiro-Wilk test to test the distribution of the sample data. We also use another non-parametric test approach, which is also called Quantile-Quantile (Q-Q) plot normality test, to confirm the distribution of meter reading data [36]. On the plot, when two data sets are identically distributed, the Q-Q plot will be shown a line. Then, we know that the greater the departure from the reference line, the greater the chance that the two data sets are drawn with different distributions.

### 3.4 Machine Learning Based Approaches for Energy Usage Forecasting

To accurately forecast energy usage in the smart grid, we use the following machine learning based approaches: neural network based machine learning, the standard SVM and the least squares SVM.

#### 3.4.1 Neural Network Based Machine Learning

There are a number of research efforts on neural networks [16, 17]. A classic example of one of these neural networks is the Backward Propagation (BP) neural network, which consists of three layers: input layer, hidden layer, and output layer. Note that the error between real value and estimated value will be propagated backward from output layer to hidden layer and from hidden layer to input layer. The error of each layer can be re-estimated and the weights can be assigned correspondingly. Parameters for neural networks are set through a training process that uses known data sets as input. After the training process, the trained model can then be used to carry out forecasting.

#### 3.4.2 Standard SVM and LS-SVM

The standard SVM was originally proposed by V. N. Vapnik et al. [7]. Generally speaking, the SVM is one of the popular methods to efficiently classify data and to build a classifier, which can be further used to carry out forecasting. In SVM, the data and associated features can be treated as a point and vectors in multi-dimensional space. The basic principle of a standard SVM is to find a hyperplane, which could divide the points into different spaces. By doing so, we can classify data into different categories [27]. In order to minimize the classification error, the proper hyperplane needs to be determined.

The least squares SVM that is also denoted as LS-SVM is an enhanced SVM [33]. In a LS-SVM, there are two major enhancements in comparison with the standard SVM. First, the inequality constraints are substituted by equality constraints. Second, the squared loss function is used in the objective function [34]. In our experiment, we use the radial basis function as the kernel function in LS-SVM due to its wide use.

#### 3.4.3 Workflow for Energy Usage Forecasting

As shown in Figure 1, the main process of machine learning based approaches can be divided into the following three steps: (i) data preprocessing, (ii) input feature selection, and (iii) energy usage forecasting. In the following, we describe these steps in detail.

**Step 1: Data Preprocessing.** To make our data more suitable for energy forecasting, data preprocessing needs to be performed first. Note that the real-world energy usage data
cannot be directly used due to the following reasons: (i) the data is lacking attribute values that could be caused by the measurement noise of meters; and (ii) existing noises or bad data could be deviated from the norm values due to malfunction or unexpected events in the system (e.g., failures, power cuts, and/or natural disasters, etc.).

To address these issues, we introduce an interpolation mechanism to fill the missing values in the experimental data set and smooth incorrect data values with the average value of points around them. The missing data is filled using a linear interpolation mechanism. For bad data, because energy usage has continuity, the data located before and after adjacent time periods should not have a distinct change. Therefore, the average value in a continuous period of time can be considered as a baseline. Then, data beyond the baseline could be treated as bad data. Our experiments on the aforementioned real-world data set shows that the percentages for missing values and bad data are 3.08 % and 3.26 % on average, respectively. Therefore, around 6.34 % of data in the real-world data set used in this paper needs to be reprocessed using the mechanism discussed above.

**Step 2: Input Feature Selection.** As described in Section 3.2, various factors (e.g., weather and/or user’s behaviors) can affect levels of energy usage. To achieve accurate energy usage forecasting, the selection of input features is important. The common way for feature selection is to choose related input variables such as the energy usage data in the past few days, humidity, temperature, and wind speed.

Recall that each component in the training data set is denoted as a feature. Here, the type of feature should be considered to include in input vectors. In this paper, the input features consist of two basic features: (i) hourly historical energy usage, and (ii) weather information. For the historical energy usage data, the measurements of the previous three hours are selected as input elements. The relieff [30] algorithm was used to determine the importance of features. In particular, the algorithm appraises features one by one and assigns a weight to each feature to indicate its importance. The larger the weight, the higher the importance of the feature. Table 5 and Table 6 illustrate the weight of all features related to weather in the experimental data and the results of the input features selection, respectively. In the experiments, the top three largest weights are selected and the energy usages in three hour timespans are chosen as input features for each house. To achieve rapid convergence during the training process, the network input data and the corresponding output data for the forecasting models are normalized such that all data is mapped into the range of [−1, 1].

**Table 5. Weight of Weather Features**

<table>
<thead>
<tr>
<th>ID</th>
<th>Max_T</th>
<th>Mean_T</th>
<th>Min_T</th>
<th>Max_W</th>
<th>Mean_W</th>
</tr>
</thead>
<tbody>
<tr>
<td>1002</td>
<td>0.2539</td>
<td>0.2367</td>
<td>0.5011</td>
<td>1.4756</td>
<td>0.6090</td>
</tr>
<tr>
<td>1035</td>
<td>-0.0004</td>
<td>0.0060</td>
<td>0.0032</td>
<td>-0.0032</td>
<td>0.0090</td>
</tr>
<tr>
<td>1044</td>
<td>0.0013</td>
<td>0.0015</td>
<td>0.0013</td>
<td>0.0006</td>
<td>0.0019</td>
</tr>
</tbody>
</table>

*T denotes temperature, W denotes wind speed*

**Step 3: Energy forecasting with SVMs.** After the feature selection, the energy usage data should be divided into two parts: (i) training set, and (ii) testing set. The training set is used to train the learning models and the optimal setting for parameters. The important parameters include the width of ε-sensitive tube ε and the error cost C, which are discussed in Section 3.4. After completing the above process, a trained SVM model is complete. Then, the trained model is ready to predict future energy usage.

Note that the training process of SVM can be formulated as solving a quadratic programming (QP) problem, which is optimized by a numerical method. The time complexity of the QP problem is of $O(n^3)$, where $n$ is the number of training examples. For LS-SVM, the QP problem can be transformed into linear equations, thus the time complexity reduces to $O(n^2)$. Therefore, the time complexity of SVMs increases with an increase of training examples, which will not be correlated with a class of energy consumers. It worth noting that the development of distributed computing, parallel computing, and cloud computing can be used to speed up the training and decision process described in the paper.

4. **Performance Evaluation**

In this section, we introduce the performance evaluation results. We first introduce the experimental setup and then present the results of statistical modeling and energy usage

---

**Figure 1. Workflow of Machine Learning Based Energy Usage Forecasting**
Based on the real-world meter reading data set described in Section 3.2, we carried out extensive experiments to evaluate the effectiveness of our developed statistical modeling and machine learning based energy usage forecasting approaches. MATLAB R2010b\(^3\) was used to implement our developed approaches and the experiments were performed on a laptop PC (Centrino Duo, 2.3 GHz, 3 GB RAM). The toolkit LIBSVM in Matlab [4], a library for SVMs that includes the implementation of both SVM and LS-SVM, was used in our experiments. For comparison purposes, the neural network toolbox in Matlab was also used to evaluate the performance of the Backward Propagation (BP) neural network based forecasting approach, one of the classical neural networks that consists of three layers: input layer, hidden layer, and output layer [17].

4.1 Results of Statistical Modeling

To perform the statistical modeling of energy usage, the two non-parametric test approaches: Shapiro-Wilk test and Q-Q plot normality test as we discussed in Section 3.3 are used. In our experiments, the meter reading measurements over the following three time windows are aggregated: (i) morning (8:00-12:00), (ii) afternoon (14:00-18:00), and (iii) evening (20:00-24:00). Due to the space limitations, we only used. In our experiments, the meter reading measurements that we discussed in Section 3.3 are necessary for the purpose.

The experimental data shows that the meter readings of 148 houses can be approximated by a Gaussian distribution. In addition, more than 40% of the remaining 52 houses contain a number of 0 values and error information, which largely deviate from the normal values, leading to the failures of the tests.

The Shapiro-Wilk test [31] with a significance level (\(\alpha = 0.05\)) is used for the measurements in individual time windows. It is worth noting that \(\alpha\) is defined as the probability that a Gaussian distribution approximation is mistakenly rejected whereas it is actually true. Here, we consider two hypotheses: (i) \(H_0\): the data follows a Gaussian distribution; and (ii) \(H_1\): the data does not follow a Gaussian distribution. The \(P\)-value, in contrast to the threshold \(\alpha\), is computed based on the test statistics, which can be denoted as the probability, in the case of the null hypothesis \(H_0\), of sampling results being equal to or being closer to the actual sampling results. As such, when the \(P\)-value is less than the predetermined significance level \(\alpha\), the observed results are be highly unlikely under the null hypothesis.

In our experiments, the \(P\)-value obtained from the meter reading measurements for the morning, afternoon, and evening windows are illustrated in Table 7. As shown in the table, in addition to the \(P\)-value for the morning meter reading measurements at house 1002, the remaining \(P\)-values are larger than the threshold of 0.05. Therefore, the energy usage in morning, afternoon, and evening windows of these houses can be approximated with a Gaussian distribution. It is worth noting that the Shapiro-Wilk test on morning data measurements from house 1002, in which the \(P\)-value is 0.000813, is an example of a failure case. As the \(P\)-value is far less than the threshold of 0.05, the morning data from house 1002 cannot be approximated with a Gaussian distribution.

The Q-Q plot normality test [36] is also used to test the distribution of meter reading measurements. As an example, the energy usage in house 1002 in three time windows is shown in Figure 2. The trend of points in Figures 2(b) and 2(c) has a higher degree of approximation to a straight line than the one in Figure 2(a), which indicates that the energy usages at noon and evening times can be better be approximated with the Gaussian distribution. This is because the closer the points are to a line, the closer the reading is to a Gaussian distribution. In Figure 2, there is significant deviation in the quantiles associated with the tails of the distribution whereas there is close agreement near the median. To summarize, the results of the two statistical test approaches draw the same conclusion, that is, the meter reading measurements for the three time windows at the three houses can be approximated with a Gaussian distribution.

4.2 Results of Energy Usage Forecast

Experiments based on the real-world meter reading data set used in this paper were conducted to validate the effectiveness of two types of SVM presented in Section 3 and BP neural network based approaches in terms of the accuracy of energy usage forecasting. In our experiments, based on the models learned through the training process from the historical energy usage of the past 500 hours, we show the the accuracy of energy usage forecasting in the next 48 hours.

To measure the accuracy of forecasting, the following three metrics are considered: (i) \(MAPE\) (Mean Absolute Percentage Error), (ii) \(MSE\) (Mean Square Error), and (iii) Coefficient of Regression \(\gamma^2\), which are used to measure the error between the actual and predicted energy usage. These metrics are defined as follows: $MAPE = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$, $MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$, and $\gamma^2 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{y})}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$, where \(y_i\), \(\hat{y}_i\), and \(\bar{y}\) are actual value, forecasted value, and mean value of the actual value, respectively.

We conducted a large number of experiments on meter reading data for 200 houses. Due to space limitations, only a limited number of results are shown here for demonstration purposes. Based on the workflow showed in Section 3.4.3, the generic optimization mechanism provided by the LIB-
SVM toolkit [4] is used to select key parameters for the SVM, including the width of insensitive tube $\varepsilon$ and the cost of error $C$. Table 8 shows the forecasting accuracy of the two SVM based approaches in comparison with the BP neural network based approach (denoted as BPNN). From this table, the standard SVM based approach achieves the MSE at a magnitude of $10^{-4}$ and the highest coefficient of 0.88 in comparison with the LS-SVM and BPNN based approaches. For the LS-SVM based approach, all its MAPE values are smaller than 10% whereas the MSE values are around 0.01. Further, the coefficient of regression approaches 0.84, which is better than the one achieved by the BPNN based approach. This can be explained as the neural network can easily fall into a local minimum instead of the global mini-
mum, leading to a lower accuracy of prediction.

### Table 8. Effectiveness of Forecasting Results

<table>
<thead>
<tr>
<th>ID</th>
<th>Index</th>
<th>SVM</th>
<th>LS-SVM</th>
<th>BPNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1002</td>
<td>MSE</td>
<td>4.956e-04</td>
<td>0.0015</td>
<td>0.0032</td>
</tr>
<tr>
<td></td>
<td>MAPE(%)</td>
<td>6.9435</td>
<td>9.9221</td>
<td>13.6712</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>8.2092e-04</td>
<td>0.0039</td>
<td>0.0278</td>
</tr>
<tr>
<td>1035</td>
<td>γ²</td>
<td>0.7871</td>
<td>0.6939</td>
<td>1.0157</td>
</tr>
<tr>
<td></td>
<td>MAPE(%)</td>
<td>5.7728</td>
<td>9.5013</td>
<td>11.2417</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>6.368e-04</td>
<td>0.0341</td>
<td>0.0174</td>
</tr>
<tr>
<td>1044</td>
<td>γ²</td>
<td>0.8819</td>
<td>0.8405</td>
<td>0.3345</td>
</tr>
<tr>
<td></td>
<td>MAPE(%)</td>
<td>4.3568</td>
<td>10.1023</td>
<td>10.6735</td>
</tr>
</tbody>
</table>

### Table 9. Overall Forecasting Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Statistics</th>
<th>MAPE</th>
<th>γ²</th>
<th>MSE</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>Mean</td>
<td>7.1261%</td>
<td>0.7593</td>
<td>0.0037</td>
<td>335.39</td>
</tr>
<tr>
<td></td>
<td>Variance</td>
<td>0.0004</td>
<td>0.0144</td>
<td>0.0009</td>
<td></td>
</tr>
<tr>
<td>LS-SVM</td>
<td>Mean</td>
<td>14.5649%</td>
<td>0.6219</td>
<td>0.0321</td>
<td>36.22</td>
</tr>
<tr>
<td></td>
<td>Variance</td>
<td>0.002</td>
<td>0.0128</td>
<td>0.0014</td>
<td></td>
</tr>
<tr>
<td>BPNN</td>
<td>Mean</td>
<td>16.8356%</td>
<td>0.4338</td>
<td>0.0732</td>
<td>29.28</td>
</tr>
<tr>
<td></td>
<td>Variance</td>
<td>0.007</td>
<td>0.0130</td>
<td>0.0014</td>
<td></td>
</tr>
</tbody>
</table>

The accuracy of energy usage forecasting for all 283 houses, including the statistical mean and standard derivation of MAPE, γ², and MSE for the three machine learning approaches, are demonstrated in Table 9. From this table, we can observe that the SVM achieves the highest forecasting accuracy and the BPNN achieves the worst forecasting accuracy. In addition, the time overhead of these machine learning based forecasting approaches, which is defined as the total time taken for inputting data, preprocessing data, selecting features, conducting training process, and generating forecasting results based on a training model for a single house, is evaluated. The experiments were conducted on a laptop PC (Centrino Duo, 2.3GHz, 3GB RAM). As shown in Table 9, the time overhead for the SVM, LS-SVM, and BPNN are 335.39 s, 36.22 s, and 29.28 s, respectively.

It is worth noting that in order to further improve the time efficiency, more powerful PC, conducting forecast using low level language (instead of using MATLAB), and leveraging techniques (e.g., cloud computing and parallel computing) can be used. For the SVM, time overhead is much greater than that of the LS-SVM and BPNN as the genetic algorithm optimization mechanism is used to select the key parameters for the SVM, including the width of insensitive tube ε and the cost of error C. For the LS-SVM, as explained before, two major enhancements of the LS-SVM in comparison with the standard SVM: (i) using equality constraints instead of inequality constraints, and (ii) using square loss function that can significantly simplify the complexity of the problem solving process, leading to a smaller processing time for carrying our energy usage forecasting.

In Figures 3 and 4, the accuracy of forecasting for the three machine learning approaches on houses 1002 and 1035 is demonstrated. As we can see from these figures, the blue and red curves represent the actual energy usage and forecasted energy usage, respectively. The blue curve and red curve for the SVM based approaches are highly coincidental with each other as shown in Figures 3(b)(c) and 4(b)(c) whereas the results of the BPNN based approach are shown in Figures 3(a) and 4(a). A higher consistency between the real data and forecasted data in the SVM based approaches indicates that the forecast of the SVM based approaches are more accurate than the BPNN based approach. In addition, note that the blue and red curves in the SVM based approaches almost follow the same trend, indicating the forecasted results of these approaches are accurate. For the two SVM based approaches, because a generic algorithm is used to obtain the optimal ε and C, the standard SVM based approach actually achieves a higher accuracy than the experimental data shown in Figure 5.

### 5. EXTENSION

In this section, the extensions are made from the following aspects: the modeling of energy generation, the optimal demand response, and anomaly detection of malicious energy usage.

#### 5.1 Modeling of Energy Generation

The distributed energy resources are inherently stochastic. Using wind energy as an example, the total wind energy flowing through an imaginary area A at time t can be formalized as: 

\[ E = \frac{1}{2} \rho A v^2 \]

where \( \rho \) is the density of air and \( v \) is the wind speed. Here, the wind energy \( E \) is highly
correlated with the wind speed $v$. Therefore, the forecasting of wind speed is one critical issue before wind energy resources can be broadly integrated in the smart grid.

Using the prediction of wind speeds as an example and applying it to the modeling approach developed in this paper, we are able to improve the ability of forecasting distributed energy resources at the energy supply side. Similar to the prediction of energy usage shown in Sections 3.4.2 and 4, the standard SVM machine learning approach can be used to carry out the prediction of wind speeds. We conducted experiments on the wind speed data of 193 days at No. 1002 house. Recall that as shown in Table 2, the wind data in the real-world data set used in this paper consists of both the max wind speed and mean wind speed for a day. The maximum and mean value of wind speeds in three days are selected as the input features in the standard SVM machine learning approach. The learning and forecasting process follows the same workflow as we described in Section 3.4.3. The mean value of wind speeds of the next two weeks is used to test the accuracy of forecasting. Figure 6 illustrates the accuracy of wind speed forecasting. The error metrics defined in Section 4 are also used to evaluate the accuracy of forecasting. The results of error metrics are 9.3876, 1.0339, and 0.5392, respectively, showing that the standard SVM machine learning approach could achieve a high accuracy of predicting wind speeds.

5.2 Optimal Demand Response

The results developed in this paper can also be used to determine optimal demand response, which allows customers to obtain real time energy prices and enables load shifting and reduction. In the following, we show an example of how to integrate our developed modeling results into the optimization model originally proposed in [5] for conducting optimal demand response. In [5], Chen et al. derived an efficient equilibrium based on the upper and lower bounds of customer’s energy usage in a competitive market. Nonetheless, their original work did not show how to derive those bounds.

In the following, we briefly show how to apply the results developed in Section 3.2 to determine the optimal demand response. Without loss of generality, we assume that a power grid system consists of $N$ customers, who are served by a power generator. On the demand side, let the power load of each customer be $q_i(t)$ at time $t$. Then, in a time window $[1 : T]$, the bounds for minimum and maximum total energy usages, denoted as, $Q_-$ and $Q_+$, can be derived. On one hand, based on the results of the energy usage forecasting, the $Q_-$ and $Q_+$ in a near future time window are derived. In this way, the bound can be precise and is suitable for a short-time demand response process. On the other hand, based on the result of the developed statistical modeling analysis, the bounds in each time window can be derived as well. It is worth noting that bounds based on statistical modeling analysis are more general and suitable for a long-term demand response process. Choosing either the long-term bound or short-term bound can be determined by the time scope of demand response process. In the following, the bounds based on the statistical modeling are used as an example to demonstrate our idea.

Denote the mean and the standard deviation of energy usage as: $\bar{X} = \frac{1}{T} \sum_{t=1}^{T} q_i(t)$ and $S_q = \frac{1}{T} \sum_{t=1}^{T} (q_i(t) - \bar{X})^2$, respectively. Based on the statistical modeling results developed by this paper, $Q_-$ and $Q_+$ can be derived through the interval estimation mechanism [26] and are given by, $Q_- = \bar{X} - t_2(T - 1) \frac{S_q}{\sqrt{T}}$, and $Q_+ = \bar{X} + t_2(T - 1) \frac{S_q}{\sqrt{T}}$, where $t_2(T - 1)$, $\bar{X}$ and $S_T$ are the upper quantile fractile of student t distribution at the confidence level of $\alpha$, mean value and standard deviation, respectively. Then, assume that each user $i$ satisfies the following constraints in $[1 : T]$, $\sum_{t=1}^{T} q_i(t) \geq Q_-$, where $i \in N$, and $\sum_{t=1}^{T} q_i(t) \geq Q_+$, where $i \in N$. For each user $i$, a utility function: $U_i(q_i, t)$ is defined to measure its satisfaction for the energy service, supplied by the energy generator, where $q_i$ is the energy usage at time $t$. We also assume that $U_i(q_i, t)$ is continuously differentiable and increasing with respect to $t$ monotonically.

On the supply side, depending on the state of the power grid, the energy price will be dynamic over time. Assume that the energy generator has a cost of $C(Q, t)$ when it supplies energy $Q$ at time $t$. We also assume that $C(Q, t)$ increases with respect to $Q$ and the marginal cost increases with respect to $Q$.

5.3 Anomaly Detection of Malicious Energy Usage

In an energy resource management system, it is important to report energy usage information from consumers to the utility supply. Nonetheless, this decision process could be impacted by an adversary, who might compromise meters and launch false data injection attacks to disrupt the smart grid operations [22, 21, 3]. Therefore, the detection of false data injection attacks becomes a critical issue. Note that our developed energy usage forecasting can be leveraged to carry out anomaly detection. To be specific, we can compute the lower and upper bounds of energy usage in a near future time window and use them as the baseline profile for conducting anomaly detection. In the following, we briefly demonstrate how to use our statistical modeling analysis results to detect malicious energy usages.

Based on the results in Section 3.2, we now present a hypothesis testing based detection scheme. We consider two hypotheses: (i) $H_0$: the measurement is valid, and (ii) $H_1$: the measurement is under attack. Based on our statistical modeling results, we assume that the energy usage measurements $X = (X_1, X_2, \ldots, X_n)$ in the three time windows (i.e., morning, noon, and evening) follow the Gaussian distribution $N(\mu, \sigma^2)$, in which $\mu$ and $\sigma$ are all unknown parameters and $n$ is the total number of measurements.

It is worth noting that the malicious measurement’s deviation from the mean value can be treated as noise and the value of $\mu$ and $\sigma$ are unknown to the detection system. Therefore, we consider that the standard deviation of samples, denoted as $S_n = \frac{1}{n - 1} \sum (X_i - \bar{X})^2$, can reflect the dispersion of difference between the compromised measurement and the normal one. After letting $T = \frac{S_n \mu}{\sqrt{\tau}}$, we have $T \sim t(n - 1)$. Based on this, the hypothesis test can be formalized as, $T \overset{H_0}{\gtrless} \tau$, where $\tau = t_2(n - 1)$ is the threshold determined by considering the null hypothesis given a certain false positive rate $\alpha$. 

APPLIED COMPUTING REVIEW MAR. 2015, VOL. 15, NO. 1
To evaluate the effectiveness of anomaly detection based on hypothesis test, we choose the following two metrics, which are detection rate (same as the true positive) and false positive rate. Detection rate $P_D$ is defined as the probability that the attack is correctly recognized and false positive rate $P_F$ is defined as the probability that a normal measurement vector is misclassified as malicious. We use Receiver Operating Characteristic (ROC) curve to show the relationship between $P_D$ and $P_F$ and measure tradeoffs between detection rate and false positive rate. We run simulations based on measurements (e.g., measurements in the morning of 100 days on house No. 1002) to collect enough samples and estimate the mean value $\mu$. Then, we set detection threshold $\tau$ based on the false positive rate $\alpha = 0.05$. We use the measurements of 100 days to present the normal measurements, which are not manipulated by the adversary and derive $P_F$ with the detection threshold. After that, we simulate the malicious measurements in the following way. Similar to the signal-to-noise ratio (SNR), we first define signal-to-attack ratio (SAR) that is defined as $SAR = 10 \log_{10} \frac{\sigma}{\sigma + \mu}$ to quantify the strength of attacks, where $X_i$ and $c_i$ are the maliciously manipulated measurement and true measurement, respectively. We then apply the anomaly detection discussed above to derive detection accuracy $P_D$. Note that $SAR = 11dB$ and $SAR = 8dB$ represent that the adversary could change 8% and 12% of measurement values, respectively.

Figures 7 and 8 show the ROC curve of our detection algorithm. As we can see, when $SAR = 11dB$, the detection algorithm achieves an accuracy of 60% with a false positive rate of 55%, while the adversary could only change up to 8% of the true value of measurements. When $SAR = 8dB$, the detection rate approaches almost 100% with a false positive rate of 55% when the adversary can manipulate up to 12% of the true value of measurements. Here, we can obtain 90% detection rate with a false positive rate of less than 40%. As we can see from these figures, detection rate becomes higher when the attack strength increases. This is as expected, the standard deviation of malicious measurements is higher when the attack becomes stronger.

6. CONCLUSION

In this paper, the critical issue of quantifying uncertainty on the energy usage was addressed. Particularly, the Shapiro-Wilk test and Quantile-Quantile plot normality test were adopted to investigate the statistical distribution of energy usage and the machine learning based approaches (e.g., SVM and neural network) were developed to conduct the accurate forecasting of energy usage. Extensive experiments on a real-world meter reading data set were conducted to validate the effectiveness of the developed approaches. The experimental data shows that the energy usage can be largely approximated with a Gaussian distribution and the SVM-based machine learning approaches can accurately predict the energy usage. The extensions to other areas (e.g., forecasting energy generation, determining optimal demand response, and anomaly detection of malicious energy usage) were discussed as well.

7. REFERENCES

ABOUT THE AUTHORS:

Wei Yu received the B.S. degree in electrical engineering from Nanjing University of Technology, Nanjing, China, in 1992, the M.S. degree in electrical engineering from Tongji University, Shanghai, China in 1995, and the Ph.D. degree in computer engineering from Texas A&M University in 2008. He is currently an Associate Professor with the Department of Computer and Information Sciences, Towson University. Before joining Towson, he was with Cisco Systems Inc. for nine years. His research interests include cyberspace security, computer networks, and cyber-physical systems. He received the NSF Faculty Early Career Development (CAREER) award in 2014 and the Best Paper Award at the 2013 and 2008 IEEE International Conference on Communications (ICC), respectively.

Dou An received the B.S. degree in applied mathematics from Northwestern Polytechnical University, China, in 2011. He is currently working toward the Ph.D. degree in the Department of Automation Science and Technology, Xi'an Jiaotong University. His current research interests include cyber-physical systems, power grid security.

Qingyu Yang received the B.S. and M.S. degrees both in mechatronics engineering from Xi'an Jiaotong University, China, in 1996 and 1999, respectively, the PhD degree in control science and engineering from Xi'an Jiaotong University, China, in 2003. He is a professor in School of Electronics & Information Engineering at Xi'an Jiaotong University. He is also with the State Key Laboratory for Manufacturing System Engineering, Xi'an Jiaotong University. His current research interests include cyber-physical systems, power grid security, control and diagnosis of power system, and intelligent control of industrial process.

David Griffith received the PhD in electrical engineering from the University of Delaware. He worked on satellite communications systems at Stanford Telecommunications and Raytheon, and is currently with the Communication Technology Laboratory at the National Institute of Standards and Technology (NIST). His research interests include mathematical modeling and simulation of wireless communications networks, including public safety broadband networks and communications for the smart grid. He is also working on spectrum monitoring and spectrum sharing.

Guobin Xu received the B.S. degrees in mathematics and economics from Qingdao University, in 2009. He is currently a Ph.D. candidate in the Department of Computer and Information Sciences at Towson University. His research interests include cyber-physical systems, computer networks, and cyber security.
Super Resolution Face Image Based on Locally Linear Embedding and Local Correlation

Vinh Nguyen  
Center for Machine Vision and Security Research  
Kennesaw State University  
Marietta, GA, 30060, USA  
1-404-834-3906  
baovinh007@yahoo.com

Chih-Cheng Hung  
Sino-US Intelligent Information Processing Lab  
Anyang Normal University, China  
Center for Machine Vision and Security Research  
Kennesaw State University  
Marietta, GA, 30060, USA  
1-678-915-3574  
chung1@kennesaw.edu

Xiang Ma  
School of Information Engineering  
Chang’an University  
Xi’an, 710061, P.R.China  
maxiangmail@163.com

ABSTRACT
In this paper, we propose a new face hallucination algorithm based on Locally Linear Embedding and Local Correlation method (LC-LLE). The LC-LLE algorithm is an improved locally linear embedding (LLE) algorithm by combining LLE algorithm and local correlation coefficients. The main difference between LC-LLE and LLE algorithms is that LC-LLE uses two different measures for searching the nearest neighbors for matching the most similar patches, while LLE uses only Euclidean distance for searching the nearest neighbors. Specifically, we calculate the Euclidean distance between the low-resolution input patch and patches in the low-resolution training images to select $z$-NN (i.e. $z$ number of nearest patches). Then, we use the inner product for local correlation computation between the input patch and selected $z$-NN to identify $k$ nearest neighbors (i.e. $k$-NN). After that the reconstruction weights are derived using $k$-NN patches, and generate the high-resolution image patches based on the reconstruction weights. Finally, high-resolution patches are synthesized into the high-resolution image. Experimental results show that the proposed method achieves better performance for high-resolution image reconstruction than Ma’s method with LLE and PCA methods. 1

Categories and Subject Descriptors  
I.2.8 [Computing Methodologies]: Artificial Intelligence—Problem Solving, Control Methods, and Search; I.2.8 [Theory of Computation]: Analysis of Algorithms and Problem Complexity—Miscellaneous.

General Terms  
Super resolution, face hallucination, dimensionality reduction

Keywords  
Super resolution, face hallucination, locally linear embedding, correlation efficient, position patch, PCA, and dictionary pair.

1 Copyright is held by the authors. This work is based on an earlier work: RACS’14 Proceedings of the 2014 ACM Research in Adaptive and Convergent Systems, Copyright 2014 ACM 978-1-4503-3060-2. http://dx.doi.org/10.1145/2663761.2663767

1. INTRODUCTION
There are many image enhancement techniques which have been well developed in digital image processing and computer vision. Super resolution (SR) is one of image enhancement methods which have received more attentions in the past decades since Tsai and Huang published their pioneering work on multi-frame image restoration [17]. Due to the underlying complicated motion, different noise levels, and blur kernel issues, super resolution image enhancement is still a challenging problem [18]. A super resolution problem can be formulated as estimating a high-resolution (HR) image by giving a number of low-resolution (LR) images which differ in geometric transformations, lighting transformations, camera blur and noise. In general, SR methods can be divided into two categories [19]: one is the traditional approach which is known as multi-frame images or regulation-based methods. The other is the single-frame approach which is also called the learning-based or example-based methods. In the face hallucination, several learning and example-based methods have been proposed [20-21]. Liang et al. gave a survey on the face hallucination [22]. For most of those works, constraint and similarity, either globally or locally, are important parameters used in the SR reconstruction.

Due to the high dimensionality associated with the face image, several dimensionality reduction methods are employed to reduce the intrinsic high dimensionality problems. Such an intrinsic and sparse data space and high dimensionality present a challenge for feature extraction and further processing. In order to develop useful applications, two fundamental problems have to be solved [23]: How to reduce the data volume/dimensionality without loss of critical information and given a finite and fixed size of training data, the reconstruction result may not be degraded. The problem is called the “curse of dimensionality” [23]. This means that high-dimensional data spaces are mostly empty, indicating that the data structure involved exists primarily in a subspace which is also coined as the “empty space phenomenon” [23]. Two possibilities exist in reducing the high-dimensional data to avoid the empty space phenomena [23]. The first is to focus on the separation between relevant and irrelevant variables. The second is to concentrate on the dependence between the relevant variables.

Algorithms for nonlinear dimensionality reduction (i.e. feature extraction) fall broadly into two categories, global and local embedding, each of which have advantages and disadvantages [24, 25]. Some dimensionality reduction methods such as sparse principle component analysis (S-PCA), non-linear PCA, Locally
Linear Embedding (LLE), and Isomap have been widely used in many different applications. The LLE was used in the face hallucination construction [4, 9]. LLE attempts to discover nonlinear structure in high dimensional data by exploiting the local symmetries of linear reconstructions [26]. In this study, the LLE and local correlation method are used for searching the nearest neighbors for matching the most similar patches in the example-based super resolution reconstruction. The paper is organized as below: section 2 briefly describes the related work; section 3 explains the algorithm used in this paper; section 4 shows some experimental results, and the conclusion follows.

2. RELATED WORK

Baker and Kanade introduced face hallucination which is the problem of generating a high-resolution image from one or more low-resolution images [2]. In their method, they used a set of training images to generate the high-resolution image. Liu et al. [8] proposed a two-step statistical modeling approach that integrates both a global parametric model and a local nonparametric model. They used a global linear model to learn the relationship between low and high-resolution face images. Then, they used a patch-based nonparametric Markov random field for the local model. Motivated by Liu’s method, many researchers treated face hallucination as a two-step problem. Zhuang et al. [16] introduced locality preserving hallucination (LPH) algorithm that combines the locality preserving projections (LPP) and radial basis function (RBFs) to reconstruct the high-resolution image. Huang et al. [6] suggested using Procrustes analysis [13] to estimate the linear mapping matrix from the principal component analysis (PCA) coefficients of low and high-resolution face images as corresponding inputs and outputs. Canonical correlation analysis was used to maximize the relationship between the PCA coefficients of low and high-resolution face image by Huang et al. [5]. Shi and Qi used the high-pass filter to extract the high-frequency features and then applied PCA to reconstruct HR face image [12]. Wang et al. proposed a weighted adaptive sparse regularization (WASR) method. This approach is based on a distance-inducing weighted \( l_q \) norm penalty imposed on the solution. In this algorithm, both the adjusted shrinkage parameter \( q \) and the weighted \( l_q \) penalty function have been applied to make elastic description capability available in the sparse domain, leading to more conservative sparsity in an ascending order of \( q \) [14].

Chang developed face hallucination methods based on a set of low and high-resolution patch pairs [3]. Yang et al. [15] proposed a face hallucination method based on the perspective of compressed sensing. They used sparse representation of low-resolution patches to recover the corresponding high-resolution patch. Ma et al. [9] proposed a face hallucination method based on position-patch of each training images. In this method, they used locally linear embedding (LLE) to estimate the reconstruction weight of the low-resolution image, and then applied the same reconstruction weight to infer the high-resolution image. Jung et al. [7] used constrained convex optimization based on sparse representation instead of least square estimation to obtain the optimal weights for face hallucination.

LLE has many advantages over other manifold learning methods for dimensionality reduction, but it is limited when dealing with sparse or unevenly distributed data. When the LLE is used, data points located far away are mapped to nearby points [4], so it makes the searching nearest neighbor patches not quite accurate. As the result, it affects the reconstruction of the high-resolution target image. To solve this problem, we combine the methods used by Ma et al. [9] and Chen and Liu [4] in the face hallucination. Specifically, both Euclidean distance and local correlation metrics are used for searching the nearest neighbor patches based on Chen’s idea [4] and then apply LLE algorithm to reconstruct high-resolution target image. Although Chen and Liu [4] already proposed LC-LLE concept in their paper, LC-LLE has not been applied to face hallucination and extensively tested yet. Therefore, we are using their LC-LLE concept in our experiments to achieve the more accurate reconstruction high-resolution image.

3. THE PROPOSED ALGORITHM

In this section, we briefly present LC-LLE algorithm based on the ideas proposed by Chen and Liu [4]. In a high-dimensional data space \( R^D \), let \( X = \{X_1, X_2, \ldots, X_N\} \) be a set of \( N \) points. We presume the data points lie on or close to a nonlinear manifold of intrinsic dimensionality \( d < D \) (typically \( d \ll D \)). The aim of LC-LLE is to find a low-dimensional embedding of \( X \) by mapping the

---

**Figure 1.** (a) shows a low-resolution image which is down sampled from the high-resolution and the goal is to reconstruct the HR image and (b) shows two patches (size of 3 x 3) with one column overlapped and one row overlapped.
matrix $Z$ can be computed by

$$Z = \sum_{i,j} \text{weights} \cdot \text{image patch}.$$ (1)

The image patch $Y$ is computed by

$$Y = \sum_{i,j} \text{weights} \cdot \text{image patch}.$$ (2)

D-dimensional data. By denoting the corresponding set of $N$ points in the embedding space $\mathbb{R}^d$ by $Y = \{Y_1, Y_2, \ldots, Y_N\}$, the LC-LLE comprises four steps [4]:

Step 1: For each data point $X_i$, select $z$ nearest neighbors ($z$-NN) based on the Euclidean distance.
Step 2: For $z$ nearest neighbors, compute the inner product between each neighbor and $X_i$, sort them in descending order, and select the $k$ points as true neighbors ($k$-NN).
Step 3: Measure reconstruction error resulting from approximation of each $X_i$ by its true $k$ neighbors and compute reconstruction weights by minimizing the error.
Step 4: With the reconstruction weights, compute the low-dimensional embedding best preserving the local geometry represented.

By applying the above LC-LLE algorithm to face hallucination, we establish the new face hallucination with LC-LLE method. Our method is described in the following section. The formulations described here are taken from Ma et al. [9] with a few different notations.

Let $X_i^L$ and $X_i^R$ represent the low and high-resolution training face images, $n = 1, 2, \ldots, N$, where $N$ is the number of training images. Each low and high-resolution training face image are shown in Figure 1 and represented as a set of small images patches with overlap $\{X_i^L(i,j)\}_{p=1}^M$ and $\{X_i^R(i,j)\}_{p=1}^M$ respectively. Also, let $Y$ represent the testing face, and the testing face is represented in overlapped patches as $\{Y_i^L(i,j)\}_{p=1}^M$. For each testing image patch $\{Y_i^L(i,j)\}_{p=1}^M$, we select $z$ neighbors from $\{X_i^L(i,j)\}_{p=1}^M$, $\{X_i^R(i,j)\}_{p=1}^M$ by using Euclidean distance. Then inner product between each testing patch and its $z$ neighbors is applied to identify $k$ neighbors. For $k$ neighbor images, the image patches located at position $(i,j)$ such as $X_k(i,j)$ are defined as position-patches of the image patch $Y_k(i,j)$, and its reconstruction weight matrix is $w_m(i,j)$, where $w_m(i,j)$ represent the contribution of each training image patch located at the same position. All weights at the position $(i,j)$ are constrained to have a sum of one.

Each patch $Y_k(i,j)$ in the face image $\{Y_i^L(i,j)\}_{p=1}^M$ is represented by

$$Y_k(i,j) = \sum_{n=1}^{K} w_n(i,j) X^n(i,j) + e$$ (3)

where $e$ is the reconstruction error.

From (1), the minimization of the reconstruction error $e$ is used to determine the optimal reconstruction weights:

$$w(i,j) = \arg \min_{w_m(i,j)} ||Y_k(i,j) - \sum_{n=1}^{K} w_n(i,j) X^n(i,j)||^2$$ (4)

where $w(i,j)$ is a K-dimensional weight vector of each reconstruction weight $w_n(i,j)$, for $n = 1, \ldots, K$. Let

$$S = Y_k(i,j) \cdot C^T - B$$ (5)

where $B$ is a matrix with its columns being the training patches $X^n(i,j)$, and $C$ is a column vector of ones. The local covariance matrix $Z$ can be computed by $Z = S^T \cdot S$.

Eq. (2) is a constrained least squares problem with the following solution:

$$w(i,j) = (Z^{-1}C)(C^T Z^{-1}C)^{-1}$$ (6)

A more efficient way to obtain $w(i,j)$ is to solve the linear system

$$Z \cdot w(i,j) = C$$

and then rescale the weights so that the sum is one.

$w(i,j)$ are used to reconstruct the new image patch $Y_k^R(i,j)$:

$$Y_k^R(i,j) = \sum_{n=1}^{K} w_n(i,j) X^n(i,j) \equiv Y_k^L(i,j)$$ (7)

where $Y_k^R(i,j)$ is a vector and be converted into a matrix to integrate the global image. By integrating the reconstructed patches according to the original position, the global image can be synthesized. The pixels value in the overlapping regions between two adjacent patches reconstructed is averaged to obtain the pixels of the overlapping regions in the reconstructed global image.

Given the low-resolution image $Y_L$ and the original high-resolution image $Y_H$ that is $p^2$ times larger than $Y_L$, the degradation process can be expressed as [2, 8]

$$Y_L = \frac{1}{q^2} \sum_{k=0}^{q-1} \sum_{k=0}^{q-1} Y_H(qi + k, qj + l) + ns(i,j)$$ (8)

where $q$ is a positive integer and $ns(i,j)$ random noise. If $Y_H$, $Y_L$, and $n$ are represented $L \times 1$, $K \times 1$ and $K \times 1$ vectors respectively, Eq. (6) can be simplified as

$$Y_L = HY_H + ns$$ (9)

where $H$ is a $K \times L$ matrix. Eq. (7) combines a smoothing and a down-sample step and can be rewritten in image patch as

$$\{Y_L^p(i,j)\}_{p=1}^N = H \{Y_H^p(i,j)\}_{p=1}^N + ns$$ (10)

For each patch $Y_L^p(i,j)$ in the input low-resolution image, the weight $w_n(i,j)$ is obtained from (1) to (4), satisfying the equation:

$$Y_L^p(i,j) \equiv \sum_{n=1}^{K} X_L^p(i,j) w_n(i,j)$$ (11)

Replacing each low-resolution image patch $X_L^p(i,j)$ by its corresponding high-resolution sample $X_H^p(i,j)$, the result is as follows:

$$\sum_{n=1}^{K} X_H^p(i,j) w_n(i,j) = Y_H^p(i,j)$$ (12)

From (7) and (10), without considering noise, we have

$$H \cdot Y_H^p(i,j) = \sum_{n=1}^{K} H \cdot X_H^p(i,j) w_n(i,j) \equiv \sum_{n=1}^{K} Y_L^p(i,j) w_n(i,j)$$ (13)

From (9) and (11), we have

$$Y_L^p(i,j) \equiv H \cdot Y_H^p(i,j)$$ (14)
From (12), we can see that the degradation of \( Y_H(i,j) \) is close to the low-resolution input \( Y_L(i,j) \).

All the high-resolution patches \( Y_H(i,j) \) are copied and synthesized to form the final global high-resolution image \( \{ Y_H(i,j) \}_{i,j} \). In the final result, pixels of the overlapping regions are obtained by averaging the pixels value in the overlapping regions between two adjacent target patches [9]. The proposed face hallucination method is summarized in the following algorithm, and the Figure 2 shows the flowchart of the proposed algorithm.

**Algorithm:** Face hallucination based on locally linear embedding and local correlation

**Input:**

An LR testing face image \( Y_L \), LR training images \( \{ X_L^n \}_{n=1}^N \), and HR training images \( \{ X_H^n \}_{n=1}^N \), \( z \) and \( k \) values with \( z > k \).

1. Divide the LR testing image \( Y_L \) into \( 3 \times 3 \) image patches with one column of the patches overlapped for sliding the window to the right and one row of the patches overlapped for sliding the window down.

For LR testing image patch \( Y_L(i,j) \) located at position \( (i, j) \) do

2. Compute the Euclidian distance between \( Y_L(i,j) \) and LR training patches, then select \( z \) nearest neighbors (\( z \)-NN).

3. For \( z \) nearest neighbors, compute the inner product between each patch in \( z \) and \( Y_L(i,j) \), then select true \( k \) nearest neighbors (\( k \)-NN).

4. Calculate reconstruction weights, \( w \), and synthesize the HR patch (HRP) at position \( (i, j) \) based on corresponding HR patches in the \( k \) nearest neighbors.

End for

**Output:** HR face image

---

**4. EXPERIMENTAL RESULTS**

In this experiment, we randomly selected 160 low and high-resolution image pairs, in which 10 pairs to be used for testing, and 150 pairs for training from FERET database [10]. We manually aligned the original high-resolution face images by the position of two eyes and normalized to the size of 96 x 128. Then the, high-resolution images were down-sampled to low-resolution images of size 24 x 32. We also set low-resolution patches to the size of \( 3 \times 3 \) with one column of the patches overlapped when we slide the window to the right and one row of the patches overlapped when we slide the window down.

Figures 3, 5 and 7 show some high resolution results of our proposed method, the method proposed in [9] with LLE, and the method in [9] with PCA with the same training set. If the zooming factor is 4, the results of the method in [9] with PCA and LLE are more blurred than those of our method. Moreover, we also compare the peak signal-to-noise ratio (PSNR) and the structure similarity index metric (SSIM) [27] values of the images tested as shown in Figures 4, 6 and 8. The higher value is better for PSNR and SSIM values. Therefore, experimental results in Figures 4, 6 and 8 show that our proposed method has better performance than those of the method in [9] with LLE, and with PCA. In the experiment shown in Figure 7, we used FEI Face Database [28] with 300 HR-LR images for training, and 100 HR-LR images for testing. The results are shown in Figures 7 and 8. Figure 8 shows that our method is better than the two other methods.

**5. CONCLUSIONS**

In this paper, we proposed a new face hallucination based on locally linear embedding with local correlation discussed in Chen and Liu’s paper. By combining two different measures for searching the nearest neighbor patches, we can achieve more accurate nearest neighbor patches for reconstructing the HR target...
image. The reconstruction weights for high-resolution patches can be more precisely computed based on true nearest neighbor patches. The reconstruction weights are used to generate high-resolution patches which are synthesized into high resolution target image. Experimental results show that our proposed method generates better results than those of LLE and PCA algorithms. Although this is a simple experiment for us to begin with research in face hallucination, it establishes some basis for us to explore more advanced algorithms and systems in this area.

Many global and local dimensionality reduction methods have been proposed in the literature; however, their integration is not fully explored yet in many potential applications. We will investigate those innovative methods for the feasibility to replace LLE algorithm in our future work. In addition, we will look at different models by embedding the local correlation with other dimensionality reduction methods for improving the high-resolution target image.

6. ACKNOWLEDGMENT

This work was partially supported by the National Natural Science Foundation of China under Grant 61101215, the China Postdoctoral Science Foundation under Grant 2013M542308 and Grant 2014T70895.

![Figure 3](image)

Figure 3. a) The input low-resolution image (32x24), b) results of the method proposed in [9] with PCA, c) results of the method in [9] with LLE, d) results of our method, and e) the original high-resolution (128x96). The zooming factor is 4.

![Figure 4](image)

Figure 4. The column chart for PSNR and SSIM. (A) shows results of the method in [9] with PCA, (B) results of the method in [9] with LLE, and (C) results of our method.
Figure 5. a) The input low-resolution image (32x24), b) results of the method in [9] with PCA, c) results of the method in [9] with LLE, d) results of our method, and e) the original high-resolution (128x96). The zooming factor is 4.

Figure 6. The column chart for PSNR and SSIM measures. “A” denotes results of the method in [9] with PCA, “B” results of the method in [9] with LLE, and “C” results of our method.
Figure 7. a) The input low-resolution image (32x24), b) results of the method in [9] with PCA, c) results of the method in [9] with LLE, d) results of our method, and e) the original high-resolution (128x96). The zooming factor is 4.

(a) PSNR Measure.  
(b) SSIM Measure.

Figure 8. The column chart for PSNR and SSIM measures. “A” denotes results of the method in [9] with PCA, “B” results of the method in [9] with LLE, and “C” results of our method.
7. REFERENCES
**ABOUT THE AUTHORS:**

<table>
<thead>
<tr>
<th>Author</th>
<th>Affiliation</th>
<th>Education</th>
<th>Research Interests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vinh Nguyen</td>
<td>received the B.S. degree in Viet Nam National University, Ho Chi Minh, Viet Nam, in 2009. He is currently pursuing the MS. degree from Southern Polytechnic State University (now it is known as Kennesaw State University), Marietta, USA. His research interests include image processing in machine.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chih-Cheng Hung</td>
<td>received his B.S. in applied mathematics from Soochow University in Taiwan, and his M.S. and Ph.D. in Computer Science from the University of Alabama in Huntsville, AL, USA. He is a professor of Computer Science at Kennesaw State University (KSU), Marietta, GA, USA. He developed image processing software tools with the Department of Multispectral Image Processing Applications at Intergraph Corporation from 1990 to 1993 and he was an associate professor at Alabama A&amp;M University from 1993 to 1999. He is also with the Sino-US intelligent information processing lab in Anyang Normal University, China, and the Center for Computer Vision and Security in KSU. His research interests include image processing, pattern recognition and artificial intelligence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xiang Ma</td>
<td>received the B.S. degree in North China Electric Power University, Beijing, China, in 1999, the Ph. D. degree from Xi’an Jiaotong University, Xi’an, China in 2011. He is currently an associate professor with School of Information Engineering, Chang’an University, Xi’an, China. His research interests include face image processing and image processing in machine and intelligent transportation.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Post Ranking in a Blogosphere: Algorithms and Evaluation

Won-Seok Hwang  
Dept. of Electronics and Computer Engineering  
Hanyang University, Korea  
+82-2-2220-4567  
hws23@hanyang.ac.kr

Sang-Wook Kim  
Dept. of Electronics and Computer Engineering  
Hanyang University, Korea  
+82-2-2220-4567  
wook@hanyang.ac.kr

ABSTRACT
Existing algorithms of post ranking in a blogosphere do not consider the information on categories and timestamps of posts, which is valuable to improve the accuracy of post rankings. The category, created by a blogger, implies her/his interesting topics, and the timestamps, which indicates the creation times of a post and a blog action, are closely related to the quality of a post. In this paper, we propose three strategies that exploit the information related to the categories and timestamps, and address how to apply it to four existing post ranking algorithms, i.e., PostRank, BAITS, BloggerAVG, and PBFS. We evaluate the accuracy of these algorithms by conducting extensive experiments using real-world blog data. The results show that our strategies help each algorithm improve the accuracy.

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]: Information Filtering

General Terms
Algorithms, Performance, Experimentation.

Keywords
Blog, Blogosphere, Post ranking algorithm, Blog action, Category, Time interval.

1. INTRODUCTION
A blog is a personal web page composed of multiple posts and maintained by a blogger, the owner of the blog [1][2][3][4]. Blog service companies provide functions of writing a post, a comment, a scrap, and a trackback, to manage the posts. A trackback is an action of writing a new post related to someone else’s post and putting a link to the original post within the new post. Unique to blog sites in Korea, a scrap is an action of copying someone else’s post to one’s own blog. A comment is an action of writing a short opinion on someone else’s post. By using these functions, bloggers can express or share their ideas with other bloggers. We call these functions blog actions. The blog actions generate various relationships between bloggers and posts. Bloggers, posts, and blog actions can be thus viewed as a social network, which is called a blogosphere.

As the number of bloggers is increasing over time, a great number of posts on many different topics have appeared. This brings inconvenience to bloggers in finding high quality posts because a search engine finds a lot of posts that are matched to a given query. In order to overcome this problem, a ranking algorithm is needed to determine the order of posts according to their quality and relevance to a query so that users can find high quality posts only by checking the section of a query result.

Some researches have been done on algorithms specialized on post ranking [5][6]. This researches model a blogosphere as a graph that regards bloggers and posts as nodes in two different parts and does blog actions as edges. Then, it computes a score of each post by passing the scores of bloggers and posts through the edges on that graph. As a result, the bloggers who have performed the blog actions on high quality posts receive high scores and the posts on which those bloggers perform the blog actions end up with high scores.

In this paper, we propose new strategies of using categories and timestamps of blog actions in order to improve the accuracy of existing post ranking algorithms. Previous algorithms give only one score to a blogger depending on her/his ability of finding high quality posts. Since the ability of each blogger, however, may differ depending on topics, their scores need to be subdivided for different topics. In our proposed strategies, a blogger’s score is calculated separately according to the categories that s/he has made.

Bloggers make categories to organize their blogs by grouping posts they have written, scrapped, or trackbacked according to topics. The posts in different categories can be considered belonging to different topics. Therefore, a score of a blogger on a specific topic can be proportional to the quality of posts in a category (representing that topic). Furthermore, the timestamp of creating a post, and performing a scrap or trackback (i.e., blog action) can also be used to help improve the accuracy of post ranking algorithms. If the quality of a post is high, bloggers are more likely to do a scrap or a trackback on the post right after the creation of the post. In other words, as the time of a post receiving a scrap or a trackback is shorter, the quality of the post could be considered higher. Thus, when the scores of posts are computed in our proposed strategies, a post receives a high score if it has a short time interval between its creation and blog action. A weight is given to a post depending on this time interval to make post scores calculated more accurately. Also, the quality of a post can be evaluated by the order of blog actions: how much good bloggers perform a blog action earlier than other bloggers on the post.

1 Copyright is held by the authors. This work is based on an earlier work: RACS'14 Proceedings of the 2014 ACM Research in Adaptive and Convergent Systems, Copyright 2014 ACM 978-1-4503-3060-2. http://dx.doi.org/10.1145/2663761.2664204
2. RELATED WORK

Although post ranking algorithms are becoming more important, only a few researches have been studied. On the other hand, various blog ranking algorithms have been proposed. Apostolos et al. [2] modeled blogs as nodes and hyperlinks or similar-relationships between blogs as edges to generate a graph. They proposed BlogRank, which determines a ranking of each blog by analyzing the generated graph. B2Rank [3] decides the ranking of a blog by analyzing the characteristics of its blogger. The characteristics of a blogger are related to the actions of writing a comment, creating a post, and putting a hyperlink, and their timestamps of generating a hyperlink. Kritikopoulos et al. [4] proposed an algorithm of using a graph composed of explicit links and implicit links. In the graph, explicit links represent blog-blog relationships created by bloggers, and implicit links do relationships inferred by analyzing the blogs. The blog ranking algorithm can determine the ranking of a post by regarding the quality of a post as proportional to the overall quality of posts in the blog. However, because all the posts of a blog are not always of the same quality, the blog ranking algorithm cannot determine the ranking of a post precisely.

EigenRumor is an algorithm that assigns rankings to posts [5]. It gives an authority score and a hub score to a blogger and a reputation score to a post. An authority score indicates the quality of a post that the blogger created, and a hub score indicates the quality of a post that the blogger wrote comments to. A reputation score of a post is dependent on the authority score of the blogger who wrote the comment on it. There are more post ranking algorithms [6] such as Pindegree, PostRank, BAITS, BloggerAVG, and PSALSA. These algorithms are the modified versions of existing web-page ranking algorithms, i.e., Indegree [7], PageRank [8], HITS [9], HubAVG [7], and SALSA [10], to be suitable for blogosphere environment. Among these algorithms, we introduce PostRank, BAITS, and BloggerAVG in this paper.

PostRank is a modified version of PageRank. It computes the scores of posts and bloggers separately. The score of a post is dependent on the scores of bloggers who performed scraps or trackbacks on that post, and the score of a blogger is dependent on the scores of posts written by that blogger. BAITS is an algorithm based on HITS. It maintains the scores of bloggers and posts. Here, the score of a post is calculated in the same way as PostRank, but the score of a blogger is dependent on the scores of those posts that the blogger perform scraps or trackbacks on. BloggerAVG is an extension of BAITS. Its unique idea is that it prevents bloggers from receiving excessively high scores when they simply perform scraps and trackbacks on a large number of posts.

Existing post ranking algorithms compute the scores of bloggers in order to determine the rankings of posts. These algorithms, however, overlooked the fact that bloggers often have different topics of interest; hence, the knowledge depth of bloggers could differ according to topics. As a result, their scores may be calculated inaccurately, leading to inaccurate computations of the posts scores as well. Moreover, the importance of blog actions could differ according to their time intervals or orders. If three features regarding categories, time intervals, and orders are considered in the ranking algorithms, we expect the scores of posts can be computed more accurately.

3. PROPOSED STRATEGIES

In this paper, we propose three strategies to produce a more accurate ranking result by modifying the way of assigning scores to bloggers and posts in existing post ranking algorithms. The proposed strategies exploit three more information, i.e., the categories of posts, time interval of blog actions, and order of blog actions to enhance existing post ranking algorithms (i.e., PostRank, BAITS, and BloggerAVG). In addition, we propose a novel post ranking algorithm, PBFS and apply our strategies to it.

3.1 Using Category

In post ranking algorithms, the blogger score reflects the ability of performing a blog action on high quality posts. However, the ability of evaluating posts differs depending on their topics; hence the blogger score should be assigned according to the topics. In order to identify the topics, we exploit the categories to which the blogger assigns the posts. The posts in one category are likely to belong to the same topic. In the proposed strategy, the score of a blogger on one category is proportional to the scores of posts in the category on which she performed scraps or trackbacks.

We define a post-category graph in order to separate the scores of a blogger according to topics and use them in the process of assigning scores to posts. In a post-category graph, two different types of nodes represent a post and a category, respectively, and directed edges represent the existence of scraps or trackbacks.

Figure 1 shows an example of a post-category graph. A round rectangle with a dotted line represents a blogger. The bloggers are not an actual part of the graph but are drawn in order to show which categories belong to specific bloggers. A round rectangle in a circle with a dotted line represents a category. A rectangle represents a post and an arc between a rectangle and a round rectangle indicates a blog action. The arc between category #3 and post #2 shows that blogger #2 performed a scrap or a trackback on post #2, and as a result, the post is placed under category #3 of blogger #2.

The post-category graph can be represented as an adjacency matrix \( C \) that is used on computing the scores of the bloggers according to each category.

\[
C = \begin{cases} 
1 & \text{Post } p \text{ is performed scrap (trackback) in category } c \\
0 & \text{otherwise}
\end{cases}
\]

3.2 Using Time Intervals

In post ranking algorithms, blog actions such as a scrap, a trackback, and a comment are interpreted as a means of recommending a high quality post. The quality of recommended posts would be dependent on the time interval between the creation of a target post and a blog action on the post. If the quality of a post is high, a blogger would perform a scrap or a trackback on that post quickly. Thus, as the time interval of a post is shorter, the quality of the post gets higher. This paper uses the
time intervals of posts to compute the scores of posts more precisely.

In the proposed strategy, we assign *interval weights* to edges on a post-category graph by considering the time intervals to compute the scores of posts. In order to determine interval weights, we analyze the frequency of blog actions, i.e., scraps and trackbacks, as the time goes by. Here, because the period of time for the analysis is quite long, we set the time unit as 5 days.

For the analysis, we collected anonymous data from one of the largest blog world in South Korea, Naver (blog.naver.com), for six months. To identify the patterns from the analysis, we divided all the posts into 5 sets according to the creation time of posts. Then, we counted the number of blog actions on each set. The result is shown in Figure 2. The horizontal axis represents the time flow and the vertical axis represents the number of blog actions. The result shows similar patterns occur in all the sets.

The proposed strategy assigns the interval weights of each edge $w_t(x)$ to be proportional to the frequency of blog actions on each time unit, and then uses the interval weights to compute the post scores. Here, $x$ is the time interval between the creation of a post and the occurrence of the blog action on it. The distribution of interval weight $w_t(x)$ is represented as shown in Figure 3.

A post-category graph with weights can be represented as an adjacency matrix $T$ as follows, and by using this matrix, we can calculate the post scores by considering the time interval.

$$T = \begin{cases} f_{pc} = w_t(x) & \text{Scrap (trackback) is performed on post p in category c} \\ t_{pc} = 0 & \text{otherwise} \end{cases}$$

3.3 Using Order of Blog Actions

We also claim that a scrap or a trackback needs to be regarded more important than those performed lately even though their time intervals are similar. To apply this claim, we propose another notion, called *ordinal weight*, for edges. We determine an ordinal weight of blog actions by considering both the frequency and order of blog actions. The order of a blog action is $y$ in a post when it is the $y$-th one performed on the post. To get the statistics of blog actions relevant to the order, we also refer to the Naver data that we have explained earlier. In the result, we observed that there are a more number of blog actions that have a smaller value of order.

$$O = \begin{cases} o_{pc} = w_s(y) & \text{Scrap (trackback) is performed on post p in category c} \\ o_{pc} = 0 & \text{otherwise} \end{cases}$$

3.4 Modified Post Ranking Algorithms

This paper applies adjacency matrices $C$, $T$, and $O$ to three existing post ranking algorithms, PostRank, BAITS, and BloggerAVG [6], and one proposed algorithm called PBFS to
employ our three strategies. In PostRank, a post score is dependent on the scores of bloggers who performed scraps or trackbacks on the post. In this process, if the blogger scores are computed by considering categories, we can evaluate the bloggers more precisely. The accuracy of evaluating a blogger is highly related to that of evaluating the post. Also, since the time interval may represent the quality of the post, it would be useful to improve the accuracy of PostRank. In this paper, we modify the way of post score calculation in PostRank as follows.

$$\hat{p} = TC\hat{c}$$  

(1)

where a column vector $\hat{b} = [p_1, p_2, \ldots, p_n]^T$ contains the scores of all posts and a column vector $\hat{c} = [c_1, c_2, \ldots, c_l]^T$ contains the scores of all categories. Each score $c_i$ indicates the score of category $i$ of a blogger. Also, $n$ and $l$ indicate the numbers of posts and bloggers in a blogosphere, respectively. According to Equation (1), the post that receives a scrap or a trackback by a blogger with a high category score will receive a high score as well. Also, as the time interval gets shorter, the higher score will be assigned to the post.

In PostRank, we also modify the way of post score calculation with ordinal weights as follows.

$$\hat{p} = OC\hat{c}$$  

(2)

According to Equation (2), the post that receives a blog action by a blogger with a high category score will receive a high score. Also, as the order is lower, the higher score is assigned to the post. In original PostRank, the blogger score is the summation of the scores of all the posts created by her/him. Our modified PostRank assigns a score to each category for a blogger, so a blogger has several category scores according to her/his topics of interest. Since the time interval is not related to a blogger’s ability, we ignore it in computing the category score. The category scores of bloggers are computed as follows.

$$\hat{c} = WT\hat{p}$$  

(3)

In computing Equation (3), the category containing posts with high scores will eventually receive a high score. Using Equations (1), (2), and (3), we produce two versions of modified PostRank. If Equations (1) and (3) are computed repeatedly as in HITS [9], the post scores will converge and the converged ones will be the final scores. This computation produces the result considering the categories and the time intervals. We refer to this version as PostRank_CO. In addition, it is possible to compute Equations (2) and (3) repeatedly. This result involves both the categories and the order of blog actions. We refer to this version as PostRank_CT.

In order to apply our strategies to BAITS, the post score calculation is modified as shown in Equations (1) and (2) since its calculation is identical to that of PostRank. The calculation of category scores is modified as follows.

$$\hat{c} = CT\hat{p}$$  

(4)

In Equation (4), a post receives high scores when the category score is high. Using Equations (1), (2), and (4), we produce two versions of modified BAITS. One version, called BAITS_CT, repeats Equations (1) and (4) until the scores converge, then the converged ones are the final scores of posts. The other version, BAITS_CO, repeats Equations (2) and (4) to get the final result. BloggerAVG is a refined algorithm for the part of computing the blogger scores in BAITS. In this paper, we compute the blogger scores according to different categories as follows.

$$\hat{c} = \text{diag}(\frac{1}{\sum_{i=1}^{l} c_{i,j}^2}, \frac{1}{\sum_{i=1}^{l} c_{i,j}^2}, \ldots, \frac{1}{\sum_{i=1}^{l} c_{i,j}^2})CT\hat{p}$$  

(5)

Each element of a diagonal matrix, $\sum_{i=1}^{l} c_{i,j}^2$, is a summation of all the values of the $i$th row in matrix $C$ and represents the degree of the $i$th category node. If we compute the scores of each category of a blogger this way, the category score will be high if the scores of the posts in the category are high relatively.

Our strategies generate two versions of modified BloggerAVG using Equation (1), (2), and (5). One version, BloggerAVG_CT computes the scores of posts by repeating Equations (1) and (5). The other version, BloggerAVG_CO repeats Equation (2) and (5) to get the final results.

Finally, we propose a simple post ranking algorithm, called PBFS, which is a refined version of BAITS. It computes post scores by referring to the scores of posts and bloggers whose hop is less than 2 in the post-blogger graph. In addition, it considers the distance of nodes in the graph. The intuition behind this is that the scores of bloggers and posts are more related as they are located closer. A post score is computed as below:

$$\hat{p} = SB + \frac{1}{2} SST\hat{p}$$  

(6)

where a column vector $\hat{b} = [b_1, b_2, \ldots, b_m]^T$ contains the scores of all bloggers. In Equation (6), an adjacency matrix $S$ represents a post-blogger graph whose nodes represent bloggers and posts, and edges are blog actions, i.e., scraps and trackbacks.

$$S = \begin{cases} S_{bp} = 1 & \text{Blogger b performs scrap (trackback) on post p} \\ S_{bp} = 0 & \text{otherwise} \end{cases}$$

Unlike other post ranking algorithms, the post scores are computed in our algorithm by performing Equation (6) only once. We also modify PBFS by employing our strategies in two ways. Using the information on categories and interval weights, post scores are computed as below:

$$\hat{p} = TC\hat{c} + \frac{1}{2} TCT\hat{p}$$  

(7)

Similarly, we modify the computation of scores by using the information on categories and ordinal weights as below:

$$\hat{p} = OC\hat{c} + \frac{1}{2} OCT\hat{p}$$  

(8)

Equations (7) and (8) are two versions of modified PBFS: PBFS_CT and PBFS_CO, respectively. Unlike another post ranking algorithms, they are computed by performing Equations (7) and (8) only once.

4. EVALUATION

In this section, we evaluate the accuracy of post ranking algorithms and show the effectiveness of our proposed strategies.

4.1 Experimental Settings

We collected about 3 million bloggers and 1 hundred million posts for 6 months from the Naver data for our experiments. In the experiments, we only assessed if the post ranking algorithms determine correct rankings to posts but ignore the relevance of posts to queries. In order to do so, we collected the posts that are closely related to 20 queries and organized them in 20 groups. The queries were selected from the ones that A. Borodin [7] and J. Kleinberg [9] used.

<table>
<thead>
<tr>
<th>Table 1. Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alcohol Addiction, Amusement Park, Louis Armstrong, Nissan Cars, Street Basketball, Classic Guitar, Anti-Death Penalty, Gulf War, Gene Manipulation, Michael Jordan, Moon Landing, Eternal Sunshine, National Park, Internet Censorship, Cooking, Search Engine Google, Shakespeare, Stamp Collecting, Parallel Structure, Thai Tour</td>
</tr>
</tbody>
</table>

In order to evaluate the accuracy of post ranking algorithms, we had to know the ground truth (i.e., posts of high quality). In the experiments, 11 users rated the posts as “excellent”, “good”, or “bad”. We asked the users to evaluate the posts that must appear at
the top of a query result as ‘excellent’; the posts that should appear at the top or near the top of a query result are rated as ‘good’; the posts that should appear as distant from the top as possible are rated as ‘bad’. Therefore, the posts that are rated ‘excellent’ or ‘good’ are the posts that a good post ranking algorithm needs to recommend. This type of user study is often used in researches on web document ranking [9][11][12][13]. We selected precision [14] and mean average precision (MAP) [15] as an accuracy measure. Precision and MAP for 20 queries are computed separately and the averages of 20 results are considered as the result of a post ranking algorithm. We computed precision and MAP with two different true answer sets: the answer set E is a post set evaluated as ‘excellent’ by users and the answer set E&G is a post set evaluated as ‘excellent’ or ‘good’. In this experiment, we compared the top 5 and top 10 results produced by each post ranking algorithm with the ratings of volunteers to compute precisions and MAP. We measured the accuracies of PostRank, BAITS, BloggerAVG, and PBFS, and their modified algorithms combined with our strategies. The ranking algorithms in the experiment were repeated until the post scores converge; i.e., the difference between iterations becomes less than 10^-8.

4.2 Results

Table 2 shows the results of original PostRank and two modified PostRanks, PostRank_CT and PostRank_CO, equipped with the proposed strategies. The numbers in boldface indicate the results of modified ones that are more accurate than the original ones. Answer sets E and E&G were used to evaluate precision and MAP. The result using E&G shows higher precision and MAP than that of using E. This is because there are a more number of posts in E&G. PostRank_CT shows higher precision and MAP than original PostRank both with E and E&G. In p@5 and p@10, PostRank_CO performs better than original PostRank with E while it is less accurate than original PostRank. However, PostRank_CO has MAP higher than that of original PostRank. This is because PostRank_CO fails to give top-10 rankings to the posts chosen in E&G. Thus, our strategies improve the accuracy of PostRank in most cases. Among our strategies, combining categories and time intervals of blog actions is shown appropriate to PostRank more than combining categories of posts and the order of blog actions.

Table 2. Accuracy of PostRank and Modified PostRank

<table>
<thead>
<tr>
<th>Answer set</th>
<th>Algorithm</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>p@5</td>
<td>p@10</td>
</tr>
<tr>
<td>E</td>
<td>PostRank</td>
<td>0.368</td>
</tr>
<tr>
<td></td>
<td>PostRank_CT</td>
<td>0.484</td>
</tr>
<tr>
<td></td>
<td>PostRank_CO</td>
<td>0.442</td>
</tr>
<tr>
<td>E&amp;G</td>
<td>PostRank</td>
<td>0.800</td>
</tr>
<tr>
<td></td>
<td>PostRank_CT</td>
<td>0.821</td>
</tr>
<tr>
<td></td>
<td>PostRank_CO</td>
<td>0.789</td>
</tr>
</tbody>
</table>

Table 3 compares the results of BAITS and two versions of modified BAITS, BAITS_CT and BAITS_CO. In the case of using E, BAITS_CT performs better than original BAITS in terms of both precision and MAP. However, in the case of using E&G, BAITS_CT is better only in MAP. This is because only a few posts rated ‘good’ are included at the top-N list obtained by BAITS_CT. Meanwhile, in p@5, BAITS_CO shows accuracy even worse than original BAITS. On the other hand, in p@10 and MAP, BAITS_CO has accuracy better than the original one. If most users search for more than 10 posts, BAITS_CO will be expected to perform better than BAITS_CT. Also, in most cases, we observe our strategies improve the accuracy of BAITS.

Table 3. Accuracy of BAITS and Modified BAITS

<table>
<thead>
<tr>
<th>Answer set</th>
<th>Algorithm</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>p@5</td>
<td>p@10</td>
</tr>
<tr>
<td>E</td>
<td>BAITS</td>
<td>0.400</td>
</tr>
<tr>
<td></td>
<td>BAITS_CT</td>
<td>0.421</td>
</tr>
<tr>
<td></td>
<td>BAITS_CO</td>
<td>0.379</td>
</tr>
<tr>
<td>E&amp;G</td>
<td>BAITS</td>
<td>0.800</td>
</tr>
<tr>
<td></td>
<td>BAITS_CT</td>
<td>0.758</td>
</tr>
<tr>
<td></td>
<td>BAITS_CO</td>
<td>0.779</td>
</tr>
</tbody>
</table>

Table 4 compares the accuracy of original BloggerAVG, BloggerAVG_CT, and BloggerAVG_CO. In all cases, BloggerAVG_CT shows best accuracy among the three algorithms. BloggerAVG_CO has precision less than original BloggerAVG in most cases while it has MAP higher than original one. Thus, the combination of categories and time intervals is more appropriate for BloggerAVG while the combination of categories and orders is not appropriate for this algorithm.

Table 4. Accuracy of BloggerAVG and Modified BloggerAVG

<table>
<thead>
<tr>
<th>Answer set</th>
<th>Algorithm</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>p@5</td>
<td>p@10</td>
</tr>
<tr>
<td>E</td>
<td>BloggerAVG</td>
<td>0.474</td>
</tr>
<tr>
<td></td>
<td>BloggerAVG_CT</td>
<td>0.484</td>
</tr>
<tr>
<td></td>
<td>BloggerAVG_CO</td>
<td>0.505</td>
</tr>
<tr>
<td>E&amp;G</td>
<td>BloggerAVG</td>
<td>0.853</td>
</tr>
<tr>
<td></td>
<td>BloggerAVG_CT</td>
<td>0.863</td>
</tr>
<tr>
<td></td>
<td>BloggerAVG_CO</td>
<td>0.842</td>
</tr>
</tbody>
</table>

Table 5 shows original PBFS and two modified version of PBFS, PBFS_CT and PBFS_CO, with our strategies. PBFS_CT outperforms the original one in most cases while original PBFS produces a result better than the modified one in two cases: p@10 with E and MAP with E&G. On the other hand, PBFS_CO has accuracy better than original PBFS in all cases. Thus, for PBFS, we say that the combination of categories and orders is more effective to improve the accuracy.

Table 5. Accuracy of PBFS and Modified PBFS

<table>
<thead>
<tr>
<th>Answer set</th>
<th>Algorithm</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>p@5</td>
<td>p@10</td>
</tr>
<tr>
<td>E</td>
<td>PBFS</td>
<td>0.379</td>
</tr>
<tr>
<td></td>
<td>PBFS_CT</td>
<td>0.347</td>
</tr>
<tr>
<td></td>
<td>PBFS_CO</td>
<td>0.379</td>
</tr>
<tr>
<td>E&amp;G</td>
<td>PBFS</td>
<td>0.695</td>
</tr>
<tr>
<td></td>
<td>PBFS_CT</td>
<td>0.737</td>
</tr>
<tr>
<td></td>
<td>PBFS_CO</td>
<td>0.758</td>
</tr>
</tbody>
</table>

In this section, we have compared the precision and MAP of the existing post ranking algorithms and the modified algorithms by our strategies. We observed our strategies help improve the existing post ranking algorithms in most cases.
5. CONCLUSIONS
A few post ranking algorithms were proposed as a way of determining rankings to posts in a blogosphere. However, the post ranking algorithms did not use the useful information regarding the categories of posts nor the time intervals of blog actions, which left a room for more improvement of accuracy. In this paper, we proposed strategies to use this information to improve the accuracy of post ranking algorithms. The proposed strategies compute the score of a blogger by considering the categories of the blogger. Also, in computing the score of a post, they assign a more weight to the blog action if its time interval is larger or its order is earlier. These strategies are then applied to post ranking algorithms, PostRank, BAITS, BloggerAVG, and PBFS. Finally, we have shown, via extensive experiments, that the proposed strategies provide higher accuracy than the original ones in most cases.

6. ACKNOWLEDGEMENT
This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. NRF-2014R1A2A1A10054151).

7. REFERENCES
ABOUT THE AUTHORS:

Won-Seok Hwang received the B.S. degree in Computer Engineering from Hanyang University, Korea at 2007, and earned the M.S. degree in Electronics and Computer Engineering from Hanyang University, Korea at 2009. His research interests include databases, data mining, social network analysis, and recommendation.

Sang-Wook Kim received a B.S. degree in Computer Engineering from Seoul National University at 1989, and earned M.S. and Ph.D. degrees in Computer Science from KAIST at 1991 and 1994. From 1995 to 2003, he served as an Associate Professor of Division of Computer, Information, and Communications Engineering at Kangwon National University. In 2003, he joined Hanyang University, where he currently is a Professor at Department of Computer Science and Engineering. His research interests include databases, data mining, social network analysis, and recommendation. From 2009 to 2010, Dr. Kim visited Computer Science Department at CMU as a Visiting Professor. From 1999 to 2000, he also worked with IBM Watson Research Center as a Post-Doc. He is now an Associate Editor of Information Sciences and is a member of ACM and IEEE.
Efficient Hibernation Resuming with Classification-based Prefetching Scheme for Embedded Computing Systems

Chien-Chung Ho
Dept. of Computer Science and Information Engineering
National Taiwan University
f99922110@ntu.edu.tw

Sheng-Wei Cheng
Dept. of Computer Science and Information Engineering
National Taiwan University
d02922004@ntu.edu.tw

Yuan-Hao Chang
Dept. of Computer Science
Academia Sinica
johnson@iis.sinica.edu.tw

Yu-Ming Chang
Dept. of Computer Science and Information Engineering
National Taiwan University
d00922001@ntu.edu.tw

Sheng-Yen Hong
Dept. of Computer Science and Information Engineering
National Taiwan University
p98922006@ntu.edu.tw

Che-Wei Chang
Dept. of Computer Science and Information Engineering
School of Electrical and Computer Engineering
College of Engineering
Chang Gung University
chewei@mail.cgu.edu.tw

ABSTRACT

With the rapid growth of embedded computing system markets, e.g., intelligent home appliances and smart TVs, vendors and researchers are developing more user-friendly interfaces and seeking to provide more sophisticated applications with better functionalities. Such a developing trend would prolong the initialization time of these embedded computing systems. Hibernation (or suspend-to-disk) that retains a computing system’s state after power recycling is regarded as a solution to reduce the booting time of systems and applications to meet the requirement of user experiences. In contrast to the existing hibernation techniques that dump most of the memory pages to the secondary storage, we propose a classification-based prefetching scheme to improve the system performance on both of the hibernation and resuming with minimized I/O overheads by jointly considering the system/application behaviors and the usage patterns of memory pages. The proposed scheme was also implemented in Linux kernel with an evaluation board to show the capability of the proposed scheme.

1. INTRODUCTION

In the past decade, embedded computing has gained its momentum in the market, and vendors never stop developing more user-friendly interfaces and sophisticated applications with better and more functionalities. However, such a developing trend also prolongs the initialization time of the embedded computing systems, and some embedded systems, such as smart TVs, have limited tolerance for the time on initializing the system after it is powered on. As a result, fast booting has become the key technology to the success of these applications. When systems are further required to restore previous using states, the fast booting design is even more complicated [1, 2]. To address the design challenge, the hibernation technology was developed to let the system enter the suspension state and quickly resume back to the previous states. All (or most of) the data in the main memory are dumped to the secondary storage for hibernation, and loaded back to the main memory to recover the system states for resuming [3, 4]. As the memory footprints of applications and systems keep growing, the system resuming time is increasing. Such an observation motivates this work to propose an intelligent hibernation technique to reduce the resuming time and keep all application/system states.

A well-known hibernation implementation is TuxOnIce which is integrated into Linux kernel [5]. In the implementation, the resuming procedure comprises of two stages: The first stage loads the urgent image for system booting, and the second stage loads the hibernation image to resume the system’s status before the hibernation [1, 2, 6]. More specifically, when the system is powered on, the urgent image is first loaded to recover the basic functionalities of the kernel. After the urgent image is loaded to the main memory, the system is ready to allow users to login and execute applications. Subsequently, the hibernation image is loaded to the main memory to resume the saved application states by a background process. Although hibernation provides an excellent solution to address the need of fast booting, it becomes more and more challenging due to the need of improving the hibernation/resuming performance for the sys-

Categories and Subject Descriptors


General Terms

Management, Performance, Design

Keywords

Prefetch, Snapshot Image, Suspend, Resume.

Copyright is held by the authors. This work is based on an earlier work: RACS’14 Proceedings of the 2014 ACM Research in Adaptive and Convergent Systems, Copyright 2014 ACM 978-1-4503-3060-2. http://dx.doi.org/10.1145/2663761.2664215.
s with more and more complicated applications running simultaneously.

Researchers have proposed various designs to improve the efficiency of hibernation procedures [3, 7, 8]. Evaluations of the different methodologies were reported to show the superiority of each implementation [3]. Based on the analysis results on some specific applications, e.g., digital camera, some studies proposed an optimized initialization order and determined when data should be prefetched to the buffer to reduce the star-up time [7, 8]. Some works showed the steps towards a fast-booting Linux kernel using non-intrusive methods. Moreover, targeting embedded systems with temporal constraints, it shows how fast the specific real-time scheduling frameworks can be loaded and started during the system’s boot-up process [9]. Furthermore, some researchers proposed the methodologies to shrink the hibernation image without sacrificing user experiences in the resuming process of hibernation, but the required time on producing the hibernation image was not considered [10].

In addition to optimizing the performance of booting procedures, hibernation is able to save the system’s energy consumption as well [11, 12, 13]. Non-volatile memories, such as NAND flash and NOR flash, were extensively adopted in embedded systems to store the executable programs, and it is thought to be a fast booting device for operating systems over embedded systems [14]. Thus, some works further explored non-volatile memory with faster access speed to reduce the time of booting processes [15, 16, 17].

Unfortunately, compared to traditional hard drives, the unique characteristic (e.g., out-place-update) of non-volatile memory (e.g., flash memory) imposes an additional burden on system software designs. Moreover, this more likely leads to a longer initialization time while a systems is booting. Recently, some excellent works were done to tackle the issues by a trade-off between the run-time performance and the booting performance on flash-based embedded systems [18, 19, 20, 21], where their basic concept is to prioritize the metadata (indices of file systems) for quickly rebuilding application files so as to improve the time of system booting. However, there is little work that explores the system and application behaviors and the usage patterns of memory pages to optimize the performance of both hibernation and resuming of embedded computing systems, such that the system’s hibernation and resuming performance is less dependent of the number of applications running on the system.

In this work, we are interested in the design of fast hibernation and resuming schemes by exploring the system (application) behaviors and the usage patterns of memory pages. Thus, a classification-based prefetching scheme is proposed to jointly consider the system/application behaviors and the usage patterns of memory pages, so as to minimize the time delays of system hibernation and resuming. In the proposed scheme, a new page classification method, called analysis-based page classification, is designed to analyze and classify the loading priorities of pages according to their types and usage patterns; as a result, the proposed scheme can correctly (1) identify which pages should be put into the hibernation image and (2) define the sequences on loading pages from the secondary storage to the main memory with minimized time delays caused by page faults. The proposed scheme was implemented in Linux kernel ported on an evaluation board called “MStar Titania,” and compared with well-known Linux-based hibernation scheme “TuxOnIce” [5]. On this evaluation board, a series of experiments was conducted to evaluate the capability of the proposed scheme. The results show that the proposed scheme could greatly reduce the required time of hibernation and resuming processes from 7.18% to 14.24% on average under different investigated scenarios.

The rest of this paper is organized as follows: Section 2 presents the system architecture and research motivation. Section 3 presents the proposed classification-based prefetching scheme. Section 4 evaluates the effectiveness of the proposed scheme based on a real implementation. Section 5 concludes this work.

2. SYSTEM ARCHITECTURE AND MOTIVATION

2.1 System Architecture

As shown in Figure 1, a typical system that supports hibernation and resuming processes includes four modules in its memory management. These modules are Hibernator, Shrinker, Swapper, and Prefetcher. Hibernator aims to create a hibernation image that stores the system states including the running applications before hibernation. After that, Shrinker is used to reduce the size of the hibernation image to reduce the time loading/storing the image by means of evicting some pages to the swap space; this process is also referred to as virtual page reclamation. Swapper is responsible to handle the page fault and migrate the memory pages from the swap space to the physical memory, e.g., RAM. For better user experience, Prefetcher might be adopted to prefetch those pages belonging to some early-start applications; in this way, several page faults could be possibly avoided due to the on-demand paging policy in a typical memory management system.

![System Architecture](image)

During the hibernation process, the hibernator creates a snapshot image (called urgent image) containing the memory pages that store important system information, e.g., kernel pages, and creates a hibernation image for the remaining pages. If the hibernation image is too large, Shrinker is invoked to reduce the image size so as to reduce the I/O overhead on storing/loading the image. A common practice for Shrinker is to (1) keep the most-recently-used (MRU) memory pages in the hibernation image, and (2) evict the
least-recently-used (LRU) memory pages from the hibernation image and swapped out them to the swap space in the secondary storage, e.g., hard disk drives (HDDs) or solid state drives (SSDs). These evicted pages are considered invalid pages and can be brought in on the occurrence of page faults if they are not prefetched. Thus, during the resuming of the system, the hibernation image can be detected and loaded from the secondary storage to main memory, followed by a series of resuming processes to let the system go back to the previous state before hibernation.

Figure 2 depicts the detailed timeline of the resuming process on resuming a hibernated system. While the system is powered on, the urgent image is first loaded to recover the basic functionalities of the kernel. After the urgent image is loaded to main memory, the system is ready to allow users to login and execute applications. Subsequently, the hibernation image is loaded to main memory to resume the previously saved applications by a background process. Note that the loading of the hibernation image and the execution of user/system applications could be performed simultaneously once the urgent image is completed loaded back to main memory. As long as the hibernation image is fully loaded back to main memory, the whole resuming process is completed and the system resumes to the previous state.

![Figure 2. The timeline of the resuming process.](image)

### 2.2 Motivation

Hibernation images usually have size limitation due to the consideration of the hibernation performance and complexity. Thus, most of the hibernation techniques such as TuxOnIce conduct page classification according to the least-recently-used (LRU) policy, which is a simply but effective approach to select important pages and produce the hibernation image. However, with the LRU policy, pages that are needed during the system’s resuming might be removed from the hibernation image. As a result, the lack of such pages in the hibernation image might result in page faults to seriously deteriorate the performance of system resuming. For example, the hibernation image is produced when some applications (e.g., video player) that uses a large amount of pages are running. Based on the LRU policy, those pages might be put into the hibernation image but some important pages might thus removed out of the image due to the consideration of the image size. If a page that is not included in the hibernation image is requested or accessed during the system’s resuming, the missing page would trigger a page fault and the page fault handler will be invoked to load the missing page to main memory. Thus, the system will encounter a long period of time delay to wait for the loading and reallocating the missing pages.

On the other hand, it is important to select a proper size to create the hibernation image since the performance of hibernation and resuming processes is highly related to the image size. If the hibernation image is too large and includes too many data pages, the large amount of data pages might exceed the size limitation and would introduce a lot of time on loading the image. If the hibernation image is too small and includes too few data pages, a large amount of page faults might occur if a lot of pages that are not in the hibernation image are accessed during the resuming process. This is because the pages accessed by applications or the system kernel during the system’s resuming have to be loaded from secondary storages to resume the ready-to-run state, and thus will introduce a lot of time delay. Thus, the performance on resuming applications is related to the number of page faults. That is, we can reduce the resuming time (or improve the resuming performance) by reducing the number of page faults if we can assign a proper loading sequence to those demanded pages.

Moreover, the resuming time is also affected by the resuming sequence of hibernated tasks. Some researches (e.g., scheduler-assisted prefetching [22]) show that the process of demand paging can be significant improved by the prediction obtained from the analysis of the memory access pattern of tasks. The effect of resuming sequence can be better explained by a simple example, as shown in Figure 3. In the example, two different resuming sequence with two corresponding running queues, namely RunQueue 1 and RunQueue 2, are compared, and each running queue comprises three tasks, i.e., task A, task B, and task C. Suppose that task A and task C are the urgent system tasks which should be resumed and operated as soon as possible, and task B is a user task (such as video task) that will execute with a long delay. Assume that the resuming process starts to load back the hibernation image after the current task A finished. In the case of RunQueue 1, the user task B is first executed; however, the user task B is a lengthy job and will delay the time in resuming task c which is a system task and is more urgent, as compared to the case of RunQueue 2. This will directly degrades the system performance due to the page faults of system tasks, and thus deteriorates user experience.

In this paper, we aim at improving the performance of system hibernation and application resuming, and the technical problem falls on how to classify and determine pages that should be included in the hibernation image and how to decide the sequence/order to load pages from the secondary storage.

![Figure 3. The timeline of the resuming process.](image)
3. EFFICIENT HIBERNATION RESUMING WITH CLASSIFICATION-BASED PREFETCHING SCHEME

3.1 Overview

In this section, we first present a classification-based prefetching scheme to improve the performance of hibernating and resuming in embedded computing systems. Then, we present an efficient hibernation resuming prefetcher scheme to reorder the resuming sequence in the consideration of the access pattern of tasks. Our goals are to minimize the time overhead on loading pages during the system’s resuming and to reduce the overhead by the improper resuming sequence. A new page classification method, called analysis-based page classification, is implemented incorporated with a shrinker to analyze and classify the loading priority of each page for the resuming process; and the priority information can assist kernel to load pages that are needed immediately. We also propose an efficient prefetcher design to classify the different tasks and reorder their resuming sequence, so as to reduce the traffic overhead and provide a better user experience.

Figure 4. The overview of the proposed classification-based prefetching scheme.

To correctly identify the pages that need to be included in the hibernation image, the proposed prefetching scheme analyzes the page usage behavior of each task before preparing the hibernation image. Based on the analyzed results, it performs page partition by classifying the pages into different priorities of page sets, and page sets with higher priorities are put in front of those with lower priorities in the hibernation image, as shown in Figure 3.1. When the system starts to resume from the hibernation state, the proposed scheme would prefetch pages from the hibernation image to main memory in decreasing order of priority of the page sets. With the proposed scheme, the performance of the system resuming can be obviously improved. Furthermore, to effectively reduce the traffic overhead caused by an improper resuming sequence, we also propose an efficient prefetcher design incorporated with a task classifier to reorder the resuming sequence. The proposed scheme can further improve the performance of hibernation resuming and achieve a better user experience. We will explain the technical details on page classification and the proposed prefetching scheme in later sections. In Section 3.2, the design of the page classification, i.e., the analysis-based page classification, will be presented. Section 3.3 will describe the detail of the efficient hibernation resuming prefetching scheme.

3.2 Analysis-based Page Classification

The proposed analysis-based page classification is integrated into Shrinker to correctly classify the priorities of pages so that Shrinker knows which pages can be evicted from the hibernation image. When the system is requested to enter the hibernation mode, Shrinker is triggered to shrink the image size so as to reduce the I/O overhead on storing/loading the hibernation image. If Shrinker includes too many data pages into the image, the number of page faults will be significantly reduced during the system’s process. However, it will result in the longer time latency on waiting the system to load the image from the secondary storage devices to main memory. In addition, the hibernation image usually has size limitation (e.g., 1GB) in many embedded computing systems. Thus, if Shrinker includes too few data pages into the image, the system resuming process would be very efficient and the prefetching process can be quickly started. Nonetheless, the lack of many data pages in the hibernation image might result in a large amount of page faults once some applications try to execute during the system’s resuming; as a result, it will result in longer time delay to finish the system resuming and lead to bad user experience. Thus, the proposed analysis-based page classification prevents Shrinker from eliminating too many pages of the system tasks and help Shrinker to select correct pages into the hibernation image by considering/analyzing the page access behaviors of applications/tasks.

Before preparing the hibernation image, Shrinker will adopt the proposed analysis-based page classification to decide which page should be included into the hibernation image. For the urgent memory pages used to support the system execution (e.g., kernel code and device drivers), we store them in the urgent image, and the urgent image has to be first loaded from the secondary storage to main memory so as to start the resuming process. For the remaining memory pages that are not included in the urgent image, Shrink invokes the proposed analysis-based page classification (see Algorithm 1) to classify these pages and assign them with different priorities according to their allocation types and usage status. Algorithm 1 evaluates the behavior of each currently used pages in main memory and classifies them into the different page sets. For any given page \( p_i \) that is being examined, the tasks that might access the page \( p_i \) will be first detected, and those tasks are analyzed to find out their behaviors (Lines 3–4). If the examined page is only accessed by the kernel and system tasks, it will be put into the high priority page set \( P_H \) (Lines 5–6). If the examined page is only accessed by the user application tasks, it will be put into the middle priority page set \( P_M \) (Lines 8–9). If the examined page might be accessed by both of the kernel tasks and the user application tasks, it will be put into the low priority page set \( P_L \) (Lines 10–11). After the page classification, the hibernation image can be created by the three different priority page sets (Line 12). It is worth noting that to prevent the long latency on loading the hibernation image during the system’s resuming, we classify pages into three different priorities of page sets that constructs the hibernation image from the highest priority to the lowest one, and each page set occupies one partition of the hibernation.

Figure 3. The overview of the proposed efficient hibernation resuming prefetching scheme.
Thus, each partition can be independently accessed with the specific sequence on the prefetching process, and higher priority page sets are loaded to main memory before lower ones are loaded.

**Algorithm 1: Analysis-based Page Classification**

**Input:** The currently used page set $P = \{p_1, p_2, \ldots, p_n\}$ and the executing task set $T = \{t_1, t_2, \ldots, t_k\}$

**Output:** A Hibernation Image $H$ consists of the different priority page sets, where $H = P_H \cup P_M \cup P_L$

1. $H$ ← $\emptyset$
2. $P_H$ ← $\emptyset$; $P_M$ ← $\emptyset$; $P_L$ ← $\emptyset$
3. for $p_i \in P$ do
   // Analyze the access behavior and find the corresponding task
   4. $t_i$ = AnalysePageAccessBehavior($p_i$);
   5. if $t_i$ is an unnecessary task while system resuming then
      6. EvictUnnecessaryPage($p_i$);
   else
      8. if $t_i$ is an exact kernel task then
         9. // Put the page into the high prio. page set
            $P_H$ ← $P_H$ $\cup \{p_i\}$
      else
         11. if $t_i$ is an exact user task then
             12. // Put the page into the low prio. page set
                $P_L$ ← $P_L$ $\cup \{p_i\}$
          else
             13. // Put the page into the middle prio. page set
                $P_M$ ← $P_M$ $\cup \{p_i\}$
      // Create the hibernation image with the three different priority page sets
15. return $H$ ← $P_H$ $\cup P_M$ $\cup P_L$;

The analysis-based page classification can be better explained with the example shown in Figure 5: Consider a system with 6 data pages, i.e., $Page_1$-$Page_6$, in main memory. Suppose that the 6 pages do not belong to the urgent image. When the system is preparing to enter the hibernation mode, the 6 data pages are evaluated by Algorithm 1 before preparing the hibernation image. The evaluated result shows that: $Page_1$ and $Page_5$ will be accessed by the kernel tasks only, $Page_3$ and $Page_4$ will be accessed by the user tasks only, and $Page_2$ is accessed by both the kernel and user tasks. $Page_6$ is regarded as unnecessary page for the hibernation image, and it will be dumped into the swap space and then be evicted from main memory. After the analysis of page behavior is done, $Page_1$ and $Page_5$ are put into the high priority page set $P_H$, $Page_2$ and $Page_4$ are put into the middle priority page set $P_M$ and $Page_3$ and $Page_6$ are put into the low priority page set $P_L$. Then, we will use the 3 page sets to construct the hibernation image.

### 3.3 Online Prefetching Process

When the system tries to resume the previous state from the hibernation mode, it has to load the hibernation image into main memory to finish the resuming process. In the implementation of most realistic systems, the image loading process is divided into two phases: The first phase is to load the urgent image (or the urgent page set), and the second phase is to load the hibernation image, as shown in the general approach of Figure 6. The purpose of the first phase is to bring the urgent image consisting of the device driver and kernel image into the main memory, which is indicated as “LoadPageSet1” in Figure 6. After the first phase, the system can start to execute the application and tasks after the first phase is finished. Then, the kernel will go to second phase to prefetch the hibernation image and to bring the pages of applications into the main memory, which is indicated as “LoadPageSet2” in Figure 6. However, the general approach of the second phase is to load a large size image from the secondary storage into main memory all at once. Such an approach will not only result in the significant delay on image loading but also lead to bad user experience. The rationale behind this is that redundant-extra page faults might be triggered when some applications try to access the evicted pages that are not included in the hibernation image; and the system has to load the demanded/accessed pages into the main memory so that the prefetching process is delayed.

In contrast to the general approach, the proposed scheme includes an online prefetching process to improve the system’s resuming performance based on the page sets classified by the proposed analysis-based page classification. The online prefetching process loads different priority page sets separately at run time, so as (1) to avoid the long latency caused by loading the whole image all at once and (2) to reduce the possibility of page faults caused by the requests to access the pages that have not yet been loaded to main memory. After the high priority page set is loaded into main memory, some high priority applications can be started immediately so that a better user experience can be achieved with fewer pages loaded to main memory before applications can be resumed. Thus, the online prefetching process can achieve the desired system resuming performance and good user experience by sequentially loading the page sets from the highest priority one to the lowest one. After finishing the prefetching process, the system resuming process is completed and resumed.
to the state before hibernation. Figure 6 shows the comparison of the resuming process between the general approach and the proposed prefetching scheme. With prefetching the page sets according to their priorities, it can be guaranteed that the pages of system tasks could be re-allocated on main memory earlier than those of user application tasks. Such an online prefetching strategy can avoid the longer loading latency and provide a better user experience.

![Figure 6. The comparison of the resuming process between the general hibernation process and the proposed classification-based hibernation.](image)

### 3.4 Semi-auto Classification

The performance of the resuming process can be also affected by the access pattern of tasks since the resuming sequence is highly related to the access pattern of the tasks. This is because that the hibernation status depends on the processing state of each task when the system is requested to enter hibernation mode. The using scenario varies due to the different types of applications (e.g., Internet browsing, video playback decoding, routing work). That is, the resuming sequence can be easily determined by the active tasks. However, as we mentioned in the previous sections, the improper resuming sequence of hibernated tasks will result in the degradation of resuming performance since it might invoke a huge overhead caused by loading minor user pages.

The system will be in a situation that a large number of page faults will be triggered for on-demanded pages requested by the resumed user tasks. This case will lead the system to generate more and more I/O bound transmissions because of page faults, so as to limit the resuming performance. Therefore, the performance of resuming process can be well improved if the resuming sequence can be properly determined. To be more specific, we aim at proposing the mechanism to classify tasks by a user-specified attribute and to generate a more efficient resuming sequence according to user experience. In this section, we will propose a semi-auto classification in which user can specify the resuming priorities of specific tasks, and the classification could cooperate with the proposed online prefetching process to reduce the potential overhead caused by the page faults of user tasks.

The semi-auto classification is very similar to the proposed analysis-based classification. Before preparing the hibernation image, Shrink invokes the proposed analysis-based page classification (see Algorithm 2) to classify these pages and assign them with different priorities according to their allocation types, usage status, and user specifications. Algorithm 2 evaluates the behavior of each currently used pages in main memory and classifies them into the different page sets. For any given page \( p_i \) that is being examined, the tasks that might access the page \( p_i \) will be first detected, and those tasks are analyzed to find out their behaviors (Lines 3–4). If the examined page belongs to a specific task defined by users, it will be put into the corresponding priority page set (Lines 5–6). If the examined page does not belong to any specific task defined by users, it will be examined to decide its corresponding page set by executing Algorithm 1. Then, the proposed online prefetching process can use the hibernation image generated by the proposed semi-auto page classification to efficiently resume from the hibernation state without any modification.

#### Algorithm 2: Semi-auto Page Classification

**Input:** The currently used page set \( P = \{p_1, p_2, \ldots, p_n\} \) and the executing task set \( T = \{t_1, t_2, \ldots, t_k\} \)

**Output:** A Hibernation Image \( H \) consists of the different priority page sets, where \( H = P_H \cup P_M \cup P_L \)

1. \( H \leftarrow \emptyset \);
2. \( P_H \leftarrow \emptyset; P_M \leftarrow \emptyset; P_L \leftarrow \emptyset \);
3. for \( p_i \in P \) do
   // Analyze the access behavior and find the corresponding task
   4. \( t_j = \text{AnalyzePageAccessBehavior}(p_i) \);
   5. if task \( t_j \) is predefined by user then
      6. Put the page into the predefined priority page set;
   7. else
      8. Run Algorithm 1 to classify this task;
   // Create the hibernation image with the three different priority page sets
9. return \( H \leftarrow P_H \cup P_M \cup P_L \); 

### 4. PERFORMANCE EVALUATION

#### 4.1 Experimental Setup

<table>
<thead>
<tr>
<th>CPU</th>
<th>MSStar TITANIA12 (672 MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Architecture</td>
<td>MIPS</td>
</tr>
<tr>
<td>Operating System</td>
<td>Android 2.1</td>
</tr>
<tr>
<td>Kernel</td>
<td>2.6.27</td>
</tr>
<tr>
<td>DRAM</td>
<td>128 MB</td>
</tr>
<tr>
<td>Flash Memory</td>
<td>256 MB</td>
</tr>
</tbody>
</table>

In this section, a series of experiments was conducted to show the effectiveness of the proposed classification-based prefetching scheme. In the experiment, we ported a well-known project *TuxOnIce* into the adopted evaluation board.
CPU utilization. Similar to the first scenario, the number of applications would constantly compute the value to boost the CPU utilization. Similar to the first scenario, the number of applications running on Android were computing-intensive tasks and each application allocated and accessed 10MB data of physical memory. To inspect the impact of memory size and each application, three kinds of application sets (i.e., three different scenarios) were running on the Android system on the evaluation board, where the corresponding specifications, i.e., CPU architecture, memory, DRAM and storage (flash memory), of the evaluation board are listed in Table 1. To compare these two schemes, we evaluated their performance in terms of two aspects, namely hibernation and resuming. The hibernation performance was evaluated based on the required period from the time of issuing hibernation command to that of powering off the evaluation board, whereas the resuming performance was evaluated based on the period from the time of pushing the power-on button to that of displaying the launcher (i.e., a state waiting for the user input).

To evaluate the performance of two schemes under different scenarios, three kinds of application sets (i.e., three different scenarios) were running on the Android system before the hibernation process starts. In the first scenario, all the applications running on Android were memory-intensive tasks and each application allocated and accessed 10MB data of physical memory. To inspect the impact of memory size (which was allocated to all the applications) on the performance, we varied the number of applications running on Android from 5 to 35, stepped by 5. We restricted the number of memory-intensive applications to a specific value since the physical memory on the evaluation board was limited and the operation system would automatically release (or kill) the applications by the low-memory killer on Android. In the second scenario, all the applications running on Android were computing-intensive tasks and each application would constantly compute the value to boost the CPU utilization. Similar to the first scenario, the number of computing-intensive applications was varied from 5 to 35, stepped by 5. In the last scenario, all the applications performed hybrid tasks, which means that each application executed memory-intensive and the computing-intensive tasks simultaneously. Likewise, the number of hybrid applications were varied from 5 to 35, stepped by 5.

4.2 Experimental Results

Figure 7 shows the time spent on hibernation and resuming processes respectively as well as the total time required to finish both two processes under TuxOnIce and the proposed scheme, where the x-axis specifies the number of memory-intensive applications running on Android and the y-axis denotes the required time in the unit of one second. As expected, with more applications running on Android, the time to complete the hibernation and resuming processes increases in accordance with the number of applications. This is because the larger number of active applications on the system consumes more storage space to store the hibernation image. In general, it is observed that the total time to finish the hibernation and the resuming processes under the proposed scheme is more efficient than that under TuxOnIce when the number of running applications is larger than 25, as shown in Figure 7(c). Since the proposed scheme could effectively identify the important applications by the proposed analysis-based page classification and semi-auto page classification algorithms, and evict the pages of those non-critical (or low priority) applications from hibernation image to save the time to suspend the running applications. Even though the proposed approach requires a little more time to prepare the hibernation image (see Figure 7(a)) due to the execution of the proposed analysis-based page classification and semi-auto page classification algorithms, the increased time is negligible and far less than the time saved from the online prefetching process as shown in Figure 7(b).

Figure 8 also shows the time spent on hibernation and resuming processes under the investigated schemes while all the applications before hibernation are computing-intensive.
The x-axis specifies the number of running applications and the y-axis denotes the consumed time in the unit of second. Similar to the previous results, as more applications run before hibernation, the hibernation and resuming processes take more time to complete. This is also as expected because more running applications require more storage space to store the hibernation image. We also observe that to perform hibernation process as well as resuming process under the proposed scheme takes less time than that under TuxOnIce when the number of running applications is larger than 25, as shown in Figure 8(c). On the other hand, with more applications running on the system, it is observed that the time to perform resuming process under the proposed scheme is more stable, compared to that under TuxOnIce. This is reasonable because the proposed analysis-based page classification and semi-auto classification in the proposed scheme only put the pages of those tasks in the hibernation image in the order from high priority to low priority, and the chosen number of pages of each task is quite stable in most cases such that the size of the created hibernation image proportionally increases with the increasing number of running applications, as shown in Figure 8(b). Although there is still a little timing overhead to execute the proposed classification algorithms under the proposed scheme, compared to that under the TuxOnIce, the increased time is negligible and far less than the time saved from the resuming process as shown in Figure 7(b).

Figure 9 shows performance of the investigated schemes with hybrid tasks running on the system before hibernation. As expected, the time to complete the hibernation and resuming processes increases when more applications run on the system, since the more applications would result in larger hibernation image that takes more time to suspend. It is observed that the hibernation and resuming time are longer than that with the settings in Figure 7 and Figure 8. The reason of this phenomenon is that the hybrid applications that simultaneously perform memory and computation tasks have a larger code size and allocate more memory pages in total; as a result, more dirty or high-priority pages might be generated from these kinds of applications, such that the hibernation image is enlarged.

As compared to TuxOnIce, the proposed scheme could effectively improve performance (or reduce the time) to execute the hibernation and the resuming processes, as shown in Figure 9(c). This is because the proposed analysis-based page classification algorithm would only keep those useful pages in the hibernation image so as to significantly reduce the opportunity of invoking a large amount of page faults. The proposed scheme shows the efficiency on the resuming process while the number of application is growing, as shown in Figure 9(b). In addition, the proposed scheme additionally performs prefetching for better user experience.

5. CONCLUSION

This work is motivated by the need to improve the hibernation and resuming performance of embedded computing systems in some applications such as smart TVs. In this paper, we proposed a classification-based prefetching scheme to improve the system performance on both hibernation and resuming processes with minimized I/O overheads by jointly considering the system (application) behaviors and the usage patterns of memory pages. In addition, two new page classification methods, called analysis-based page classification and semi-auto classification, are implemented with the cooperation of the existing shrinker to correctly analyze and classify the loading priority of each page so that the system’s resuming performance can be maximized with an online prefetching process. For the evaluation of the proposed scheme, all the approaches for hibernation and resuming processes were implemented on a real evaluation platform. The experimental results show that the proposed scheme could reduce the time of hibernation and resuming processes by 7.5%, 7.18% and 30.8% under memory-intensive, compute-intensive, and hybrid applications respectively, compared to the baseline scheme.

In the future, we shall explore the possibility to adaptively classify the importance of tasks to further optimize the hibernation and resuming performance in various system environments. Furthermore, we shall also explore the possibility to cooperate the proposed scheme with emerging non-volatile memories, such as PCM, to further investigate the extendability of the proposed scheme.

6. ACKNOWLEDGMENTS

This work was partially supported by the Ministry of Science and Technology under MOST Nos. 103-2221-E-001-012-MY2, 102-2221-E-001-009-MY3, 100-2221-E-002-120-MY3, 102-2221-E-002-078-MY2 and 103-2218-E-182-004-MY2.

7. REFERENCES


ABOUT THE AUTHORS:

Chien-Chung Ho received the B.S. degree in computer science and information engineering from National Chiao Tung University, Hsichu, Taiwan (R.O.C.) in 2009. He is currently a Ph.D. candidate of the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan (R.O.C.). His research interests include flash memory, storage systems and operating systems.

Sheng-Wei Cheng received the B.S. degree in computer science and information engineering from National Chiao Tung University in 2008 and the M.S. degree in computer science and information engineering from National Taiwan University in 2012. He is currently a PhD candidate in National Taiwan University. His research interests include embedded systems, real-time systems, and mobile networks.

Yu-Ming Chang received his M.S. degree in Networking and Multimedia of Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in June 2011. Now he is a PhD candidate in Computer Science and Information Engineering of National Taiwan University. He is also a chief engineer in Emerging System Lab, a division of Macronix International Co., Ltd., Hsinchu, Taiwan. Previously, he received his B.S. degree in Computer Science and Information Engineering from Chiao Tung University in June 2009. In addition, he was elected as a member of the Phi Tau Phi Scholastic Honor Society at M.S. degree in 2011. His primary research interests include operating systems, storage systems, and embedded systems.

Yuan-Hao Chang received his Ph.D. degree in Networking and Multimedia of Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in 2009. Now he is an assistant research fellow at the Institute of Information Science, Academia Sinica, Taipei, Taiwan, since Aug. 2011. Previously, he was an assistant professor at the Department of Electronic Engineering, National Taipei University of Technology, Taipei, Taiwan, between Feb. 2010 and Jul. 2011. His research interests include storage systems, embedded systems, and computer system architecture. He is a member of IEEE.
Che-Wei Chang is an assistant professor at the Department of Computer Science and Information Engineering, School of Electrical and Computer Engineering, College of Engineering, Chang Gung University, Taiwan. He received his Ph.D. degree in Computer Science and Information Engineering from National Taiwan University and won the Ph.D. Dissertation Award of the Institute of Information and Computing Machinery (IICM) in 2012. He was a visiting junior researcher at the Karlsruhe Institute of Technology (KIT), Germany and supported by the scholarship of KIT in 2011. He worked as a postdoctoral research fellow at the Research Center for Information Technology Innovation, Academia Sinica, Taiwan from 2012 to 2013. His research interests range from multi-core scheduling design to system software implementation.

Sheng-Yen Hong received the M.S. degree in computer science and information engineering from National Taiwan University in 2012. Now he is an software engineer in MediaTek Inc.