Chapter 1

Related Work and Background

1.1 MRFs and Energy Function

1.1.1 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:

$$p(\boldsymbol{y}) = \frac{1}{Z}\prod_C \Psi_C(\boldsymbol{y}_C) \tag{1.1}$$

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:

$$p(\boldsymbol{y}) = \frac{1}{Z}exp(-\sum_C E_C(\boldsymbol{y}_C)) \tag{1.2}$$

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*:

$$\boldsymbol{y}^* = \underset{\boldsymbol{y}}{\text{argmax}}\, p(\boldsymbol{y}) = \underset{\boldsymbol{y}}{\text{argmin}}(-\sum_C E_C(\boldsymbol{y}_C)) \tag{1.3}$$

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 $\mathcal{C}$ [18] 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:

$$E(\boldsymbol{y}; \boldsymbol{w}) = \sum_{i \in \mathcal{N}} \boldsymbol{w}_i^U \phi^U(\boldsymbol{y}_i) + \sum_{(i,j) \in \mathcal{E}} \boldsymbol{w}_{ij}^P \phi^P(\boldsymbol{y}_i, \boldsymbol{y}_j) + \sum_{\boldsymbol{y}_C \in \mathcal{C}} \boldsymbol{w}_C^H \phi^H(\boldsymbol{y}_C) \tag{1.4}$$

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:

$$E(y_t, w) \leq E(y, w) \, , \forall y \neq y_t, y \in \mathcal{Y} \tag{1.5}$$

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 [18]:

$$\boldsymbol{w}^* = argmax_{\boldsymbol{w}}(E(y_t, w) - E(y, w)) \, , \forall y \neq y_t, y \in \mathcal{Y} \tag{1.6}$$

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 [16] 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. [13] to encode pairwise consistency. The *inference* problem is solved by using a graph-cut [3, 4] algorithm and the max margin framework [19] has been addressed to learn parameters of the energy function.

1.1.2 Construction of stocks relationship graph

1.1.3 Configuration of Energy Function

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

$$E(\boldsymbol{y}; \boldsymbol{w}) = \theta^U \sum_{i \in \mathcal{N}} \boldsymbol{w}_i^U \phi^U(\boldsymbol{y}_i) + \theta^P \sum_{(i,j) \in \mathcal{E}} \boldsymbol{w}_{ij}^P \phi^P(\boldsymbol{y}_i, \boldsymbol{y}_j) \tag{1.7}$$

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

The *GrabCut* algorithm was proposed by Rother et al. [16] in order to solve background foreground segmentation problem (see figure 1.1). They first defined MRFs

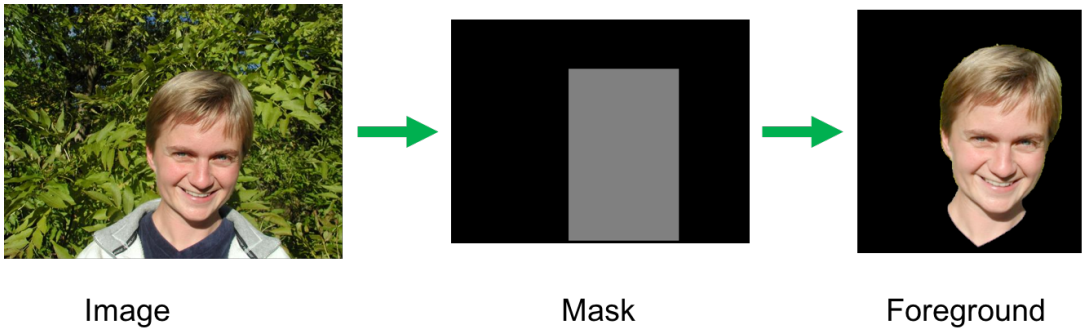

Figure 1.1: Picture on the left is the original picture. Picture on the middle is a user defined mask. The task is to extract foreground pixels within that rectangle. On the right is the ground truth foreground.

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

over an labeled image and then use *graph-cuts* [5] method to do the inference. In this section 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 pixels in an image. In order to construct MRFs, they first defined an energy function (1.9) with unary and pairwise terms:

$$G(\alpha, \boldsymbol{k}, \boldsymbol{\theta}, \boldsymbol{z}) = \sum_{i \in \mathcal{N}} \psi^U(\alpha_i, \boldsymbol{k}_i, \boldsymbol{\theta}, \boldsymbol{z}_i) + \sum_{(i,j) \in \mathcal{E}} \psi^P(\alpha_i, \boldsymbol{z}_i) \tag{1.8}$$

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 and will be explained in the next paragraph.

In our configuration, we use the graph described in 1.1.2 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. 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* [16].

The pairwise function ψ^P in (1.8) 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. This energy function was later used to construct an *st min-cut* graph which can be inferred efficiently using *graph-cuts* [5] algorithm. This gives some insights to their second contribution.

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 thesis we use GMMs trained by GrabCut algorithm for our unary terms ϕ^U in equation (1.7).

The pairwise function ϕ^P in our energy function (1.7) is defined as a Potts model:

$$\phi^P(\boldsymbol{y}_i, \boldsymbol{y}_j) = \begin{cases} 0 & \text{if} x_i = x_j \\ \psi^P_{ij}(y_i, y_j) = \frac{\lambda}{d_{ij}} [\![y_i \neq y_j]\!] \exp\left\{ -\frac{\|x_i - x_j\|^2}{2\beta} \right\} & \text{otherwise} \end{cases} \tag{1.9}$$

where d_{ij} is the graph distance between stocks i and j. x_i and x_j are stock market price vectors.

Chapter 2

Methodology

2.1 Lower Linear Envelope MRFs

We begin with extending standard Markov Random Fields (see equation (1.9)) to include the lower linear envelope potential. We then show how to perform exact inference in models with these potentials. In 2.2 we will discuss learning the parameters of the models. Major work in this section is done by Gould [8].

2.1.1 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 [10, 11, 17] are able to be minimized exactly using *graph-cuts* like algorithms [6, 9] when they satisfy submodularity condition [2]. Kohli et al. [14] and Gould [7] adapted those results to perform exact inference on lower linear envelope potentials. In this section we mainly focus on describing the *st min cut* graph constructed by Gould [7, 8] for exact inference (**??**) of energy function containing lower linear envelope potentials.

Following the approach of Kohli and Kumar [12], Gould [7, 8] transformed the weighted lower linear envelope potential (**??**) into a quadratic pseudo-Boolean function by introducing $K-1$ auxiliary variables $\boldsymbol{z} = (z_1, \ldots, z_{K-1})$ with $z_k \in \{0, 1\}$:

$$E^c(\boldsymbol{y}_c, \boldsymbol{z}) = a_1 W_c(\boldsymbol{y}_c) + b_1 + \sum_{k=1}^{K-1} z_k \left((a_{k+1} - a_k) W_c(\boldsymbol{y}_c) + b_{k+1} - b_k \right) \tag{2.1}$$

for a single clique $c \in \mathcal{C}$. Under this formulation, Gould [7, 8] showed that minimizing the pseudo-Boolean function over $\boldsymbol{z}$ is equivalent to selecting (one of) the active functions(s) from equation (**??**). Another important property of optimized $\boldsymbol{z}$ under this formulation is that it automatically satisfies the constraint [8]:

$$z_{k+1} \leq z_k \tag{2.2}$$

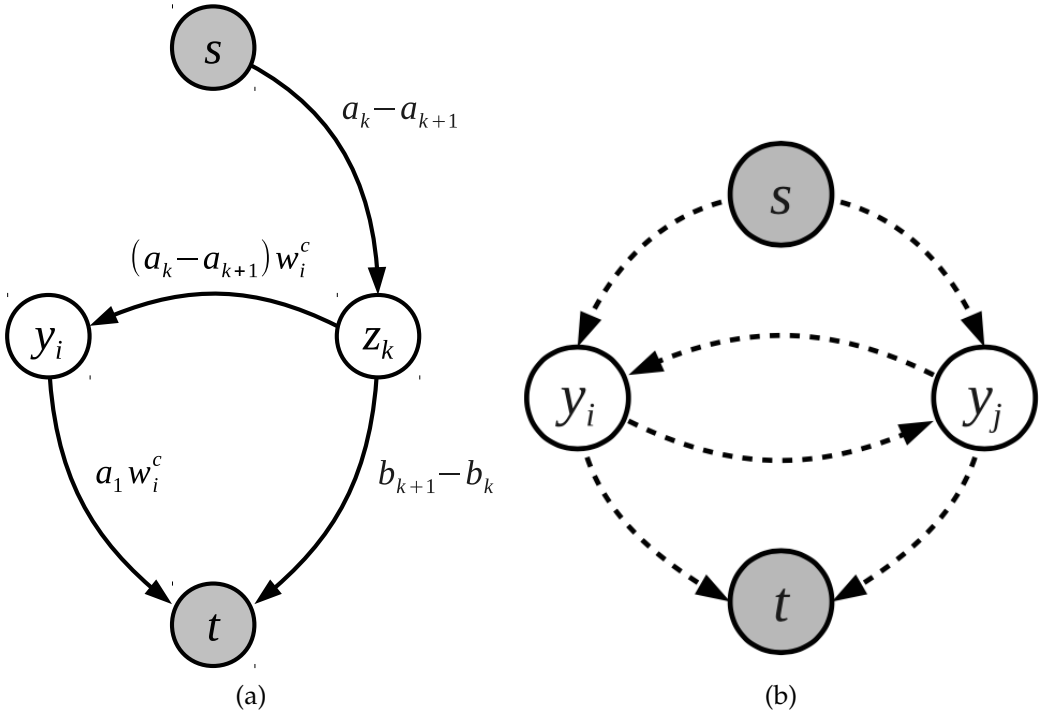

Figure 2.1: *st*-graph construction [8] for equation (2.3), unary and pairwise terms. Every cut corresponds to an assignment to the random variables, where variables associated with nodes in the $\mathcal{S}$ set take the value one, and those associated with nodes in the $\mathcal{T}$ set take the value zero. With slight abuse of notation, we use the variables to denote nodes in our graph.

this property give rise to further development of parameter vector (**??**) and feature vector (**??**) which are used in latent structural SVM.

In order to construct the *st-min-cut* graph, Gould [8] rewrote equation (2.1) into *posiform* [2]:

$$E^c(\boldsymbol{y}_c, \boldsymbol{z}) = b_1 - (a_1 - a_K) + \sum_{i \in c} a_1 w_i^c y_i + \sum_{k=1}^{K-1} (b_{k+1} - b_k)\, z_k \qquad (2.3)$$
$$+ \sum_{k=1}^{K-1} (a_k - a_{k+1})\, \bar{z}_k + \sum_{k=1}^{K-1} \sum_{i \in c} (a_k - a_{k+1})\, w_i^c \bar{y}_i z_k$$

where $\bar{z}_k = 1 - z_k$ and $\bar{y}_i = 1 - y_i$. a_1 is assumed to be greater than 0 so that all coefficients are positive (recall we assume $b_1 = 0$ in section **??** and we have $a_k > a_{k+1}$ and $b_k < b_{k+1}$). After proving *submodularity* of the energy function (2.3), Gould [8] constructed the *st-min-cut* graph based on equation (2.3).

The construction is explained in Figure 2.1. Figure (a) denotes construction for equation (2.3). For each lower linear envelope potential edges are added as follows: for each $i \in c$, add an edge from y_i to t with weight $a_1 w_i^c$; for each $i \in c$ and $k = 1, \ldots, K-1$, add an edge from z_k to y_i with weight $(a_k - a_{k+1}) w_i^c$; and for

$k = 1, \ldots, K-1$, add an edge from s to z_k with weight $a_k - a_{k+1}$ and edge from z_k to t with weight $b_{k+1} - b_k$. Figure (b) denotes construction for unary and pairwise terms (see [15]). 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.

2.2 Learning the Lower Linear Envelope with Latent Information

With the inference algorithm in hand, we now can develop the learning algorithm for weighted lower linear envelope potentials using the latent structural SVM framework. We begin by transforming the equation (2.1) into a linear combination of parameter vector and feature vector. Then a two-step algorithm was developed to solve the latent structural SVM.

Bibliography

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. 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.
7. S. Gould. Max-margin learning for lower linear envelope potentials in binary Markov random fields. In *Proc. of the International Conference on Machine Learning (ICML)*, 2011.
8. 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.
9. P. L. Hammer. Some network flow problems solved with psuedo-boolean programming. *Operations Research*, 13:388–399, 1965.
10. H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 25:1333–1336, 2003.
11. H. Ishikawa. Higher-order clique reduction in binary graph cut. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
12. P. Kohli and M. P. Kumar. Energy minimization for linear envelope MRFs. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
13. 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.
14. P. Kohli, L. Ladicky, and P. H. S. Torr. Graph cuts for minimizing higher order potentials. Technical report, Microsoft Research, 2008.
15. 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.
16. C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extrac-

tion using iterated graph cuts. In *Proc. of the Intl. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2004.

17. 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.
18. M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph-cuts. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2008.
19. 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.