Bare Demo of IEEEtran.cls for IEEE Computer Society Conferences

Lin Lai
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: http://www.michaelshell.org/contact.html

Chang Li
Research School of Computer Science
Australian National University
Canberra, ACT, Australia
Email: spacegoing@gmail.com

***Abstract*—Trading strategies basing on both financial analysis and machine learning techniques are becoming increasingly popular due to their ability to capture micro market price movements and leverage big data. An important class of works are focusing on exploiting the structural relationships between companies for accurate stock price prediction. In this paper we develop an algorithm for learning the parameters of unary and binary potentials in binary markov random fields (MRFs) under the max-margin framework. We first show how to train unary potentials using market price data and Gaussian Mixture Models (GMMs). Then, we developed a graph-cut based algorithm to solve the inference problem exactly. We demonstrate the learning of potentials' parameters using a max-margin learning framework. Experiment is conducted by comparing performances between our formulation and conventional SVM method. Results show that our method outperforms SVM by 27.9% on train set and 40.5% on test set.**

1. Introduction

Quantitative trading algorithms using machine learning and data mining technologies have been raising much interest these years [7, 8, 12, 18]. Machine learning algorithms leveraging Big Data and novel hardware can form the basis for effective stock price movement reasoning and artificial intelligent trading decision making. This gives traders a wild range of new insights and opportunities to combine prior knowledge of financial market with non-observable information into trading strategies.

Price movement prediction is an extremely difficult optimization and mathematical modeling problem regarding its basis in stochastic movements and complex dynamic system.

Since the beginning of 2017, some blue chip stocks which have low valuations, solid market positions and steady earnings have been in hot pursuit of funds, with prices rapidly rising up. However, according to the Wind Financial Terminal, by May 25, 2017, more than 80% of stocks have been fallen, and specifically, the number of stocks which have declined by more than 20% has been near 1200, accounting for 40% of stocks (rule out the sub-new stocks). These phenomena show fierce differentiation of Chinas stock market.

Corresponding to the A share market, due to the great difference of configuration between industry and single stock, the performances of active funds also suffered a period of great division. The prices of funds which heavily invested in the blue chip stocks were driven up, boosting the yields and hitting new highs. But meanwhile, the net worth scale of some funds seriously shrank whose heavy warehouse stocks are those of SME stock market or defense industry plate, resulting heavy fall of multiple funds. Similarly, the performances of public enlisting funds represent extreme divergences.

Price prediction is a significantly difficult optimization and mathematical modeling problem regarding its basis in stochastic movement and complex dynamic system. The algorithms which can accurately predict the fluctuation of stock price bring huge underlying income for fund managers and financial investors. Machine learning algorithms leveraging big data are able to form the basis for effective stock price prediction and trading decision making, so quantitative trading algorithms using machine learning and data mining technologies have been raising much interest these years [7, 8, 12, 18]. Consequently, a wild range of new insights and opportunities are provided for the traders to combine prior knowledge of finance with non-observable information.

In this paper, we demonstrate the application of machine learning techniques together with fundamental financial analysis in trading strategies development. We model the complex relationships between companies by using the famous Markov Random Fields (MRFs) and leveraging the structure information implied in those relationships by Structural Support Vector Machine (SSVM). In order to encode complex consistency information between stocks, we trained two Gaussian Mixture Models (GMMs) (one for positive movement, one for negative movement). The cutting-plane algorithm was developed in favor of making the learning problem of SSVM tractable and convergence.

Key contributions in this paper are:

(i) Encoding consistency information between collabo-

rating companies using GMMs.

(ii) Modeling stock price movements using MRFs and SSVM.

The remainder of this paper is organized as follows. The next section discusses three main problems we focus on: energy function, inference and learning problem. Then we introduce our energy function formulation together with our exact inference using st-min-cut algorithm. The following section describes our MRFs with unary and pairwise potential functions' quadratic programming formulation under the max-margin framework (SSVM) and the cutting-plane algorithm for solving SSVM. In the last section we describe our experiment configuration and show the improvements made by our algorithm by comparing with previous works. We also give some real-world application insights in that section by back testing a trading strategy developed using our algorithm.

2. Related Work and Background

2.1. Markov Random Fields

Markov Random Fields are also known as *undirected graphical model* can be seen as a regularized joint log-probability distribution of arbitrary non-negative functions over a set of maximal cliques of the graph [1]. Let C denotes a maximal clique in one graph and $\boldsymbol{y}_C$ denotes the set of variables in that clique. Then the joint distribution can be written as:

$$p(\boldsymbol{y}) = \frac{1}{Z}\prod_{C}\Psi_C(\boldsymbol{y}_C) \tag{1}$$

where Ψ is called *potential functions* which can be defined as any non-negative functions and $Z = \sum_{\boldsymbol{y}}\prod_C \Psi_C(\boldsymbol{y}_C)$ which is a normalization constant. To infer labels which best explains input data set, we can find the *maximum a posteriori* (MAP) labels by solving $\boldsymbol{y}^* = \text{argmax}_{\boldsymbol{y}}\, p(\boldsymbol{y})$. Because potential functions are restricted to be non-negative, it gives us more flexible representations by taking exponential of those terms. Thus the joint distribution becomes:

$$p(\boldsymbol{y}) = \frac{1}{Z}exp(-\sum_{C} E_C(\boldsymbol{y}_C)) \tag{2}$$

where E is called *energy functions* which can be arbitrary functions. Therefore, *maximum a posteriori* problem is equivalent to *energy minimization* problem, which is also known as *inference*:

$$\begin{aligned}\boldsymbol{y}^* &= \underset{\boldsymbol{y}}{\text{argmax}}\, p(\boldsymbol{y}) = \underset{\boldsymbol{y}}{\text{argmax}}\left(exp(-\sum_{C} E_C(\boldsymbol{y}_C))\right) \\ &= \underset{\boldsymbol{y}}{\text{argmin}}(\sum_{C} E_C(\boldsymbol{y}_C))\end{aligned} \tag{3}$$

To optimize the performance we can also consider a weighted version of energy functions. In order to do this we can decompose energy functions over nodes $\mathcal{N}$, edges $\mathcal{E}$ and higher order cliques $\mathcal{C}$ [22] then add weights on them accordingly. Let $\boldsymbol{w}$ be the vector of parameters and ϕ be arbitrary feature function, then the energy can be decomposed as a set linear combinations of weights and feature vectors:

$$\begin{aligned}E(\boldsymbol{y};\boldsymbol{w}) = &\sum_{i\in\mathcal{N}} \boldsymbol{w}_i^U \phi^U(\boldsymbol{y}_i) + \sum_{(i,j)\in\mathcal{E}} \boldsymbol{w}_{ij}^P \phi^P(\boldsymbol{y}_i, \boldsymbol{y}_j) \\ &+ \sum_{\boldsymbol{y}_C\in\mathcal{C}} \boldsymbol{w}_C^H \phi^H(\boldsymbol{y}_C)\end{aligned} \tag{4}$$

where U denotes *unary* terms, P denotes *pairwise* terms and H denotes *higher order* terms (when $|C| > 2$ namely each clique contains more than two variables).

A weight vector $\boldsymbol{w}$ is more preferable if it gives the ground-truth assignments $\boldsymbol{y}_t$ less than or equal to energy value than any other assignments y:

$$E(y_t, w) \leq E(y, w)\ ,\ \forall y \neq y_t,\ y \in \mathcal{Y} \tag{5}$$

Thus the goal of *learning* MRFs is to learn the parameter vector $\boldsymbol{w}^*$ which returns the lowest energy value for the ground-truth labels y_t relative to any other assignments y [22]:

$$\boldsymbol{w}^* = argmax_{\boldsymbol{w}}(E(y_t, w) - E(y, w))\ ,\ \forall y \neq y_t,\ y \in \mathcal{Y} \tag{6}$$

So far we have introduced three main research topics of MRFs: definition of *energy function* (potential functions), *inference* problem (MAP or energy minimization) and *learning* problem.

As for energy function, our work focus on a generalization of k-means clustering known as Gaussian Mixture Models to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussians. The grabcut [20] algorithm is used to train GMMs and the final results are used to construct the unary terms. For pairwise terms, we use the Potts model introduced by Kohli et al. [15] to encode pairwise consistency. The *inference* problem is solved by using a graph-cut [3, 4] algorithm and the max margin framework [23] has been addressed to learn parameters of the energy function.

3. Energy function and exact inference

3.1. Construction of MRF (stocks' relationship) graph

As a complex system, the stock market is full of abundant of variables which affect the value of shares to rise or fall. Specifically, if some companies collaborate in their business, their stock prices even will consistently fluctuate. To concretely capture such kind of relationship among different companies, we can unfold it with an undirected graph (MRFs). Each node in the graph represents a single company, and the edge between a pair of nodes implies that the two related companies work in close collaboration.

In this paper, we will select the 300 stocks covered by SHANGHAI SHENZHEN 300 INDEX (CSI 300) as the experiment subjects (the data supplied by Wind Financial Terminal). Since collaboration exists among many corporations, if inference is decided by a message-passing algorithm (the complexity is exponential in the tree-width of the graph), the problem will become intractable. As mentioned above, we are capable of using minimum cuts to solve inference queries in polynomial time to figure out the collaboration and competition relationship among the 300 companies through *sougo.com* search engine:

1. Company A-Company B collaboration
2. Company A-Company B competition

To reduce the probability of mistakes in falsely judging the relationship, we set a threshold value as 1.5: if the number of results searching with the first query is 1.5 times bigger than that with the second one, we assume that the collaboration exists between the company A and B. Correspondingly, an edge will be used to connect the related nodes in the MRFs. Our spider code and graph data are available on Github https://github.com/spacegoing/sogou_spider.

The unary terms are measured using GMMs and pairwise terms are measured using Potts model, requiring the SSVM to learn the labeling from the feature vectors at the nodes.

3.2. Configuration of Energy Function

In this section we described the configuration of our energy function. We mainly introduce the *GrabCut* algorithm which we use to generate our unary terms. The *Potts model* is used as our pairwise terms. Thus our energy function can be written as:

$$E(\boldsymbol{y}; \boldsymbol{w}) = \theta^U \sum_{i \in \mathcal{N}} \boldsymbol{w}_i^U \phi^U(\boldsymbol{y}_i) + \theta^P \sum_{(i,j) \in \mathcal{E}} \boldsymbol{w}_{ij}^P \phi^P(\boldsymbol{y}_i, \boldsymbol{y}_j) \tag{7}$$

where θ^U and θ^P are parameters to be optimized.

The *GrabCut* algorithm was proposed by Rother et al. [20] in order to solve background foreground segmentation problem (see figure 1). They first defined MRFs over an labeled image and then use *graph-cuts* [5] method to do the inference. The equivalence of the labeled image in our work is the stocks relation graph described in section 3.1.

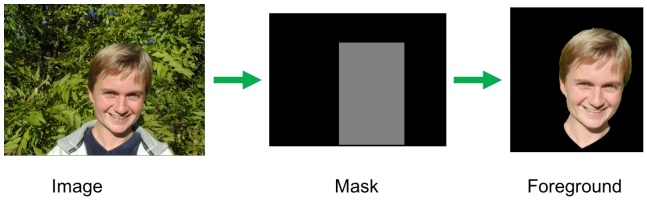

Figure 1: Picture on the left is the original picture. Picture on the middle is a user defined mask. The task is to extract foreground pixels within that rectangle. On the right is the ground truth foreground.

Algorithm 1 GrabCut training algorithm

1: **repeat**
2: *Assign GMM components* to stocks:
$\boldsymbol{k}_i^* = \text{argmin}_{\boldsymbol{k}_i} \psi^U(\alpha_i, \boldsymbol{k}_i, \boldsymbol{\theta}, \boldsymbol{z}_i)$
3: *Learn GMM parameters* from data z:
$\boldsymbol{\theta} = \text{argmin}_{\boldsymbol{\theta}} \sum_{i \in \mathcal{N}} \psi^U(\alpha_i, \boldsymbol{k}_i, \boldsymbol{\theta}, \boldsymbol{z}_i)$
4: *Estimate segmentation*: graph-cuts inference:
$\min_\alpha \min_{\boldsymbol{k}} E(\alpha, \boldsymbol{k}, \boldsymbol{\theta}, \boldsymbol{z})$
5: **until** convergence

We use two of their contributions: estimating color distribution (foreground and background) using *Gaussian Mixture Models* (GMMs) and an *EM* like two-step algorithm to train their model.

Suppose there are N stocks in our stocks graph. In order to construct MRFs, we first defined an energy function (4) with unary and pairwise terms:

$$G(\alpha, \boldsymbol{k}, \boldsymbol{\theta}, \boldsymbol{z}) = \sum_{i \in \mathcal{N}} \psi^U(\alpha_i, \boldsymbol{k}_i, \boldsymbol{\theta}, \boldsymbol{z}_i) + \sum_{(i,j) \in \mathcal{E}} \psi^P(\alpha_i, \boldsymbol{z}_i) \tag{8}$$

where i is the index of pixels, $\alpha \in 0, 1$ is the label for pixel i. 0 is for the background and 1 is for the foreground. $\boldsymbol{z}$ denotes the pixel vector in RGB color space. $\boldsymbol{k}$ and $\boldsymbol{\theta}$ are all parameters vectors.

In our configuration, we use the graph described in section 3.1 in replacement of the image. Each node in the graph represents a stock instead of pixel and the edge between stocks represents their pairwise relationship. The label $\alpha \in 0, 1$ equals 1 when the stock price has a positive movement, and vice versa. $\boldsymbol{z}$ denotes the stock's market price vector instead of pixel's RGB value

$$\boldsymbol{z} = \begin{bmatrix} \text{open} \\ \text{high} \\ \text{low} \\ \text{close} \\ \text{volume} \\ \text{macd}_5 \\ \text{macd}_{10} \\ \text{kdj}_9 \end{bmatrix} \tag{9}$$

Other parameters are the same with their configurations.

The *Gaussian Mixture Models* (GMMs) with K components (typically $K = 5$) is used for generating unary terms. Two GMMs, one for positive movement and one for negative movement, are jointly trained by the algorithm. $\boldsymbol{k} = \boldsymbol{k}_1, \ldots, \boldsymbol{k}_i, \ldots, \boldsymbol{k}_N$ with $\boldsymbol{k}_i \in 1, \ldots, K$ assigns each stock (node) i to a unique GMMs component. The component is either belonging to positive movement's GMMs or negative movement's GMMs, which is depended on the label $\alpha_i \in 0, 1$. $\boldsymbol{\theta}$ is the parameter vector which contains parameters of standard GMMs plus *mixture weighting coefficients* [20].

The pairwise function ψ^P in (8) is defined as a smoothness indicator which measures both feature vector (stock

price vector) and spatial distances (graph distances) simultaneously. It is used to encourage coherence of similar pixel pairs. Energy function (8) was later used to construct an *st min-cut* graph which can be inferred efficiently using *graph-cuts* [5] algorithm.

To optimize the performance, a two-step learning algorithm is used. The algorithm first re-assign GMMs components ($\boldsymbol{k}$) to each pixel then update parameters $\boldsymbol{\theta}$ with new assignments. The result of the trained GMMs are used directly into *graph-cuts* algorithm 1 as unary terms. Finally the label α_i for each pixel i is inferred jointly using *graph-cuts* algorithm. This whole procedure is repeated until convergence (or reaches termination conditions). We briefly summarized this procedure in Algorithm 1

In this paper we use GMMs trained by GrabCut algorithm for our unary terms ϕ^U in equation (7).

A Potts model is defined as

$$\psi_{ij}^P(y_i, y_j) = \frac{\lambda}{d_{ij}} [\![y_i \neq y_j]\!] \exp\left\{ -\frac{\|x_i - x_j\|^2}{2\beta} \right\} \quad (10)$$

where d_{ij} is the graph distance between stocks i and j. x_i and x_j are stock market price vectors. The pairwise function ϕ^P in our energy function (7) becomes

$$\phi^P(\boldsymbol{y}_i, \boldsymbol{y}_j) = \begin{cases} 0 & \text{if } i = j \\ \psi_{ij}^P(y_i, y_j) & \text{otherwise} \end{cases} \quad (11)$$

3.3. Exact Inference

Exact inference on MRFs has been extensively studied in past years. Researchers found that, energy functions which can be transformed into quadratic pseudo-Boolean functions [13, 14, 21] are able to be minimized exactly using *graph-cuts* like algorithms [9, 11] when they satisfy submodularity condition [2]. We mainly focus on describing the *st-min-cut* graph constructed for energy function containing unary and pairwise potentials.

The construction is explained in Figure 2. It denotes construction for unary and pairwise terms (see [17]). For unary edges (4 edges on both sides), weights on each edge are corresponding to values in input unary terms accordingly. For pairwise edges (2 edges in the middle), both edges share the same weight which equals to the input pairwise term. Every cut corresponds to an assignment to the random variables, where variables associated with nodes in the $\mathcal{S}$ set take the value one, and those associated with nodes in the $\mathcal{T}$ set take the value zero. With slight abuse of notation, we use the variables to denote nodes in our graph.

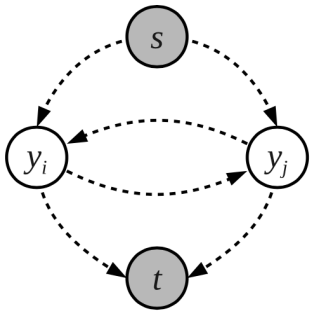

Figure 2: st-graph construction [10] for unary and pairwise terms.

4. Learning the MRFs

Traditional Support Vector Machines (SVMs) is used to solve binary classification problem. Crammer and Singer [6] extended SVM into multiclass classifier by generalizing the concept of *margin* to measure multiclass distances and a quadratic objective function was constructed. To approach structural prediction Tsochantaridis et al. [23] extends their approach by specifying discriminant functions that exploit the structure and dependencies within label space $\mathcal{Y}$.

When training Structural SVMs, the parameter vector $\boldsymbol{w}$ is determined by minimizing the (regularized) risk on the training set $(x_1, y_1), ..., (x_n, y_n)$. Risk is measured through a user-supplied loss function $\Delta(y, \hat{y})$ that quantifies how much the prediction $\hat{y}$ differs from the correct output y. Note that Δ is typically nonconvex and discontinuous and there are usually exponentially many possible structures $\hat{y}$ in the output space $\mathcal{Y}$. The Structural SVM formulation[23] overcomes these difficulties by replacing the loss function Δ with a piecewise linear convex upper bound (margin rescaling). To train Structural SVMs we then solve the following convex optimization problem

$$min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 + \mathcal{C} \sum_{i=1}^{n} \Big[max_{\hat{y} \in \mathcal{Y}} \big[\Delta(y_i, \hat{y}) + \boldsymbol{w} \cdot \phi(x_i, \hat{y}) \big] - \boldsymbol{w} \cdot \phi(x_i, y_i) \Big] \quad (12)$$

Despite the typically exponential size of $\mathcal{Y}$, this optimization problem can be solved efficiently using cutting-plane or stochastic gradient methods.

Let $\mathcal{Y}_t = \{0, 1\}^n$ be the set of all possible assignments for the t-th training example. The (margin-rescaling) max-margin approach formulates learning as a quadratic programming optimization problem. Let

$$\delta\phi_t(\boldsymbol{y}) \triangleq max_{\boldsymbol{y} \in \mathcal{Y}} \big[\Delta(\boldsymbol{y}_t, \boldsymbol{y}) + \boldsymbol{w} \cdot \phi(x_t, \boldsymbol{y}) \big] - \boldsymbol{w} \cdot \phi(x_t, \boldsymbol{y}_t)$$

be the difference between feature representations for some assignment $\boldsymbol{y}$ and the t-th ground-truth assignment $\boldsymbol{y}_t$, $C > 0$ is a regularization constant, and $\Delta(\boldsymbol{y}, \boldsymbol{y}_t)$ measures the loss between a ground-truth assignment $\boldsymbol{y}_t$ and any other assignment. In our work we use the Hamming loss, which measures the proportion of variables whose corresponding assignments disagree. More formally, the Hamming loss is defined as $\Delta(\boldsymbol{y}, \boldsymbol{y}') = \frac{1}{n} \sum_{i=1}^{n} [\![y_i \neq y_i']\!]$, where $[\![P]\!]$ is the indicator function taking value one when P is true and zero otherwise.

The number of constraints in the Quadratic Programming problem is exponential in the number of variables,

Algorithm 2 Learning lower linear envelope MRFs.

input training set $\{\boldsymbol{y}_t\}_{t=1}^T$, regularization constant $C > 0$, and tolerance $\epsilon \geq 0$
initialize active constraints set $\mathcal{A}_t = \{\}$ for all t
repeat
  solve $\text{MAXMARGINQP}\big(\{\boldsymbol{y}_t, \mathcal{A}_t\}_{t=1}^T, \boldsymbol{D}^2, \mathbf{0}\big)$ to get $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\xi}}$
  convert from $\hat{\boldsymbol{\theta}}$ to (a_k, b_k) representation
  for each training example, $t = 1, \ldots, T$ **do**
    compute $\boldsymbol{y}_t^\star = \text{argmin}_{\boldsymbol{y}} E(\boldsymbol{y}; \hat{\boldsymbol{\theta}}) - \Delta(\boldsymbol{y}, \boldsymbol{y}_t)$
    if $\hat{\xi}_t + \epsilon < \Delta(\boldsymbol{y}_t^\star, \boldsymbol{y}_t) - E(\boldsymbol{y}_t^\star; \hat{\boldsymbol{\theta}}) + E(\boldsymbol{y}_t; \hat{\boldsymbol{\theta}})$ **then**
      $\mathcal{A}_t \leftarrow \mathcal{A}_t \cup \{\boldsymbol{y}_t^\star\}$
    end if
  end for
until no more violated constraints
return parameters $\hat{\boldsymbol{\theta}}$

and a standard approach to solving the max-margin QP is the cutting-plane algorithm. Briefly, at each iteration the algorithm checks for the most violated constraint. If found, it will be added into the constraint set. The algorithm terminates when no more violated constraints are found (see Algorithm 2). Since the loss function is subtracted from the energy function during loss-augmented inference, the supermodular loss becomes a submodular objective and therefore admits tractable minimization.

5. Experiments

We test our algorithm on a Ubuntu 16.04 LTS server, with 4 2.4GHz Intel Xeon processors and 128GB memory. The Maxflow v3.04 library developed by Kolmogorov and Rother [16] is used for implementing the graph-cut algorithm. The cutting-plane algorithm were performed using Tsochantaridis et al. [23]'s SVM^{light} C++ library. We use Matlab R2016b's quadratic programming engine for solving our learning algorithm's objective problem. Regular SVM training and testing is implemented by scikit-learn's svm module.

5.1. Dataset preparation and configuration

Our use CSI 300's daily market price (open, high, low, close, volume), from 1st Jan 2015 to 31st Dec 2016, as our fundamental dataset. There are $246 + 245 = 491$ samples in total. We then calculated our technical analysis indicators (MACD-5, MACD-10, KDJ-9) basing on this dataset. We further processed our dataset with studentized residual normalization. Finally those samples were split into two collections: 394 samples for training set and 97 samples for testing purpose. Our data are collected from Wind dataset. During training stage, we used leave-one-out 10-fold cross-validation to avoid over-fitting. The penalization parameter C in equation (12) was set to 0.001.

MODEL	TRAIN ACCURACY	TEST ACCURACY
SSVM	74.7%	63.1%
SVM	46.8%	22.6%

TABLE 1: Results comparison between SSVM and SVM.

5.2. Comparsion with SVM

From table 1, we can conclude that the train accuracy and test accuracy of SSVM is better than that of SVM, which proves that our model is better than the traditional method.

5.3. Real-world application

According to the result of cluster analysis, the top 10 companies which have the closest cooperation with other companies among the CSI 300 are China Life Insurance Co., Sanan Optoelectronics, Sinopec Group, Huadong Medicine Co., Shenzhen Inovance Technology Co., Liaoning Cheng Da Co., Yongtai energy Limited by Share, Wanxiang Qianchao Co., China Communications Construction Co., and Bohai Financial Investment Holding Co.. All of these companies are the industry leaders and they even have a great number of hybrid businesses, especially the top 3 of them: Sinopec Group, China Life Insurance Co. and Liaoning Cheng Da Co..

As the biggest chemical manufacturer in China and the No. 2 refiner in the world, the Sinopec Group has a series of comparative advantages: its refining capacity accounts for half of the Chinas oil market share, and the distribution of its refineries is close to the consumer market, resulting in its strategic position. The complete pluralization and integration and the balanced development of upriver and downstream lead to its ability to fight fluctuation.

The controlling shareholder of China Life Insurance Co. is an exclusive state-funded corporation. The strong background of state-owned shareholder offers abundant of customer resource and great financial investment platform. Recently, the China Life Insurance Co. have reached strategic partnerships with many local governments and famous enterprises. When the industry enters differentiation phase, the position of industry leader shows up. In 2016, Liaoning Cheng Da Co. properly handled the vaccine safety problems, changed crises to opportunity and built out a new distribution channel accessing to the countys Centers for Disease Control through the cooperation with domestic large-scale circulation enterprises, maintaining the leading superiority in the business. Along with the reform of state-owned enterprises, the company successfully transformed into consolidation model.

References

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123:155–225, 2002.

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 26:1124–1137, 2004.

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *Proc. of the International Conference on Computer Vision (ICCV)*, 1999.

[5] Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *Proc. of the International Conference on Computer Vision (ICCV)*, 2001.

[6] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292, 2002.

[7] K. M. Cremers and A. Petajisto. How active is your fund manager? a new measure that predicts performance. *Review of Financial Studies*, 22(9):3329–3365, 2009.

[8] M. Fasanghari and G. A. Montazer. Design and implementation of fuzzy expert system for tehran stock exchange portfolio recommendation. *Expert Systems with Applications*, 37(9):6138–6147, 2010.

[9] D. Freedman and P. Drineas. Energy minimization via graph cuts: Settling what is possible. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[10] S. Gould. Learning weighted lower linear envelope potentials in binary markov random fields. *IEEE transactions on pattern analysis and machine intelligence*, 37(7):1336–1346, 2015.

[11] P. L. Hammer. Some network flow problems solved with psuedo-boolean programming. *Operations Research*, 13:388–399, 1965.

[12] C.-M. Hsu. A hybrid procedure for stock price prediction by integrating self-organizing map and genetic programming. *Expert Systems with Applications*, 38 (11):14026–14036, 2011.

[13] H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 25:1333–1336, 2003.

[14] H. Ishikawa. Higher-order clique reduction in binary graph cut. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[15] P. Kohli, M. P. Kumar, and P. H. S. Torr. P3 & beyond: Solving energies with higher order cliques. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[16] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts—A review. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 2007.

[17] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 26: 65–81, 2004.

[18] C. K.-S. Leung, A. Cuzzocrea, and F. Jiang. Discovering frequent patterns from uncertain data streams with time-fading and landmark models. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems VIII*, pages 174–196. Springer, 2013.

[19] R. K. MacKinnon and C. K. Leung. Stock price prediction in undirected graphs using a structural support vector machine. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2015 IEEE/WIC/ACM International Conference on*, volume 1, pages 548–555. IEEE, 2015.

[20] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extraction using iterated graph cuts. In *Proc. of the Intl. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2004.

[21] C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[22] M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph-cuts. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2008.

[23] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. In *Journal of Machine Learning Research*, pages 1453–1484, 2005.