Minimisation du risque empirique avec des fonctions de perte nonmodulaires

Jiaqian Yu

To cite this version:

Jiaqian Yu. Minimisation du risque empirique avec des fonctions de perte nonmodulaires. Autre. Université Paris Saclay (COmUE), 2017. Français. NNT : 2017SACLC012 . tel-01514162

HAL Id: tel-01514162
https://theses.hal.science/tel-01514162

Submitted on 25 Apr 2017

NNT : 2017SACLC012

THÉSE DE DOCTORAT DE L'UNIVERSITÉ PARIS-SACLAY PRÉPARÉ À CENTRALESUPÉLEC

Ecole doctorale n°580
Sciences et Technologies de l'Information et de la Communication
Spécialité de doctorat : Traitement du Signal et des Images

par

JIAQIAN YU

Empirical risk minimization with non-modular loss functions

Thèse présentée et soutenue à Châtenay-Malabry, le 22 mars 2017.

Composition du Jury :

M.	FRANCIS BACH	Directeur de recherche INRIA	(Président du Jury)
M.	NIKOS KOMODAKIS	Associate professor Ecole des Ponts ParisTech	(Rapporteur)
M.	STEFAN ROTH	Professor Technische Universität Darmstadt	(Rapporteur)
Mme.	FLORENCE D'ALCHÉ-BUC	Professor Télécom ParisTech	(Examinateur)
M.	NIKOS PARAGIOS	Professor Université Paris-Saclay & Inria	(Examinateur)
M.	MATTHEW B. BLASCHKO	Professor KU Leuven	(Directeur de thèse)

Abstract

This thesis addresses the problem of learning with non-modular losses. In a prediction problem where multiple outputs are predicted simultaneously, viewing the outcome as a joint set prediction is essential so as to better incorporate real-world circumstances. In empirical risk minimization, we aim at minimizing an empirical sum over losses incurred on the finite sample with some loss function that penalizes on the prediction given the ground truth. In this thesis, we propose tractable and efficient methods for dealing with non-modular loss functions with correctness and scalability validated by empirical results.

First, we analyze the feasibility of using a structured output support vector machine (SVM) with margin rescaling and a supermodular loss function. We present the hardness of incorporating supermodular loss functions into the inference term when they have different graphical structures. We then introduce an alternating direction method of multipliers (ADMM) based decomposition method for loss augmented inference, that only depends on two individual solvers for the loss function term and for the inference term as two independent subproblems. In this way, we gain computational efficiency and achieve more flexibility in choosing our loss function of interest. We show that the novel supermodular loss function can empirically achieve better performance on an image segmentation task.

Second, we show the necessity of using submodular loss functions in structured prediction problems. A tight and computationally efficient surrogate function for learning with submodular functions has not been previously developed. We show that margin rescaling and slack rescaling in a structured output SVM lead to tight convex surrogates, if and only if the underlying loss function is increasing in the number of incorrect predictions. However, the gradient or cutting-plane computation for these functions is NP-hard for non-supermodular loss functions. We propose instead a novel surrogate loss function for submodular losses, the Lovász hinge, which leads to $\mathcal{O}(p \log p)$ complexity with $\mathcal{O}(p)$ oracle accesses to the loss function to compute a gradient or cutting-plane. We validate the correctness of the Lovász hinge on various prediction tasks including multilabel prediction tasks on the Pascal VOC and the MS COCO datasets. We show that for submodular loss functions, training with the Lovász hinge achieves lower empirical error value than margin rescaling and slack rescaling, which is expected from a correctly defined convex surrogate.

Finally, based on the previous contributions, we are able to introduce a novel convex surrogate operator for general non-modular loss functions, which provides for the first time a tractable solution for loss functions that are neither supermodular nor submodular. This surrogate is based on a canonical submodular-supermodular decomposition. It takes the sum of two convex surrogates that separately bound the supermodular component and the submodular component using slack-rescaling and the Lovász hinge, respectively. It is further proven that this surrogate is convex, piecewise linear, an extension of the loss function, and for which subgradient computation is polynomial time. We further show that the Dice loss which is defined based on the Sørensen-Dice difference function is neither supermodular nor submodular. Empirical results are reported using the Sørensen-Dice loss and a set of non-submodular loss functions demonstrating the improved performance, efficiency, and scalability of the novel convex surrogate.

Contents

1 Introduction **5**

1.1 Context and Motivation . . . 5

1.1.1 Empirical Risk Minimization . . . 5

1.1.2 Submodularity . . . 6

1.1.3 Loss Functions . . . 7

1.2 Contributions and Thesis Outline . . . 9

2 Mathematical Foundations **11**

2.1 Submodular Functions . . . 11

2.1.1 Definition of A Submodular Function . . . 12

2.1.2 Submodular Function Analysis . . . 15

2.2 Maximum A Posteriori Inference and Submodularity . . . 19

2.2.1 Markov Random Field . . . 19

2.2.2 Maximum A Posteriori . . . 21

2.2.3 Graph Cut and Submodular constraints . . . 21

2.3 Empirical Risk Minimization . . . 22

2.3.1 Empirical Risk . . . 22

2.3.2 Surrogate Functions . . . 23

2.4 Structured Output SVM . . . 24

3 Efficient Learning with Supermodular Loss Functions **27**

3.1 Related Work . . . 28

3.2 Loss Augmented Inference (LAI) . . . 30

3.3 A Supermodular Loss Function for Binary Segmentation . . . 32

3.3.1 A novel supermodular loss function . . . 32

3.3.2 Non-submodularity of the LAI . . . 33

3.4 ADMM Algorithm for LAI . . . 34

3.5 Experimental Results . . . 36

3.6 Discussion . . . 40

4 Lovász Hinge: Learning with Submodular Loss Functions 51
4.1 Introduction . . . 52
4.2 Submodular Loss Functions . . . 53
4.3 Existing Convex Surrogates . . . 54
4.3.1 Extension . . . 56
4.3.2 Slack Rescaling . . . 57
4.3.3 Margin Rescaling . . . 58
4.3.4 Complexity of Subgradient Computation . . . 59
4.4 Lovász Hinge . . . 60
4.4.1 Lovász Hinge . . . 60
4.4.2 Complexity of Subgradient omputation . . . 63
4.4.3 Visualization of Convex Surrogates . . . 63
4.5 Hardness Result for Learning Multiple Correct Results . . . 69
4.5.1 Learning . . . 70
4.5.2 Supermodular losses and the Structured Output SVM . . . 70
4.5.3 Submodular losses and the Lovász Hinge . . . 71
4.5.4 The Inference Problem . . . 72
4.6 Jaccard Loss . . . 72
4.7 Experimental Results . . . 74
4.7.1 A Synthetic Problem . . . 74
4.7.2 PASCAL VOC Image Classification . . . 75
4.7.3 PASCAL VOC Multilabel Prediction . . . 77
4.7.4 MS COCO Multilabel Prediction . . . 78
4.7.5 Empirical Results . . . 80
4.8 Discussion . . . 80

5 Convex Surrogate Operator for General Non-modular Loss Functions 83
5.1 Introduction . . . 84
5.2 A Submodular-Supermodular Decomposition . . . 84
5.3 A Canonical Decomposition . . . 86
5.4 A Convex Surrogate Operator $\mathbf{B}_\mathbf{D}$ . . . 87
5.4.1 Definition of $\mathbf{B}_\mathbf{D}$ . . . 87
5.4.2 Properties of $\mathbf{B}_\mathbf{D}$ . . . 88
5.5 Sørensen-Dice loss . . . 91
5.6 Experimental Results . . . 93
5.6.1 Dice Loss . . . 93
5.6.2 Face Classification in Video Sequences . . . 93
5.7 Discussion . . . 98

6 Conclusion **101**
6.1 Contributions 101
6.2 Publication list 104

A Résumé **107**

Bibliography **108**

List of Figures

1.1 Examples of structured prediction problem in nature language processing and images segmentation problem. 6

1.2 Examples of training samples for an image segmentation problem where the foreground objects are *people*. Images are taken from the Pascal VOC dataset [Everingham et al., 2010] 7

1.3 Set functions in general. The intersection of submodular functions and supermodular functions, is the set of modular functions. In general, there are many functions that are neither submodular nor supermodular. 8

1.4 The structure of this thesis in a Venn diagram related to the submodularity of the loss functions. 9

2.1 An example on multi-label prediction problem as a set prediction problem. The groundtruth of the problem can be formulated as a base set that contains all labels of interest, while each pattern can be seen as a subset of this base set. Images are taken from Pascal VOC dataset [Everingham et al., 2010] . . 13

2.2 Submodular polyhedron $P(F)$ and base polyhedron $B(F)$ for $p = 2$ (left) and $p = 3$ (right), for a non-decreasing submodular function. Illustration from Bach [2013]. 16

2.3 The surface of the Lovász extension for a submodular function in case $p = 2$: $f(\{\emptyset\}) = 0$, $f(\{1\}) = 0.8$, $f(\{2\}) = 0.6$, $f(\{1, 2\}) = 0.4$. The red dot represent the values of the set function f on the vertex of the unitcube. . . . 17

2.4 An example shows that the maximum of two concave functions is not concave in general, thus the maximum of two submodular functions is not submodular in general. 19

2.5 Examples of different graph structures. Figure 2.5(a) shows a 4-connected neighborhood graph and Figure 2.5(b) shows an 8-connected neighborhood graph. 20

2.6 An illustration of MRF with observed variable x and unobserved variable y in a 4-connected graph structure. Illustration from Prince [2012]. 21

2.7 An illustration of graph-cut (max-flow min-cut) problem with node *Source* and *Sink*. Illustration from Prince [2012]. 22

2.8 The loss function is considered as a set function, with the input set equals to the misprection set, which is a subset of the base set V (Equation (2.3.1)). . 23

2.9 Plots of the 0-1 loss and some common used surrogate functions including: hinge loss, squared hinge loss, logistic regression loss, and exponential loss. The functions are plotted as a function of the margin $yh(x)$. 24

3.1 Non-submodularity of the joint loss augmented inference procedure using the same mapping to a set function for inference and loss functions. The inference procedure can be solved by graph cuts when the sum of the diagonal elements of E is less than the sum of the off diagonal elements. While it is enforced during optimization that $w_{00} + w_{11} - w_{01} - w_{10} \geq 0$, the presence of γ in the off diagonal, the exact position depending on the value of y, removes the guarantee of a resulting submodular minimization problem. 33

3.2 Example images from the dataset and the extracted features. 3.2(a) original RGB space image; 3.2(b) groundtruth; 3.2(c) the user-labelled seeds; 3.2(d) the extended seed region; 3.2(e) the distance features to foreground seed based on RGB space; 3.2(f) the distance features to background seed based on RGB space; 3.2(g) the GMM appearance model based on RGB space; 3.2(h) the distance features to foreground seed based on the RGB-space GMM appearance. 35

3.3 The computing time for the loss augmented inference, on different problem sizes. The red histograms stands for ADMM and the blue for MinNorm. As we can see, the calculation by ADMM is always faster than by MinNorm, and there is no overlap between the computing time by the two methods. 42

3.4 The running time increase along with the problem size. Both algorithm increase linearly in log scale while the ADMM has a time reduction from 10 times to 10^2 times along with the increase of the problem size. 43

3.5 The segmentation results of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs. 44

3.6 A pixelwise comparison, in the semantic segmentation task [Gulshan et al., 2010], of the ground truth (denoted g in the legend), the prediction from training with Hamming loss (denoted h), and the prediction when training with the proposed supermodular loss (denoted s). We note that there are many regions in the set of images where the supermodular loss learns to correctly predict the foreground when Hamming loss fails (orange regions corresponding to $g = +1$, $h = -1$, and $s = +1$). 45

3.7 A pixelwise comparison, in the structural brain segmentation task [Rohlfing, 2012], of the ground truth (denoted g in the legend), the prediction from training with Hamming loss (denoted h), and the prediction when training with the proposed supermodular loss (denoted s). We note that there are many regions in the set of images where the supermodular loss learns to correctly predict the foreground when Hamming loss fails (orange regions corresponding to $g = +1$, $h = -1$, and $s = +1$). 46
3.8 The segmentation results (continued) of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs. 47
3.9 The segmentation results (continued) of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs. 48
3.10 The segmentation results (continued) of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs. 49

4.1 Figure 4.1(a) shows the hinge loss as a function of $g^j(x)y^j$ and Figure 4.1(b) shows the transformed mapping as a function of $1 - g^j(x)y^j$ 56
4.2 The Lovász hinge surfaces with a submodular increasing loss and a submodular non-increasing loss from different views; the x and y axes represent the value of s_i^1 and s_i^2 in Equation (4.4.2); the z axis represents the value of the convex surrogate; the solid red dots represent the values of ℓ at the vertices of the unit hypercube. The convex surrogate strategies yield extensions of the discrete loss as the red dots touch the surfaces. 65
4.3 Margin rescaling surfaces with a submodular increasing loss and a supermodular increasing loss from different views. The convex surrogate strategies yield extensions of the discrete loss as the red dots touch the surfaces. . . . 66
4.4 Slack rescaling surfaces with a submodular increasing loss and a supermodular increasing loss from different views. The convex surrogate strategies yield extensions of the discrete loss as the red dots touch the surfaces. . . . 67
4.5 Magin rescaling surfaces and slack rescaling surfaces with a submodular non-increasing loss from different views. The convex surrogate strategies fail to yield extensions of the discrete loss as the red dots are below the surfaces. . 68

4.6 Lovász hinge with submodular non-increasing ℓ while thresholding negative components of s is still applied (cf. the caption of Figure 4.2 for the axes notation). Although the red dots touch the surface, the surface is no longer convex due to the thresholding strategy. 69
4.7 The disributions of samples for a synthetic binary classification problem motivated by the problem of early detection in a temporal sequence. As in, e.g. disease evolution, the distribution of early stage samples differs from that of late stage samples. 75
4.8 Example images from the PASCAL VOC dataset. Figure 4.8(a) and Figure 4.8(b) contain three categories: *people*, *table* and *chair*. Figure 4.8(c) contains *table* and *chair*. 77
4.9 Examples from the Microsoft COCO dataset. Figure 4.9(a) contains all the categories of interest (cf. Section 4.7); Figure 4.9(b) contains *dining table*, *fork* and *cup*; Figure 4.9(c) is not a dining scene but contains *people*. 79
4.10 The primal-dual gap as a function of the number of cutting-plane iterations using the Lovász hinge with submodular loss, a SVM (labeled 0-1), and margin and slack rescaling with greedy inference (labeled $\mathbf{M}$ and $\mathbf{S}$). Figure 4.10(a) for the experiment using Equation (4.7.5) and Figure 4.10(b) for Equation (4.7.6). This demonstrates that empirical convergence of the Lovász hinge is at a rate comparable to an SVM, and is feasible to optimize in practice for real-world problems. 82

5.1 Plots of Equation (5.5.4) (red) and Equation (5.5.5) (blue) as a function of n_A. As these two plots cross, neither function bounds the other. 92
5.2 The data for the synthetic problem. The negative samples are drawn from a mixture of Gaussians. 93
5.3 Examples of the face track images. Figure 5.3(a) shows the “Buffy” role thus a positive-labelled image and Figure 5.3(b) shows a negative-labelled image. An automated pipeline described in Everingham et al. [2006, 2009]; Sivic et al. [2009] was used for feature extraction (Figure 5.3(c)). 94
5.4 The plot of the four loss functions used in our experiments as in Equations (5.6.1) to (5.6.4). The x axis is the number of mispredictions for each track (we show here the loss functions corresponding to track length equal to 10 as an example), and the y axis is the value of loss function. The original losses are drawn in red; the supermodular components are drawn in green, and the submodular components in blue. 95

5.5 The primal-dual gap as a function of the number of cutting-plane iterations using different convex surrogates for the four non-modular functions in Equations (5.6.1) to (5.6.4). The primal-dual gap from $\mathbf{B_D}$ is drawn in red; the gap from $\mathbf{S}$ is drawn in green and gap from Hamming loss (labelled 0-1, and equivalent to a SVM) in blue. Our convex surrogate operator $\mathbf{B_D}$ can achieve a comparable convergence rate to an SVM, demonstrating that optimization is very fast in practice and the method scales well to large datasets. 97

6.1 The structure of the contribution of this thesis in a Venn diagram related to the submodularity of the loss functions. 103

List of Tables

2.1 Table of notation in this section. 12
2.2 A summary of submodular minimization methods. 18

3.1 The cross comparison of average loss values (with standard error) using different loss functions during training and during testing on Grabcut dataset . Training with the same supermodular loss functions as used during testing yields the best results. Training with supermodular losses even outperform the Hamming loss in terms of evaluating by Hamming loss. A Wilcoxon sign rank test shows that training with Δ gives significantly better results in all cases ($p \leq 2 \times 10^{-3}$). 36
3.2 The cross comparison of average loss values (with standard error) using different loss functions during training and during testing on IBSR dataset. (cf. comments for Table 3.1). 36
3.3 The comparison between ADMM and an LP relaxation for solving the loss augmented inference. The 1st column shows the optimal energy values (10^3) (Equation (3.2.4)); columns 2–4 show the computation time (s) for one calculation on downsampled images of varying size. 39
3.4 The cross comparison of average loss values (with standard error) using the supermodular loss and Hamming loss (labeled 0-1) during training and test time). 39

4.1 For the synthetic problem, the cross comparison of average loss values (with standard error) using different convex surrogates for the submodular loss in Equation (4.7.1) during training, 0-1 loss for comparison, and testing with different losses. 76
4.2 For the VOC image classification task, the cross comparison of average loss values using the Jaccard loss as well as 0-1 loss during training and during testing for the 20 categories. 76
4.3 For the VOC multilabel prediction task, the cross comparison of average loss values (with standard error) using submodular loss as in Equation (4.7.3) and Equation (4.7.4), as well as 0-1 loss during training and during testing. 78

4.4 For the MS COCO prediction task, the comparison of average loss values (with standard error) using the submodular losses as in Equation (4.7.5), Equation (4.7.6) and Equation (4.7.7), as well as 0-1 loss during training and during testing. 81

5.1 For the synthetic data experiment, the cross comparison of average loss values (with standard error) using different surrogate operations during training, and different evaluation functions during test time. Δ_D is the Dice loss as in Equation (5.5.1). 94

5.2 For the face classification task, the cross comparison of average loss values (with standard error) using different surrogate operator and losses as in Equation (5.6.1) to Equation (5.6.4) during training, respectively. For the cases that the submodular component is non-negative, i.e. using Δ_1 and Δ_2, the lowest empirical error is achieved when using $\mathbf{B_D}$. 94

5.3 The comparison of the computation time (s) for one loss augmented inference. 98

6.1 A summary of loss functions that appeared in this thesis, on given a groundtruth y and a predicted output $\tilde{y}$, α, β, γ and l_{max} are real-valued parameters, F is the set of frequent itemsets. 102

Acknowledgements

First and foremost, I would like to thank my supervisor Prof. Matthew B.Blaschko. This thesis would not be possible without him. His insights, wide and deep knowledge, passion on science have always been a source of inspiration to me. I appreciate every discussion, meeting, conference call we had that always helped me overcome difficulties, gave me motivation to keep working on challenging problems. I feel so lucky that I had a chance to work with Matthew, with who we can discuss math problems with a sandwich in hand, who shared his ideas and experience on academic work, who gave me down-to-earth programming advice and correct my English. His patient advising and encouragement companied me throughout the journey of my PhD.

I would like to thank the committee members. Many thanks to Nikos Komodakis and Stefan Roth for reviewing my manuscript. Stefan Roth gave detailed comments on my manuscript that helped me improve the revision of the manuscript. I appreciate also all the comments and insightful suggestions from Florence d'Alché-Buc, Francis Bach and Nikos Paragios on my manuscript and on the future work. It is a great honor to have them in my committee.

I could not wish for a better environment to do my PhD than the Center for Visual Computing. I would like to thank our lab director Prof. Nikos Paragios who supported me to participe in conferences, to apply for different programs and awards. I would like to thank Natalia for helping with all the administrative procedures during this years. Many thanks to Eva for providing helpful suggestions on doing research and a good presentation. I've also enjoyed working with Eva for teaching the Machine Learning course. I would also like to thank all the labmates. I appreciate every discussion we had during the lab meetings, the seminars, every happy lunch time and coffee time. All these happy moments are my most valuable memories. Special thanks to Wacha for her kindness, her encouragement, for being my French teacher during this years and in additionally helping me with the French abstract of this manuscript, to Eugene for every insightful discussion and for helping with checking the language of this manuscript, to Siddartha for replying patiently my questions regarding using the cluster, to Mihir, Hari, Evgenios, Marie-Caroline, Khue, Stefan, Maxim, Alp, Starvos, Puneet, Enzo, Jieying for sharing together the happy moments in the lab and for their help and support on work, that whenever I felt confused and had questions, they can give greatly useful comments and feedback. They are my colleagues in the lab but also friends in life. I particularly appreciate Marie-Caroline, Mihir, Hari, Siddartha, Khue, Eugene and Eva for their helpful feedbacks on the practice talk of my defense. It has been a great pleasure to be surrounded by such lovely, smart and excellent people.

I would like to express my gratitude to the China Scholarship Council for supporting financially my PhD. I also appreciate the mobility grant from the Universié Paris-Saclay for supporting me visiting the Center for Processing Speech & Images at KU Leuven.

There are so many people to whom I want to extend my heartfelt gratitude, to my friends and colleagues in Leuven: Xuanli, Xu, Yu-hui and José for their helps there and for making my time in Leuven so colorful, to my long-time friends from China that made me always feel supported and encouraged during this PhD journey. They made me feel not lonely for being far from my country. Extra special thanks goes to Ben. This PhD journey would not have been as joyful and smooth without his companionship and encouragement.

Last but not least, I would like to give my sincere thanks to my beloved parents and grandparents, for their unconditional love and tremendous support.

To My Parents,

To Ben.

Chapter 1

Introduction

Contents

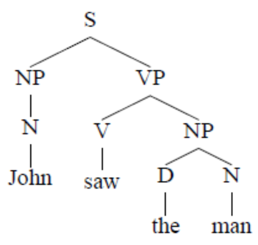
1.1 Context and Motivation **5**
1.1.1 Empirical Risk Minimization 5
1.1.2 Submodularity 6
1.1.3 Loss Functions 7
1.2 Contributions and Thesis Outline **9**

1.1 Context and Motivation

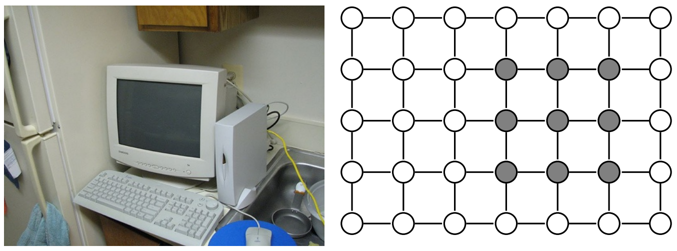
Machine learning builds on many fields including optimization and statistics. It addresses many problems including artificial intelligence, computer vision, natural language processing and so on. Structured prediction as one domain of machine learning focuses on the study of the prediction problem while modeling real-world concepts as structured objects. For example, in natural language processing, a sentence is represented as having chain structures or tree structures [Collins and Duffy, 2001, 2002]; in image parsing problem, a scene is described using visual grammars [Tu et al., 2005; Han and Zhu, 2009]; in graphical model, graphs with directed or undirected edges are used to show the conditional probability relation between connected elements, e.g. pixels in images.

1.1.1 Empirical Risk Minimization

In structured prediction problem, in particular in the scenario of supervised learning, it is often the case that a series of training samples $\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$ are given in order to learn a model, train the parameters e.g. a weight vector in the model, so as to apply this trained model to make prediction during test time given unseen features. Illustrated in Figure 1.2 is a set of training samples in a image foreground-background segmentation problem where x_i are pixelwise features and y_i are labels. In particular, in structured prediction problems, we consider that y_i is a structured objects, and the interdependencies



(a) An illustration from Collins and Duffy [2002] to show a tree structure of one sentence.



(b) Pixels in an image [Everingham et al., 2010] are represented by nodes in a graph where edges are showing for conditional probabilistic relation between linked nodes.

Figure 1.1: Examples of structured prediction problem in nature language processing and images segmentation problem.

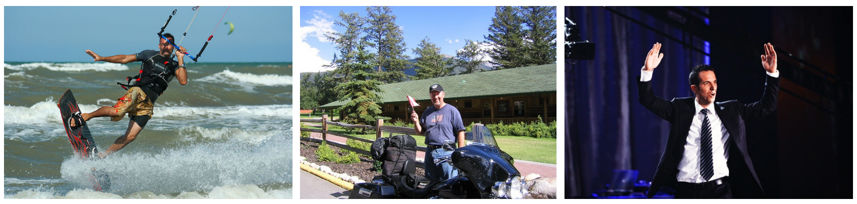
between the components of y_i is taken into account.

In empirical risk minimization, we aim at minimizing an empirical sum over losses incurred on the finite sample with some loss function that penalizes on the prediction given the groundtruth. A trivial example of the loss function is the Hamming loss, also called 0-1 loss, that counts the number of elements that are mispredicted. However, optimizing for a prediction problem with 0-1 loss, which is relatively simple, is known to be NP-hard [Feldman et al., 2012]. In practice, an approximation that upper bounding the 0-1 loss is always utilized, which is often convex in order to achieve computation feasibility and convenience. We call this approximation a *surrogate* loss function. We will see some other commonly used convex surrogate functions for binary classification problems in Chapter 2.

1.1.2 Submodularity

Submodularity is a property of a set function. A set function is a function whose input is a set and whose output is a real number. A set function is submodular when the marginal value added by including one extra element in the set is decreasing in the size of the input set. The negative of a submodular function is a supermodular function. A function is modular if it is both submodular and supermodular. Finally, a function is called *non-modular* if it is not modular. The set of non-modular functions includes the functions that are strictly submodular or strictly supermodular, but also includes the functions that are neither submodular nor supermodular. Figure 1.3 shows the Venn diagram of general functions from the perspective of submodularity. We will formally introduce the definitions and notions in Chapter 2.

Submodular functions first appeared in the context of combinatorial optimization with the concept first introduced by Whitne and Tutte for matroids, by Choquet for the capacity theory and by Ore for graphs [Tutte, 1966; Schrijver, 2002; White, 1986; Ore and Ore, 1962; Fujishige, 2005]. Submodular set functions have been frequently considered in the


(a) A set of input images

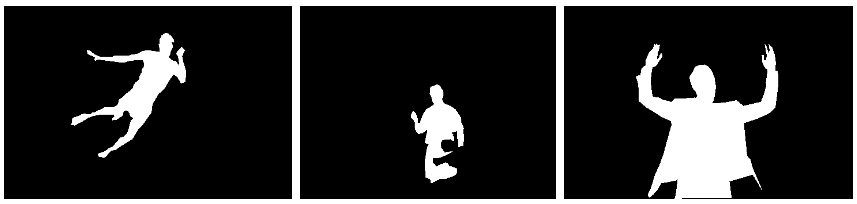

(b) A set of output segmentations

Figure 1.2: Examples of training samples for an image segmentation problem where the foreground objects are *people*. Images are taken from the Pascal VOC dataset [Everingham et al., 2010]

context of machine learning problems. The submodularity of a problem is central to the feasibility of its solution [Kolmogorov and Zabin, 2004; Kirillov et al., 2015]. The diminishing returns property of submodular functions makes them suitable for many real world applications, such as active learning, document summarization, video summarization, feature selection, sensor placement, distributed computing and many other domains [Wei et al., 2015; Lin and Bilmes, 2011; Gygli et al., 2015; Krause et al., 2008; Chen et al., 2014].

In real-world circumstances, there are two contexts in which we study submodularity:

1. the way in which we describe structured objects, namely the statistical inference problem;

2. the way in which we evaluate one structured output prediction given the groundturth output, namely the loss function.

In this thesis, we focus on the study of the loss functions that are utilized in a prediction system. We will look at loss functions of interest for different applications from the perspective of submodularity.

1.1.3 Loss Functions

Statistical learning has largely addressed problems in which a loss function decomposes over individual training samples. However, there are many circumstances in which non-decomposable, namely non-modular losses must be minimized. This can be the case for example multiple output predictions are simultaneously made and are used to make one single decision as a real-world outcome, e.g. predicting pixels so as to recognizing an object, predicting one label among multiple labels of interest so as to recognizing a scenario, etc.

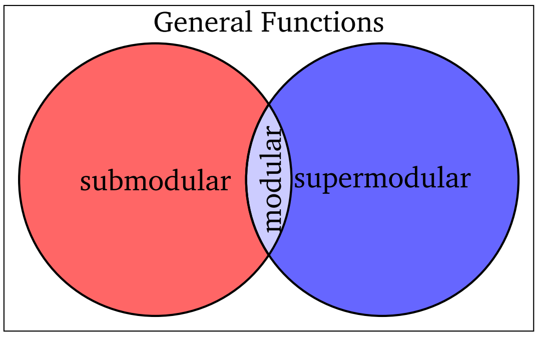


Figure 1.3: Set functions in general. The intersection of submodular functions and supermodular functions, is the set of modular functions. In general, there are many functions that are neither submodular nor supermodular.

Taking a high level view of one action of the prediction, the dependencies among the *effect* of the output predictions should therefore be better incorporated through a non-modular loss function.

If the relevant loss at test time is non-modular, it is essential to optimize the correct loss during training time [Díez et al., 2015; Gao and Zhou, 2013]. Non-modular losses have been (implicitly) considered in the context of structured prediction problems. Cheng et al. [2010] uses a rank loss which is strictly supermodular; Petterson and Caetano [2011] and Doppa et al. [2014] use a non-submodular loss based on F-score; Nowozin [2014] has considered the intersection over union loss which is strictly submodular [Yu and Blaschko, 2015a]; Pletscher and Kohli [2012] applied an area/volume based label-count loss that enforces high-order statistics which is strictly supermodular; Osokin and Kohli [2014] proposed a layout-aware loss function that takes into account the topology/structure of the object which is also strictly supermodular.

Tasks of interest In this thesis, we will mainly focus on three research questions:

1. Given a supermodular loss function of interest, is it tractable to incorporate it in to existing surrogate functions such as margin rescaling in a structured output supporter vector machine (SVM) [Taskar et al., 2004; Tsochantaridis et al., 2005]? Is the structured output SVM tractable to learn with any supermodular loss function?

2. In the case that a submodular loss function is preferred, can we develop a convex surrogate function for submodular loss functions that is extension of the discrete loss function? Is it always computationally feasible?

3. Can we develop a convex surrogate function for loss functions that are neither submodular nor supermodular?

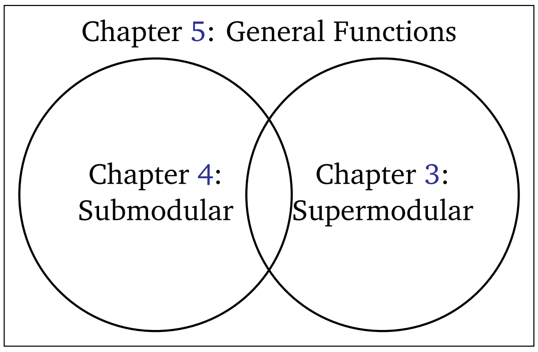


Figure 1.4: The structure of this thesis in a Venn diagram related to the submodularity of the loss functions.

1.2 Contributions and Thesis Outline

Following the motivation and previous research questions of interest, this thesis is organized as follows:

In Chapter 2, we introduce basic concepts and formal math notation from discrete optimization, combinatorial optimization, statistical learning theory and computer vision problems, including the definition of set functions, submodularity, Markov Random Fields, empirical risk minimization, structured output SVM, etc., which are necessary for the following chapters.

In Chapter 3, we analyze the feasibility of using a structured output SVM with margin rescaling and a supermodular loss function. We present the hardness of incorporating supermodular loss functions into the inference term when they have different graphical structures. We then introduce an alternating direction method of multipliers based decomposition method for loss augmented inference, which allows us to gain computational efficiency, making new choices of loss functions practical for the first time. We validate the proposed methods on an image segmentation task. We also release an open-source package online which provides an alternative API for structured output SVM. The content of this chapter is based on the following publication:

- Jiaqian Yu and Matthew B. Blaschko. Efficient learning for discriminative segmentation with supermodular losses. In Proceedings of the British Machine Vision Conference. BMVA Press, 2016.

In Chapter 4, we show the necessity of using submodular loss functions in structured prediction problems. A tight and computationally efficient surrogate function for learning with submodular functions has not been previously developed. In this chapter, we introduce the notion of *extension*. We show necessary and sufficient conditions for margin rescaling and slack rescaling in a structured output SVM to yield extension. We then propose a novel convex surrogate loss function, the Lovász hinge, which makes learning with submodular loss functions tractable (polynomial time) for the first time. We validate the correctness of the Lovász hinge on various prediction tasks by showing that training with the same loss function as during test time yields the lowest empirical risk values. The content of this chapter is based on the following publications:

- Jiaqian Yu and Matthew B. Blaschko. The Lovász hinge: A convex surrogate for submodular losses. 2015. arXiv:1512.07797;
- Matthew B. Blaschko and Jiaqian Yu. Hardness results for structured learning and inference with multiple correct outputs. In Constructive Machine Learning Workshop at ICML, Lille, France, July 2015;
- Jiaqian Yu and Matthew B. Blaschko. Learning submodular losses with the lovász hinge. In Proceedings of the 32nd International Conference on Machine Learning, volume 37, pages 1623–1631, 2015.

In Chapter 5, based on the contribution of the previous chapter, we are able to introduce a novel convex surrogate operator for general non-modular loss functions, which provides for the first time a tractable solution for loss functions that are neither supermodular nor submodular. This surrogate is based on a canonical submodular-supermodular decomposition. We validate the correctness and scalability of the proposed method on a face classification task in a video sequence. The content of this chapter is based on the following publication:

- Jiaqian Yu and Matthew B. Blaschko. A convex surrogate operator for general non-modular loss functions. International Conference on Artificial Intelligence and Statistics, volume 51 of Journal of Machine Learning Research, pages 1032–1041, 2016.

In Chapter 6 we conclude the thesis, summarize our contributions, and discuss prospective future work. A diagram of the structure of the thesis is shown in Figure 1.4.

Chapter 2

Mathematical Foundations

Contents

2.1 Submodular Functions **11**
2.1.1 Definition of A Submodular Function 12
2.1.2 Submodular Function Analysis 15
2.2 Maximum A Posteriori Inference and Submodularity **19**
2.2.1 Markov Random Field 19
2.2.2 Maximum A Posteriori 21
2.2.3 Graph Cut and Submodular constraints 21
2.3 Empirical Risk Minimization **22**
2.3.1 Empirical Risk 22
2.3.2 Surrogate Functions 23
2.4 Structured Output SVM **24**

Overview In this chapter, we introduce some basic concepts that are necessary for the remainder of this thesis. First, we introduce the notion of set functions, submodularity and properties of submodular functions. Second, we provide an overview of graph-cuts in computer vision, submoudular potential, and Markov Random Fields (i.e. undirected graphical models). Then, we present the principle of empirical risk minimization and surrogate functions. Last, we will give a brief introduction to the structured output support vector machine.

2.1 Submodular Functions

The material presented in this section is partially based on Lovász [1983]; Bach [2013]; Fujishige [2005]. Table 2.1 summarizes a list of notation introduced in this section.

Notation	Meaning
$V = \{1, 2, \dots, p\}$	the base set
$\mathcal{P}(V)$	the power set of V
$\mathbb{R}$	set of real numbers
$\emptyset$	the empty set
$\mathbb{R}_+, \mathbb{Z}_+$	the non-negative of $\mathbb{R}, \mathbb{Z}$
$f : \mathcal{P}(V) \mapsto \mathbb{R}$	a general set function
$\mathbf{I}_A$	indicator vector of set A
$\lvert A\rvert$	cardinality of A
$P(f)$	submodular polyhedron
$B(f)$	base polyhedron
$s \in \mathbb{R}^p$	a real-valued vector of length p
s^j	the j^{th} element in s
$s(A)$	the value $\sum_{j \in A} s^j$

Table 2.1: Table of notation in this section.

2.1.1 Definition of A Submodular Function

Let us first introduce the definition of a set function. We say a *set function* is a function whose input is a set and whose output is a real number. Given a finite base set $V = \{1, \cdots, p\}$, $p = |V|$, a set function f maps from the power set $\mathcal{P}(V)$ of the base set V to the set of real numbers $\mathbb{R}$,

$$f : \mathcal{P}(V) \mapsto \mathbb{R},$$

denoted $f(A), \forall A \subseteq V$. For notational convenience, we usually identify the power set $\mathcal{P}(V)$ as 2^V, then identify each element of the power set with a binary indicator vector:

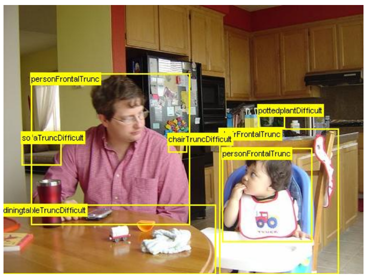
$$\mathbf{I}_A := (\mathbf{1}_x)_{x \in A}, \text{ such that } \mathbf{1}_x := \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A. \end{cases}$$

Namely, a set $A \subseteq V$ can be uniquely defined by the indicator vector $\mathbf{1}_A$, and vice versa.

For example, the function that assigns to each set to its cardinality, i.e. the number of members of the set, is a set function. In practice, there are many problems that can be formulated by using set functions. Figure 2.1 shows an example of a set prediction problem, the multi-label prediction problem. The ground truth of the problem can be formulated as a base set that contains all labels of interest, while each pattern can be seen as a subset of this base set. Any evaluation functions, score functions or loss functions that measure on the quality of an output prediction can be seen as set functions. Depending on the context, we may interpret the input set as the set of predictions, or the set of mispredictions. Without loss of generality, for our mathematical analysis, we will assume that set function on the empty set is zero:

$$f(\emptyset) = 0.$$

Submodular set functions play an important role among these set functions, similar to convex functions on vector spaces, for the reason that submodular functions can be



$$
\begin{aligned}
V = \{&\texttt{`bottle'}, \texttt{`cat'}, \texttt{`chair'}, \texttt{`table'},\\
&\texttt{`bike'}, \texttt{`person'}, \texttt{`sofa'}, \texttt{`dog'},\\
&\texttt{`tv/monitor'}, \texttt{`pottedplant'}\}\\
A = \{&\texttt{`person'}, \texttt{`table'}, \texttt{`chair'}\}
\end{aligned}
$$

Figure 2.1: An example on multi-label prediction problem as a set prediction problem. The groundtruth of the problem can be formulated as a base set that contains all labels of interest, while each pattern can be seen as a subset of this base set. Images are taken from Pascal VOC dataset [Everingham et al., 2010]

minimized in polynomial time. Many functions that occur in practical problems turn out to be submodular functions and we will see several examples in this chapter. Submodular functions may be defined through several equivalent properties:

Definition 2.1 (Submodularity)**.** *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is **submodular** if for all $A, B \subseteq V$, it holds*

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B). \tag{2.1.1}$$

Submodular function is also commonly know by its *diminishing gains* property, that we introduce in the following theorem:

Theorem 2.1 (Submodularity (diminishing returns) [Fujishige, 2005])**.** *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is **submodular** if and only if for all $B \subseteq A \subset V$ and $x \in V \setminus B$, it holds*

$$f(B \cup \{x\}) - f(B) \geq f(A \cup \{x\}) - f(A). \tag{2.1.2}$$

A function is *supermodular* if its negative is submodular, namely it satisfies one of the following equivalent definitions

Definition 2.2 (Supermodularity)**.** *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is **supermodular** if for all subsets $A, B \subseteq V$, if only of it holds*

$$f(A) + f(B) \leq f(A \cup B) + f(A \cap B). \tag{2.1.3}$$

Theorem 2.2. *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is **supermodular** if for all $B \subseteq A \subset V$ and $x \in V \setminus B$, it holds*

$$f(B \cup \{x\}) - f(B) \leq f(A \cup \{x\}) - f(A). \tag{2.1.4}$$

A function is *modular* if it is both submodular and supermodular, namely the equalities in the previous definitions hold. We call a function is *non-modular* if it is not modular. We note that a non-modular function can be neither submodular nor supermodular. Furthermore, a modular function can be written as a dot product between a binary indicator vector in $\{0, 1\}^p$ encoding a subset of V and a coefficient vector in $\mathbb{R}^p$ which uniquely identifies the modular function.

Theorem 2.3. *If a set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is modular, then it can be written as*

$$f(A) = \sum_{j \in A} a^j \tag{2.1.5}$$

for $A \subseteq V$, some coefficient vector $a \in \mathbb{R}^{|V|}$ and the superscript j indicates the j^{th} element in the vector a.

Proof. By definition, $\forall B \subseteq A \subset V$ and $x \in V \setminus B$:

$$f(B \cup \{x\}) - f(B) = f(A \cup \{x\}) - f(A).$$

Set $B = \emptyset$, we have

$$f(A \cup \{x\}) = f(A) + f(\{x\}), \qquad \forall A \subset V.$$

This will give us that

$$f(A) = \sum_{x \in A} f(\{x\}), \qquad \forall A \subseteq V,$$

which is equivalent to have a vector $a \in \mathbb{R}^{|V|}$ and $a^x := f(\{x\})$. $\square$

We sometimes say that a modular function is *decomposable* over the elements of the base set. For example, Hamming loss function (also called the 0-1 loss function) is a modular function with a coefficient vector of all ones and a subset defined by the entries that differ between two sets, that to say counting the number of different elements between two sets.

Submodular functions have some nice closedness properties [Fujishige, 2005]:

Theorem 2.4. *Given $f_1, \ldots, f_m$ submodular functions on V and $\lambda_1, \ldots, \lambda_m \geq 0$, then the function*

$$g(A) = \sum_{i \in \{1,\ldots,m\}} \lambda_i f_i(A), \qquad \forall A \subseteq V$$

is submodular.

Theorem 2.5. *If $A \subseteq V$, $f(A)$ is submodular on V, and $B \subseteq V$*

$$\begin{aligned} g(A) &= f(A \cap B) && \textit{(Restriction)} \\ g(A) &= f(A \cup B) && \textit{(Conditioning)} \\ g(A) &= f(V \setminus A) && \textit{(Reflection)} \end{aligned}$$

are all submodular.

Necessary to the sequel of the thesis, we introduce also some notions of general properties of set functions.

Definition 2.3 (Increasing)**.** *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is **increasing** if for all subsets $A \subset V$ and elements $x \in V \setminus A$, it holds that*

$$f(A) \leq f(A \cup \{x\}). \tag{2.1.6}$$

Definition 2.4 (Non-negativity). *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is non-negative if $f(A) \geq 0,\ \forall A \subseteq V$.*

Definition 2.5 (Symmetry). *A set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is symmetric if $f(A) = c(|A|)$ for some function $c : \mathbb{Z}_+ \mapsto \mathbb{R}$. ($\mathbb{Z}$ is the set of integers and $\mathbb{Z}_+$ is the set of non-negative integers.)*

The following are some examples of submodular functions which can be found in Bach [2013, Chapter 6].

Theorem 2.6 (Cardinality-based set function). *If $f : \mathcal{P}(V) \mapsto \mathbb{R}$ and there exist a function $c : \mathbb{Z}_+ \mapsto \mathbb{R}$ such that $f(A) = c(|A|)$, where $|\cdot|$ is the cardinality of A. Then f is submodular if and only if c is concave.*

Theorem 2.7 (Cut function). *Given a weight vector $d : V \times V \mapsto \mathbb{R}_+$, a cut function is defined as*

$$f(A) = d(A, V \setminus A) = \sum_{k \in V, j \in V \setminus A} d(k, j), \quad \forall A \subseteq V, \tag{2.1.7}$$

where $d(B, C) := \sum_{k \in B, j \in C} d(k, j), \forall B, C \subseteq V$. f is submodular.

Theorem 2.8 (Cover function). *Given a non-negative set function $\tilde{f} : \mathcal{P}(V) \mapsto \mathbb{R}_+$, a cover function is defined as*

$$f(A) = \sum_{B \subseteq V, B \cap A \neq \emptyset} \tilde{f}(B) = \sum_{B \subseteq V} \tilde{f}(B) \min\{1, |A \cap B|\}. \tag{2.1.8}$$

f is submodular.

2.1.2 Submodular Function Analysis

The study of submodular functions is strongly linked with special convex polyhedra that allows us to utilize the methods in convex analysis.

Definition 2.6 (Base polyhedra [Fujishige, 2005; Bach, 2013]). *Given $f : \mathcal{P}(V) \mapsto \mathbb{R}$ a submodular function and $f(\emptyset) = 0$. The submodular polyhedron $P(f)$ and the base polyhedron $B(f)$ are defined as:*

$$P(f) = \{s \in \mathbb{R}^p, \forall A \subseteq V, s(A) \leq f(A)\} \tag{2.1.9}$$

$$B(f) = \{s \in \mathbb{R}^p, s(V) = f(V), \forall A \subseteq V, s(A) \leq f(A)\} \tag{2.1.10}$$

$$= P(F) \cap \{s(V) = f(V)\}. \tag{2.1.11}$$

where $s(A) = \sum_{j \in A} s^j$ and s^j indicates the j^{th} element of s.

We notice that these polyhedra are the intersection of the hyperplanes $\{s \in \mathbb{R}^p, s(A) \leq f(A)\}$. For a modular function $f(A) = \sum_{x \in A} a_x$, $a \in \mathbb{R}^p$, then $P(f) = \{s \in \mathbb{R}^p, \forall x \in V, s_x \leq a_x\} = s \leq a$ which is an isomorph to the negative orthant. Figure 2.2 shows examples with $p = 2$ and $p = 3$. The following theorem makes the strong convex duality holds, which will be made use of in establishing the link between submodular analysis and convex analysis.

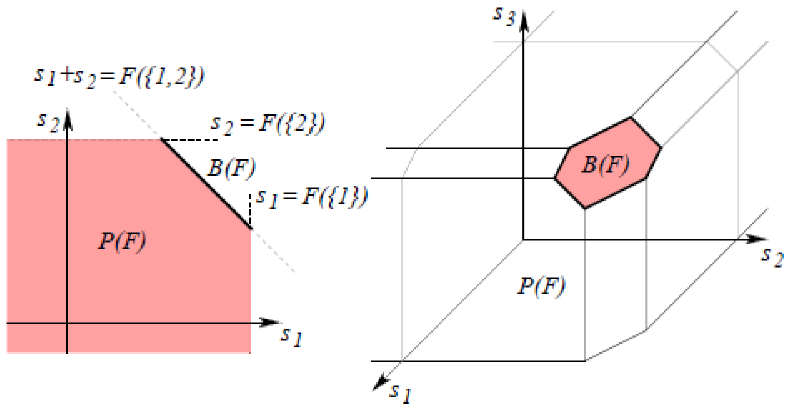


Figure 2.2: Submodular polyhedron $P(F)$ and base polyhedron $B(F)$ for $p = 2$ (left) and $p = 3$ (right), for a non-decreasing submodular function. Illustration from Bach [2013].

Theorem 2.9. *Given f a submodular function with $f(\emptyset) = 0$. If $s \in P(f)$ then for all $t \in \mathbb{R}^p$ such that $t \leq s$ (i.e. , $\forall j \in, t^j \leq s^j$), we have $t \in P(f)$ and $P(f)$ has non-empty interior.*

We can extend a submodular function with discrete domain to a piecewise linear convex function with continuous domain, which is called the Lovász extension [Lovász, 1983]. We now introduce the definition of the Lovász extension of a general (not necessarily submodular) function, which is sometimes called the *Choquet integral* [Choquet, 1953]:

Definition 2.7 (Lovász extension [Lovász, 1983])**.** *Consider a set function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ where $|V| = p$. The Lovász extension $\hat{f} : [0,1]^p \to \mathbb{R}$ of f is defined as follows: $\forall s \in [0,1]^p$ we order its components in decreasing order as $s^{\pi_1} \geq s^{\pi_2} \geq \cdots \geq s^{\pi_p}$ with a permutation π, then $\hat{f}(s)$ is defined as:*

$$\hat{f}(s) = \sum_{j=1}^{p} s^{\pi_j} \left(f\left(\{\pi_1, \cdots, \pi_j\}\right) - f\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)\right). \tag{2.1.12}$$

The Lovász extension is an extension of a set function defined on the vertices of the hypercube $\{0,1\}^p$ to the full hypercube $[0,1]^p$.

Theorem 2.10 (Lovász [1983])**.** *A function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ is submodular if and only if the Lovász extension $\hat{f}$ of f is convex. Moreover, we have*

$$\hat{f}(s) = \max_{\mu \in B(f)} s^\top \mu. \tag{2.1.13}$$

This theorem shows the close relationship between submodularity and convexity. We plot empirically the surface of the Lovász extension for a submodular function f in the case $p = 2$ in Figure 2.3, with $f(\{\emptyset\}) = 0$, $f(\{1\}) = 0.8$, $f(\{2\}) = 0.6$, $f(\{1,2\}) = 0.4$. We note that the surface is convex.

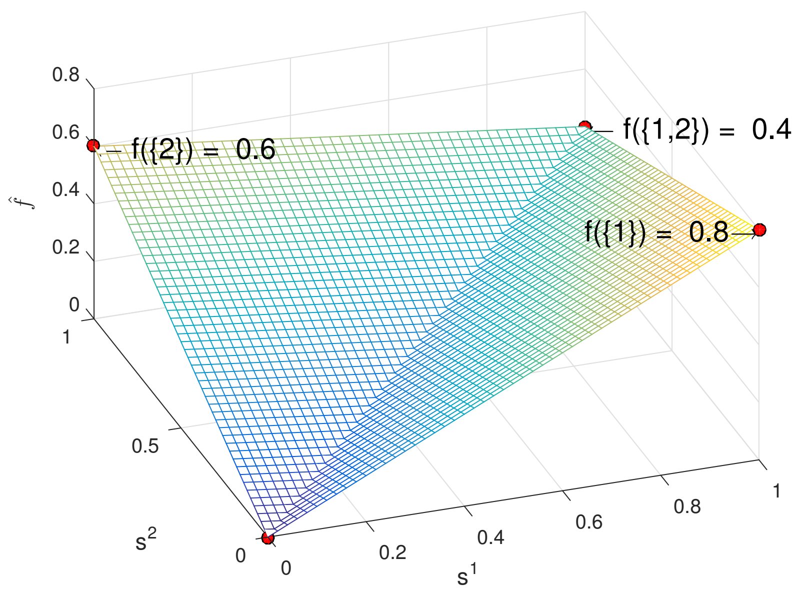


Figure 2.3: The surface of the Lovász extension for a submodular function in case $p = 2$: $f(\{\emptyset\}) = 0$, $f(\{1\}) = 0.8$, $f(\{2\}) = 0.6$, $f(\{1,2\}) = 0.4$. The red dot represent the values of the set function f on the vertex of the unitcube.

In convex analysis, we have that the support function of a non-empty closed convex set $A \in \mathbb{R}^p$ is given by

$$\sup_{\mu \in A} s^\intercal \mu, \forall s \in \mathbb{R}^p,$$

which leads to a convex function of s. The link between the Lovász extension and submodular polyhedra is through the so-called "greedy algorithm" : the Lovász extension is the support function of the base polyhedron and may be computed in closed form.

Theorem 2.11 (Greedy algorithm [Lovász, 1983; Edmonds, 1971]). *Given a submodular function f such that $f(\emptyset) = 0$, $\forall s \in [0,1]^p$ we order its components in decreasing order with a permutation π. We have $P(f)$ the submodular polyhedron and $B(f)$ the base polyhedron. Denote $\mu^{\pi_j} = f(\{\pi_1, \cdots, \pi_j\}) - f(\{\pi_1, \cdots, \pi_{j-1}\})$, then μ is on the base polyhedron i.e. $\mu \in B(f)$, and also*

(i) if $s \in \mathbb{R}^p_+$, μ with the original order is a maximizer of $\max_{\mu \in B(f)} s^\intercal \mu$, and

$$\max_{\mu \in B(f)} s^\intercal \mu = \hat{f}(s);$$

(ii) μ is a maximizer of $\max_{\mu \in P(f)} s^\intercal \mu$, and

$$\max_{\mu \in P(f)} s^\intercal \mu = \hat{f}(s).$$

A proof with detailed analysis can be found in Bach [2013, Proposition 3.2 to 3.5]. That is to say, to find a maximum of $s^\intercal \mu$, which is the same procedure as computing a subgradient of the Lovász extension, is precisely the following procedure:

minimize convex extension	
ellipsoid algorithm	[Grötschel et al., 1988]
minimum norm point	[Fujishige, 1980, 2005]
combinatorial algorithms	
$\mathcal{O}(n^4T + n^5 logM)$	[Iwata, 2003]
$\mathcal{O}(n^6 + n^5T)$	[Orlin, 2009]

Table 2.2: A summary of submodular minimization methods.

(i) sort the p components of s, which is in $\mathcal{O}(p \log p)$;

(ii) compute the value of $s^\intercal \mu$ which needs $\mathcal{O}(p)$ oracle accesses to the set function.

Moreover, we have that the minimum of the Lovász extension is the minimum of the set function:

Theorem 2.12 (Lovász [1983])**.** *For a submodular function $f : \mathcal{P}(V) \mapsto \mathbb{R}$ and its Lovász extension $\hat{f}$, we have*

$$\min_{A \in V} f(A) = \min_{s \in \{0,1\}^p} \hat{f}(s) = \min_{s \in [0,1]^p} \hat{f}(s). \tag{2.1.14}$$

This theorem shows that the Lovász extension as a convex relaxation is *exact*, which implies that submodular minimization is polynomial time solvable [Grötschel et al., 1981; Lovász, 1983; Schrijver, 2002]. Table 2.2 lists general submodular minimization algorithms.

Theorem 2.13. *Given two submodular functions f_1 and f_2, then their maximum*

$$f_{max}(A) = \max\{f_1(A), f_2(A)\}, \forall A \subseteq V$$

is not *submodular in general (see e.g. Section 1.2 [Krause and Golovin, 2014]).*

Proof. We can take a simple counterexample. Take f_1 and f_2 both symmetric and can be written as some concave functions with respect to the cardinality of the input set A. It is then trivial to see that the maximum of two concave functions is in general *not* concave. Thus by theorem 2.6, the maximum of f_1 and f_2 is in general *not* submodular. This counterexample is illustrated in Figure 2.4. □

Theorem 2.14. *Given two submodular functions f_1 and f_2, then their minimum*

$$f_{min}(A) = \min\{f_1(A), f_2(A)\}, \forall A \subseteq V$$

is not *submodular in general (see e.g. Section 1.2 [Krause and Golovin, 2014]).*

Proof. We can come up a counterexample as follows: in the case of $p = 2$, we set:

$$f_1(\{\emptyset\}) = 0, f_1(\{1\}) = 1, f_1(\{2\}) = 0, f_1(\{1,2\}) = 0.8;$$
$$f_2(\{\emptyset\}) = 0, f_2(\{1\}) = 0, f_2(\{2\}) = 1, f_2(\{1,2\}) = 0.8.$$

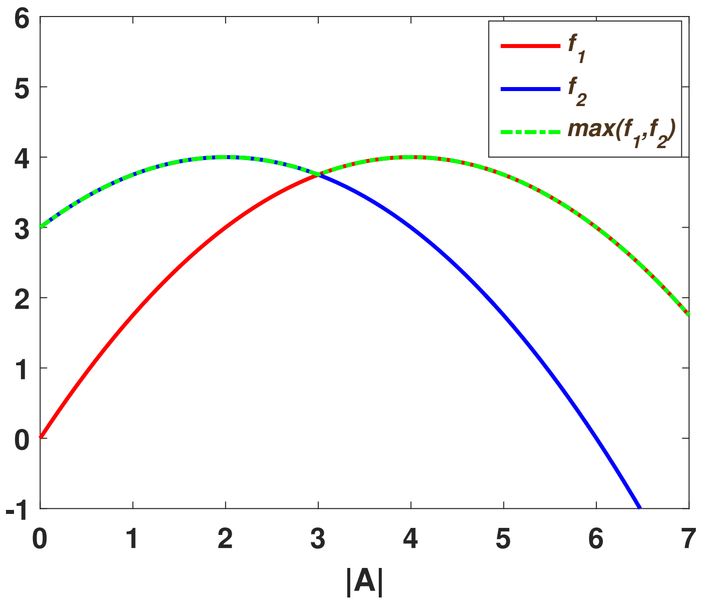


Figure 2.4: An example shows that the maximum of two concave functions is not concave in general, thus the maximum of two submodular functions is not submodular in general.

Then both f_1 and f_2 are submodular. However, for $f_{\text{min}} = \min\{f_1, f_2\}$:

$$f_{\text{min}}(\{\emptyset\}) = 0, f_{\text{min}}(\{1\}) = 0, f_{\text{min}}(\{2\}) = 0, f_{\text{min}}(\{1,2\}) = 0.8$$

is *not* submodular. □

2.2 Maximum A Posteriori Inference and Submodularity

One of the most important applications of submodularity in machine learning and computer vision is the Maximum A Posteriori (MAP) inference problem solved by graph cuts under submodularity constraints. In this section we will introduce some basic concepts that will be necessary to the following chapters in this thesis.

2.2.1 Markov Random Field

In the field of probability, Markov random fields (MRF) or undirected graphical models, are tools for modeling data (often images), that relate graph theory and probability theory. We denote $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ for the notation of a graph, with the set of vertices (also called nodes) $\mathcal{V} = \{1, 2, \ldots, n\}$, e.g. (super)pixels in images, and the set of edges $\mathcal{E}$ where a typical edge is $(i, j) \in \mathcal{E}, i, j \in \mathcal{V}$, which will be determined by the structure of the graph. Figure 2.5(a) shows a 4-connected neighborhood and Figure 2.5(b) shows an 8-connected neighborhood graph. All pairs of nodes that are connected form the edge set $\mathcal{E}$. Following the notation in [Koller et al., 2007], we denote $\mathcal{N}_{i,\mathcal{E}}$ the set of neighbors of i in the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

An MRF is determined by

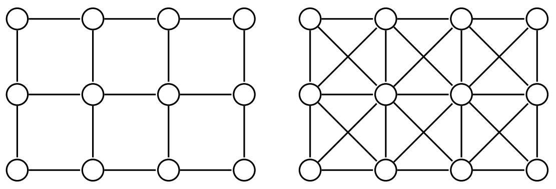

(a) 4-connected neighborhood graph (b) 8-connected neighborhood graph

Figure 2.5: Examples of different graph structures. Figure 2.5(a) shows a 4-connected neighborhood graph and Figure 2.5(b) shows an 8-connected neighborhood graph.

1. a set of nodes $V = \{1, 2, \ldots, n\}$;
2. a set of random variables $y = \{y^1, y^2, \ldots, y^n\}$ associated with each of the nodes;
3. a set of pairs of neighbors, i.e. the edge set $\mathcal{E}$, that indicate the probabilistic connection between the variables.

To be an MRF, the Markov property should be obeyed:

$$P(y^i | x^{V|\{i\}}) = P(y^i | y^j, j \in \mathcal{N}_{i,\mathcal{E}}). \tag{2.2.1}$$

Theorem 2.15 (Hammersley Clifford Theorem [Clifford, 1990]). *Any distribution P that obeys the Markov property in Equation (2.2.1) if and only if P can be written in the* Gibbs distribution*:*

$$P(y) = \frac{1}{Z} \exp\left(\sum_{c \in \mathcal{C}} -\Psi_c(y)\right), \tag{2.2.2}$$

where $\mathcal{C}$ is the set of maximal cliques of $\mathcal{G}$, and Z is the partition function:

$$Z = \sum_y \exp\left(\sum_{c \in \mathcal{C}} -\Psi_c(y)\right). \tag{2.2.3}$$

The functions Ψ are defined so as to form an energy function $E(y) = \sum_{c \in \mathcal{C}} \Psi_c(x)$. In general, if we consider that these functions are parametrized by a vector w, we have

$$E(y, w) = \sum_{c \in \mathcal{C}} \Psi_c(y, w), \tag{2.2.4}$$

$$P(y) = \frac{1}{Z(w)} \exp\left(-E(y, w)\right). \tag{2.2.5}$$

In pairwise graphs, the energy function includes unary potentials that are decomposable over the nodes of the graph, and the pairwise potentials that are decomposable over the edges:

$$E(y, w) = \underbrace{\sum_{i \in \mathcal{V}} U_i(y^i, w)}_{\text{unary potential}} + \underbrace{\sum_{(k,l) \in \mathcal{E}} P_{kl}(y^k, y^l, w)}_{\text{pairwise potential}}. \tag{2.2.6}$$

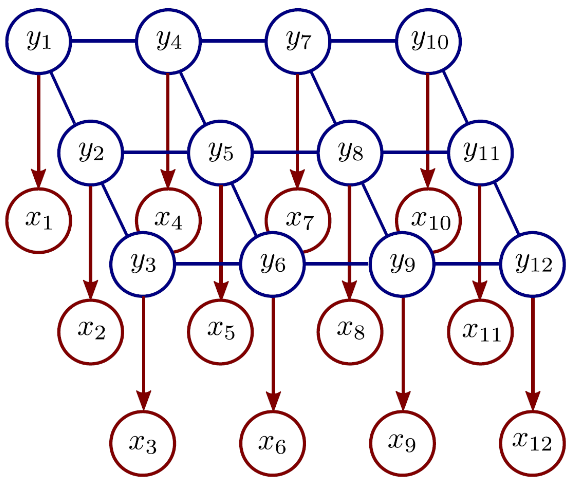


Figure 2.6: An illustration of MRF with observed variable x and unobserved variable y in a 4-connected graph structure. Illustration from Prince [2012].

2.2.2 Maximum A Posteriori

The maximum a posteriori (MAP) inference is one of the most common estimations of an MRF in computer vision problems. Figure 2.6 shows an illustration of an MRF with observed variables x^i and unobserved variables y^i. This is often the case in image segmentation problem with x^i the features and y^i the groundtruth labels. Given an observed variable x, we want to make our hypothesis on $y := \hat{h}(x)$ such that

$$\hat{h}(x) := \arg\max_y P(y|x) = \arg\max_y \frac{P(x|y)P(y)}{P(x)} = \arg\max_y P(x|y)P(y) \tag{2.2.7}$$

with a Markov prior on y, which gives us a $P(y)$ and

$$P(x|y) \propto \prod_i \exp(-E(x,y,w)) \tag{2.2.8}$$

are our *unary* probabilities (potentials). Then the MAP inference problem is equivalent to an energy minimization problem with:

$$\hat{h}(x) = \arg\min_y \sum_i U_i(x^i, y^i, w) + \sum_{(k,l)\in E} P_{kl}(x^k, x^l, y^k, y^l, w), \tag{2.2.9}$$

which for pairwise potential can be solved by a graph cut formulation (Figure 2.7)

2.2.3 Graph Cut and Submodular constraints

Under the context of MRF, let's first consider the binary case, namely $y^i \in \{0,1\}$. The unary potential $U(y^i,\cdot)$ in the energy function in Equation (2.2.6) is decomposable over the nodes in the graph, thus is modular. The pairwise potential $P(y^k, y^l, \cdot)$ is submodular (Definition 2.1) if for every pair of variables $\forall (k,l) \in \mathcal{E}$ we have

$$P_{kl}(0,0) + P_{kl}(1,1) \leq P_{kl}(1,0) + P_{kl}(0,1). \tag{2.2.10}$$

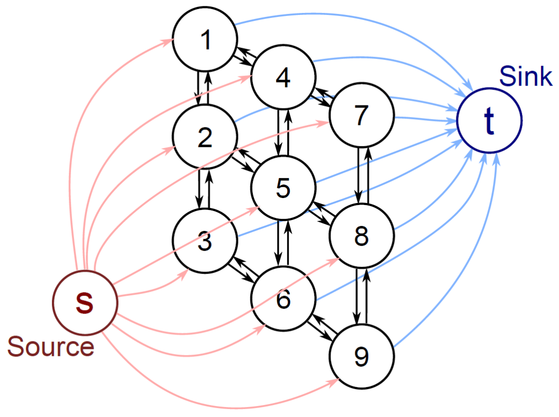


Figure 2.7: An illustration of graph-cut (max-flow min-cut) problem with node *Source* and *Sink*. Illustration from Prince [2012].

More generally in the case of multilabels, i.e. $y^i \in \{1, 2, \cdots, N\}$, the submodularity constraint for every pair of variables $\forall(k,l) \in \mathcal{E}$ is

$$P_{kl}(\alpha_1, \beta_1) + P_{kl}(\alpha_2, \beta_2) \leq P_{kl}(\alpha_1, \beta_2) + P_{kl}(\alpha_2, \beta_1), \tag{2.2.11}$$

where $\alpha_1 \leq \alpha_2$ and $\beta_1 \leq \beta_2$.

Theorem 2.16 (Graph-cuts solvable [Kolmogorov and Zabin, 2004])**.** *Given the energy minimization problem in Equation* (2.2.9)*,i.e. the energy is a summation of unary potential and pairwise potential, exact inference is solvable by graph-cuts (also called max-flow min-cut algorithm) if the energy is submodular, namely obeys Equation* (2.2.10) *for binary problems and Equation* (2.2.11) *for multilabel problems.*

2.3 Empirical Risk Minimization

Empirical risk minimization is a statistical principle underline much of machine learning.

2.3.1 Empirical Risk

We are interested in the general problem of learning a function $\hat{h} : \mathcal{X} \mapsto \mathcal{Y}$ (often called a *hypothesis*, or prediction function) that maps from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$. In a supervised learning scenario, we are given a training set of labelled samples $\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$. For instance, in a binary image segmentation task, $\hat{h}$ maps from one input color image to a binary mask that indicates the region of the foreground object.

In order to quantify the performance of a hypothesis $\hat{h}$, we will consider learning with a discrete *loss* function

$$\Delta : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}_+.$$

$$y = \begin{pmatrix} +1 \\ +1 \\ +1 \\ +1 \\ \vdots \\ +1 \end{pmatrix} \quad \text{vs} \quad \tilde{y} = \begin{pmatrix} +1 \\ -1 \\ -1 \\ +1 \\ \vdots \\ +1 \end{pmatrix} \quad \Longrightarrow \quad \text{misprediction set } = \{2, 3, \ldots\} \subset V$$

Figure 2.8: The loss function is considered as a set function, with the input set equals to the misprection set, which is a subset of the base set V (Equation (2.3.1)).

$\Delta(y, \tilde{y})$ measures the mismatch between a ground truth y, and a predicted response that we get $\tilde{y} = \hat{h}(x)$. We will assume that $\Delta(y, y) = 0$, and $\Delta(y, \tilde{y}) \geq 0$, $\forall \tilde{y} \neq y$. In the meantime, if $\mathcal{Y}$ has some structure that allows its element y to be represented as a subset of a base set V s.t. $|V| = p$, $\Delta(y, \cdot)$ will be isomorphic to a set function $\ell : \mathcal{P}(V) \mapsto \mathbb{R}_+$ for which the input set is the misprediction elements in the base set V (Figure 2.8)

$$\Delta(y, \tilde{y}) = \ell(\{j | y^j \neq \tilde{y}^j\}). \tag{2.3.1}$$

We can now analyse loss functions as set function, and we can inherit properties of set functions such as submodularirty or supermodularity.

The risk associated with the hypothesis $\hat{h}$ is then defined as the expectation of the loss function

$$\mathcal{R}(\hat{h}) = \mathbf{E}[\Delta(y, \hat{h}(x))] = \int_{\mathcal{X}\times\mathcal{Y}} \Delta(y, \hat{h}(x))\, dP(x, y). \tag{2.3.2}$$

The goal is to minimize the risk $\mathcal{R}$ over a class of functions $\mathcal{F}$. However, the joint probability distribution $P(x, y) = P(y|x)P(x)$ is unknown and the only available information is contained in the training set.

In empirical risk minimization, we approximate the risk by an empirical sum over losses incurred on the finite sample, using e.g. an i.i.d. sampling assumption [Vapnik, 1995]:

$$\mathcal{R}(\hat{h}) \underbrace{\approx}_{\text{i.i.d.}} \hat{\mathcal{R}}(\hat{h}) := \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i^*, \hat{h}(x_i)) \tag{2.3.3}$$

2.3.2 Surrogate Functions

As it is defined, the loss function Δ may be in general over a non-convex domain and is in general NP-hard to minimize directly. Central to the practical application of the empirical risk minimization principle, one must approximate, or upper bound the discrete loss function Δ with a *surrogate function*, frequently a convex one for computational reasons. Figure 2.9 shows some commonly used convex surrogates for binary classification problems ($\mathcal{Y} = \{-1, +1\}$), including the hinge loss, $\max(1 - yh(x), 0)$, and the squared hinge loss (L2 loss), $\max(1 - yh(x), 0)^2$, used by the Support Vector Machine (SVM) [Cortes and Vapnik,

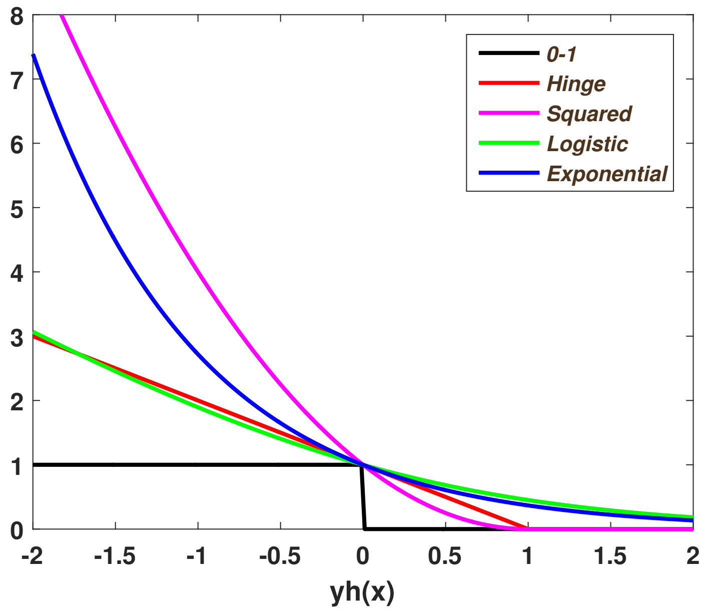


Figure 2.9: Plots of the 0-1 loss and some common used surrogate functions including: hinge loss, squared hinge loss, logistic regression loss, and exponential loss. The functions are plotted as a function of the margin $yh(x)$.

1995; Crammer and Singer, 2001]; the log-loss, $\log(1+\exp(-yh(x))$, used in logistic regression [Friedman et al., 2000]; the exponential loss, $\exp(-yh(x))$, used in `AdaBoost` [Freund and Schapire, 1995]. The functions are plotted as a function of the margin $yh(x)$.

Convexity provides strong advantages regarding computational complexity. Bartlett et al. [2006] have studied the consistency of these non-negative surrogate loss functions. They provided a quantitative relationship between the risk assessed by using these surrogate functions and the risk accessed by using the 0-1 loss.

2.4 Structured Output SVM

In structured prediction problems, we consider the case where elements of $\mathcal{Y}$ are structured objects such as sequences, strings, trees or graphs. The Structured Output SVM (SOSVM) is a popular framework in the regularized risk minimization framework [Taskar et al., 2004; Tsochantaridis et al., 2005] that aims at learning a parametrized function $h : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ over input/output pairs from which a prediction can be derived by maximizing

$$\hat{h}(x) = \arg\max_{y \in \mathcal{Y}} h(x, y; w) \tag{2.4.1}$$

over the response from a given input x. The SOSVM framework assumes h to be represented by an inner product between an element of a reproducing kernel Hilbert space and some combined feature representation of inputs and outputs $\phi(x, y)$

$$h(x, y; w) = \langle w, \phi(x, y) \rangle, \tag{2.4.2}$$

Algorithm 1 Cutting plane algorithm for solving the problem in Equation (2.4.3) and (2.4.4) – Margin rescaling.

```
1: Input: $(x_1, y_1), \cdots, (x_n, y_n), C, \epsilon$
2: $S^i = \emptyset, \forall i = 1, \cdots, n$
3: repeat
4:   for $i = 1, \cdots, n$ do
5:     $\hat{y}_i = \arg\max_y H(y_i, \tilde{y}) = \arg\max_{\tilde{y}} \Delta(y_i, y) + \langle w, \phi(x_i, \tilde{y})\rangle - \langle w, \phi(x_i, y_i)\rangle$ % most violated constraint
6:     $\xi^i = \max\{0, H(y_i, \hat{y}_i)\}$
7:     if $H(y_i, \hat{y}_i) > \xi^i + \epsilon$ then
8:       $S^i := S^i \cup \{\hat{y}_i\}$
9:       $w \leftarrow$ optimize Equation (2.4.3) with constraints defined by $\cup_i S^i$
10:    end if
11:  end for
12: until no $S^i$ has changed during an iteration
13: return $(w, \xi)$
```

but we will see in the sequel that we can relax this dependence on the class of functions (cf. Chapter 4).

In order to train the weight vector w, the following formulations are proposed:

$$\min_{w,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i, \tag{2.4.3}$$

$$\text{s.t. } \forall i, \forall \tilde{y} \in \mathcal{Y} : \langle w, \phi(x_i, y_i)\rangle - \langle w, \phi(x_i, \tilde{y})\rangle \geq \Delta(y_i, \tilde{y}) - \xi_i \tag{2.4.4}$$

$$\text{or} \quad \Delta(y_i, \tilde{y})\left(\langle w, \phi(x_i, y_i)\rangle - \langle w, \phi(x_i, \tilde{y})\rangle\right) \geq \Delta(y, \tilde{y}) - \xi_i \tag{2.4.5}$$

called margin-rescaling constraints and slack-rescaling constraints, respectively.

Cutting Plane Method A cutting plane algorithm [Joachims et al., 2009] is commonly used to solve this optimization problem, applied to both margin-rescaling and slack rescaling, as shown in Algorithm 1 and in Algorithm 2.

Loss Augmented Inference As shown in the algorithms, the bottleneck of the framework is in the Line 5, where a maximization step needs to be solved. This a maximization on a summation (or multiplication) of the loss function Δ and the joint feature function with respect to an output y . We call this step loss augmented inference:

$$\arg\max_{\tilde{y}\in\mathcal{Y}} \Delta(y_i, \tilde{y}) + \langle w, \phi(x_i, \tilde{y})\rangle - \langle w, \phi(x_i, y_i)\rangle, \tag{2.4.6}$$

$$\arg\max_{\tilde{y}\in\mathcal{Y}} \Delta(y_i, \tilde{y})(1 + \langle w, \phi(x_i, \tilde{y})\rangle - \langle w, \phi(x_i, y_i)\rangle), \tag{2.4.7}$$

for margin rescaling and slack rescaling, respectively.

Algorithm 2 Cutting plane algorithm for solving the problem in Equation (2.4.3) and (2.4.5) – Slack rescaling.

```
1: Input: $(x_1, y_1), \cdots, (x_n, y_n), C, \epsilon$
2: $S^i = \emptyset, \forall i = 1, \cdots, n$
3: **repeat**
4:    **for** $i = 1, \cdots, n$ **do**
5:       $\hat{y}_i = \arg\max_y H(y_i, y) = \arg\max_{\tilde{y}} \Delta(y_i, \tilde{y})(1 + \langle w, \phi(x_i, \tilde{y})\rangle - \langle w, \phi(x_i, y_i)\rangle)$ % most
         violated constraint
6:       $\xi^i = \max\{0, H(y_i, \hat{y}_i)\}$
7:       **if** $H(y_i, \hat{y}_i) > \xi^i + \epsilon$ **then**
8:          $S^i := S^i \cup \{\hat{y}_i\}$
9:          $w \leftarrow$ optimize Equation (2.4.3) with constraints defined by $\cup_i S^i$
10:      **end if**
11:   **end for**
12: **until** no $S^i$ has changed during an iteration
13: **return** $(w, \xi)$
```

In the case that Δ is a modular loss function, and $\langle w, \phi(x, \tilde{y})\rangle$ corresponds to a random field model with submodular potentials, loss augmented inference can be solved by a graph-cut procedure by modifying the unary potentials [Anguelov et al., 2005; Szummer et al., 2008].

Chapter 3

Efficient Learning with Supermodular Loss Functions

Contents

3.1 Related Work 28
3.2 Loss Augmented Inference (LAI) 30
3.3 A Supermodular Loss Function for Binary Segmentation 32
3.3.1 A novel supermodular loss function 32
3.3.2 Non-submodularity of the LAI 33
3.4 ADMM Algorithm for LAI 34
3.5 Experimental Results 36
3.6 Discussion 40

Overview Several supermodular losses have been shown to improve the perceptual quality of image segmentation in a discriminative framework such as a structured output SVM. These loss functions do not necessarily have the same structure as the segmentation inference algorithm, and in general, we may have to resort to generic submodular minimization algorithms for loss augmented inference. Although these come with polynomial time guarantees, e.g. $\mathcal{O}(n^4T + n^5 \log M)$ using the algorithm of Iwata (T being the time for a single function evaluation, and M being a bound on the largest absolute value of the function), or a pseudo-polynomial time guarantee for the Fujishige-Wolfe minimum norm algorithm, they are not practical to apply to image scale data. Many supermodular losses come with strong optimization guarantees, but are not readily incorporated in a loss augmented graph cuts procedure. This motivates our strategy of employing an ADMM decomposition for loss augmented inference. In doing so, we create a new API for the structured output SVM that separates the MAP inference of the model from the loss augmentation during training. In this way, we gain computational efficiency, making new choices of loss functions practical for the first time, while simultaneously making the inference algorithm employed during training closer to the test time procedure. We show improvement both in accuracy and computational performance on the Microsoft Research Grabcut database and a brain structure segmentation task, empirically validating the use of a supermodular loss during training, and the improved computational properties of the proposed ADMM approach over the Fujishige-Wolfe minimum norm point algorithm. An open source implementation of the proposed learning framework is released at `https://github.com/yjq8812/efficientSegmentation`. This work presented in this chapter is base on the following paper:

- Jiaqian Yu and Matthew B. Blaschko. Efficient learning for discriminative segmentation with supermodular losses. In Proceedings of the British Machine Vision Conference. BMVA Press, 2016.

3.1 Related Work

Discriminative structured prediction is a valuable tool in computer vision that has been applied to a wide range of application areas, and in particular object detection and segmentation [Anguelov et al., 2005; Blaschko and Lampert, 2008; Nowozin and Lampert, 2011; Osokin and Kohli, 2014; Pletscher and Kohli, 2012; Szummer et al., 2008]. It is frequently applied using variants of the structured output support vector machine (SVM) [Taskar et al., 2004; Tsochantaridis et al., 2005] in which a domain specific discrete loss function is upper bounded by a piecewise linear surrogate. In the case of image segmentation, this discrete loss function has frequently been taken to be the Hamming loss, which simply counts the number of incorrect pixels (see e.g. [Anguelov et al., 2005; Szummer et al., 2008]). Following the principle of empirical risk minimization, one might expect that minimization of the desired loss at training time would lead to the best performing loss at test time. However, it has recently been shown that in the finite sample regime, minimizing a different loss can lead to better performance even when measured using Hamming loss [Osokin and Kohli, 2014]. In that work, a supermodular loss function was employed, and a custom graph cuts solution was found to the loss augmented inference problem necessary for computation of a subgradient or cutting plane of the learning objective [Joachims et al., 2009].

Several non-modular loss functions have been considered in the context of image segmentation. Nowozin [2014] has considered the intersection over union loss in the context of a Bayesian framework. This loss made popular by such benchmarks as the PASCAL VOC segmentation challenge [Everingham et al., 2010] is non-modular (c.f. Chapter 4). Ranjbar et al. [2010] trained with a piecewise linear approximation (which is modular w.r.t false negatives and false positives) for the intersection over union loss. Other supermodular losses are proposed such as an area/volume based label-count loss that enforces high-order statistics [Pletscher and Kohli, 2012], or a layout-aware loss function that takes into account the topology/structure of the object [Osokin and Kohli, 2014]. A message passing based optimization scheme is proposed for optimizing several families of structured loss functions [Tarlow et al., 2010; Tarlow and Zemel, 2012], which assumes the loss function is constructed by a grammar for which the productions specify function composition [Tarlow et al., 2010]. By contrast, we provide a generic framework for decomposing the loss function from model inference that assumes a custom solver for the loss, but that does not assume the loss belongs to a specific compositional grammar. We concern ourselves primarily with supermodular loss functions in this work as they lead to provable polynomial time loss augmented inference problems (an essential step in training structured output SVMs), while non-supermodular loss functions lead to NP-hard optimization in general.

It is a time consuming process to develop custom loss-augmented solvers for different combinations of loss functions and inference procedures. We show in this work that a direct combination of two submodular graph cuts procedures may in fact lead to a non-submodular minimization problem, and reparametrizations or novel graph constructions may be necessary. Furthermore, if we attempt to solve a non-submodular minimization problem approximately, this may lead to poor convergence of the learning procedure and catastrophic failure of the learning algorithm has been observed in this case [Finley and Joachims, 2008].

An alternative approach is to resort to generic submodular optimization algorithms, such as that of Iwata [Iwata, 2003] which has complexity $\mathcal{O}(n^4T + n^5 \log M)$, or Orlin [Orlin, 2009] with complexity $\mathcal{O}(n^6+n^5T)$, where T is the time for a single function evaluation and M is an upper bound on the absolute value of the function. Although these optimization algorithms are polynomial, the exponent is sufficiently large as to render them infeasible for images of even less than one megapixel. In practice, the Fujishige-Wolfe minimum norm algorithm [Fujishige, 1980, 2005] is empirically faster despite having comparatively limited theoretical guarantees [Chakrabarty et al., 2014]. However, we will show that even this state of the art generic optimization strategy is infeasible for relatively small consumer images.

Specific subclasses of submodular functions come with lower complexity optimization algorithms, and we should be able to exploit these known classes in a general learning framework. Examples include decomposable submodular functions [Stobbe and Krause, 2010; Nishihara et al., 2014], several notions of symmetry [Kolmogorov, 2012; Queyranne, 1998], and graph partition problems [Kolmogorov and Zabin, 2004; Charpiat, 2011]. A problem with the current API for loss augmented inference is that it is assumed that the loss function will decompose with a structure compatible to that of the inference problem. We address the case that this assumption does not hold and that separate efficient optimization procedures are available for the loss and for inference.

We propose to use Lagrangian splitting techniques to separate loss maximization from the inference problem. Strategies such as dual decomposition have become popular in Markov Random Fields (MRF) inference [Komodakis et al., 2007], while later developments such as the alternating direction method of multipliers (ADMM) [Bertsekas, 1999; Boyd et al., 2011] have improved convergence guarantees. Other strategies involving a quadratic penalty term have also been proposed in the literature (although still with the assumption that the loss decomposes as the inference) [Meshi et al., 2015]. We make use of ADMM to separate these inference problems and apply them to a supermodular loss function that cannot be straightforwardly incorporated in a submodular graph partition problem for loss augmented inference. Instead we allow separate optimization strategies for the loss maximization and inference procedures yielding substantially improved computational performance, while making feasible the application of a wide range of supermodular loss functions by changing a single line of code.

3.2 Loss Augmented Inference (LAI)

We discriminatively train a graph cuts based segmentation system using a structured output SVM [Tsochantaridis et al., 2005]. We construct a supermodular loss function that is solvable with graph cuts, but that when incorporated in a joint loss-augmented inference leads to non-submodular potentials which causes graph cuts based optimization to fail. We therefore use an ADMM based decomposition strategy to perform loss augmented inference. This strategy consists of alternatingly optimizing the loss function and performing maximum *a posteriori* (MAP) inference, with each process augmented by a quadratic term enforcing the labeling determined by each to converge to the optimum of the sum.

Recall that we are given a training set of labeled images $\{(x_1, y_1), \dots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$. In a binary segmentation problem we have

$$\mathcal{Y} = \{-1, 1\}^p$$

where $+1$ usually represents foreground pixels and -1 usually represents background pixel. The structured output SVM is a discriminative learning framework that has been applied in diverse computer vision applications [Anguelov et al., 2005; Blaschko and Lampert, 2008; Nowozin and Lampert, 2011; Osokin and Kohli, 2014; Pletscher and Kohli, 2012; Szummer et al., 2008]. It optimizes a regularized convex upper bound to a structured loss function, $\Delta : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}_+$. Δ measures the mismatch between a ground truth labelling, and a hypothesized labeling. With Δ provided as an input, the structured output SVM with margin rescaling minimizes [Tsochantaridis et al., 2005]:

$$\min_{w,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \tag{3.2.1}$$

$$\text{s.t. } \forall i, \tilde{y}_i \in \mathcal{Y}, \quad \langle w, \phi(x_i, y_i)\rangle - \langle w, \phi(x_i, \tilde{y}_i)\rangle \geq \Delta(y_i, \tilde{y}_i) - \xi_i. \tag{3.2.2}$$

In the case of image segmentation, we may interpret $\langle w, \phi(x, y)\rangle$ as a function that is monotonic in the log probability of the joint configuration of observed and unobserved variables (x, y) as determined by a CRF [Lafferty et al., 2001]. Under this interpretation, a standard definition of ϕ is

$$\phi(x, y) := \begin{pmatrix} \sum_{j=1}^{p} \phi_u(x, y^j) \\ \sum_{(k,l)\in\mathcal{E}} \phi_p(x, y^k, y^l) \end{pmatrix} \tag{3.2.3}$$

where ϕ_u determines a vector of features, a linear combination of which form the unary potentials of the CRF, and ϕ_p determines the pairwise potentials over a model specific edge set $\mathcal{E}$. In this chapter, we have set $\phi_p(x,\cdot,\cdot) : \{-1,1\}^2 \mapsto \{0,1\}^3$ to map to an indicator vector of three cases: (i) $y^k = y^l = -1$, (ii) $y^k \neq y^l$, or (iii) $y^k = y^l = +1$, and have placed hard constraints on the corresponding entries of w in the optimization of the structured output SVM to ensure that the pairwise potentials in the corresponding energy minimization problem remain submodular.

During training of the structured output SVM, we must perform *loss augmented inference* in order to compute a subgradient or cutting plane of the loss function. In the case of margin rescaling, this consists of computing

$$\arg\max_{\tilde{y}\in\mathcal{Y}} \langle w, \phi(x,\tilde{y})\rangle + \Delta(y,\tilde{y}). \tag{3.2.4}$$

For simplicity, we will discard the subscript i in the sequel.

As shown in Section 2.3, we consider that $\mathcal{Y}$ is isomorphic to $\{-1,1\}^p$ for some p, $\Delta(y,\cdot)$ is isomorphic to a set function $\ell : \mathcal{P}(V) \mapsto \mathbb{R}_+$ where $\mathcal{P}(V)$ is the power set of a base set with $|V| = p.$, for which the input set is the misprediction elements in the base set V

$$\Delta(y,\tilde{y}) = \ell(\{j|y^j \neq \tilde{y}^j\}) \tag{3.2.5}$$

This allows us to discuss the properties of such loss functions Δ in terms of the language of set functions as occurs in real analysis [Kolmogorov and Fomin, 1975] and discrete optimization [Schrijver, 2002]. In particular in this chapter, we are interested in Δ corresponding to a supermodular set function ℓ [Schrijver, 2002]: A supermodular function is a set function $\ell : \mathcal{P}(V) \mapsto \mathbb{R}$ which satisfies:

$$\begin{gathered}\forall A, B \subseteq V,\ A \subseteq B,\ v \in V \setminus B,\\ \ell(A\cup\{v\}) - \ell(A) \leq \ell(B\cup\{v\}) - \ell(B)\end{gathered}$$

following Definition 2.2 in Chapter 2.

In order to achieve computational feasibility during test time on the inference problem, we have guaranteed that maximization of $\langle w, \phi(x_i,\tilde{y}_i)\rangle$ with respect to $\tilde{y}$ corresponds to a submodular minimization problem. Therefore, a graph-cuts algorithm can be utilized to solve the maximization problem which is equivalent to a energy minimization problem.

Modular loss functions, such as Hamming loss, can be incorporated into the unary potentials in a graph cuts optimization framework for loss augmented inference. However, the formulation of loss augmented inference with supermodular losses as a graph cuts problem is not straightforward, despite previous work (in which a custom graph cuts formulation was derived for a specific family of supermodular losses) that indicated a supermodular loss can lead to improved segmentation quality [Osokin and Kohli, 2014].

While supermodular loss functions guarantee polynomial time solvability, they do not do so with low order polynomial guarantees in general. In fact, minimization of general submodular functions is $\mathcal{O}(n^4T + n^5 \log M)$ (where T is the cost of function evaluation and M

an upper bound on the absolute value of the function) using the algorithm of Iwata [Iwata, 2003]. In practice, general submodular minimization is often most efficiently solved using the Fujishige-Wolfe minimum norm algorithm [Fujishige, 1980], for which a pseudo-polynomial time guarantee has only been recently proven [Chakrabarty et al., 2014]. We have observed, however, that the Fujishige-Wolfe algorithm is infeasible to apply even in the case of sub-megapixel images, and scales poorly for useful supermodular loss functions. Consequently, we develop a general framework for decomposing loss augmented inference based on ADMM. This framework solely relies on a loss function being able to be efficiently optimized in isolation using a specialized solver specific to the loss function.

3.3 A Supermodular Loss Function for Binary Segmentation

3.3.1 A novel supermodular loss function

We propose a loss function that is itself optimizable with graph cuts. The loss simply counts the number of incorrect pixels plus the number of pairs of neighboring pixels that both have incorrect labels

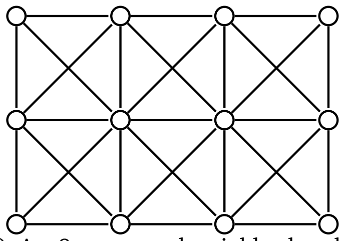
$$\Delta(y,\tilde{y}) = \sum_{j=1}^{p}[y^j \neq \tilde{y}^j] + \sum_{(k,l)\in\mathcal{E}_\ell} \gamma[y^k \neq \tilde{y}^k \wedge y^l \neq \tilde{y}^l] \tag{3.3.1}$$

where $[\cdot]$ is Iverson bracket notation, $\mathcal{E}_\ell$ is a loss specific edge set and γ is a positive weight. We have used 8-connectivity for the loss function in the experiments (Figure 3.1(a)), referred to as "8-connected loss" in the sequel. We may identify this function with a set function to which the argument is the set of mispredicted pixels.

Proposition 3.1. *Maximization of the loss function in Equation (3.3.1) is isomorphic to a supermodular function maximization problem.*

Proof sketch. Equation (3.3.1) is isomorphic to a binary random field model for which label is 1 iff a pixel has a different label from the ground truth. Under this isomorphism, neighboring pixels that both have label 1 contribute a positive amount to the energy, while all other configurations contribute zero. This corresponds to a supermodular function following Definition 2.2. □

This loss function emphasizes the importance of correctly predicting adjacent groups of pixels, e.g. those present in thin structures more than one pixel wide. While the pairwise potential in $\langle w, \phi(x,y)\rangle$ has a tendency to reduce the perimeter of the segment, the loss strongly encourages to correctly identify adjacent pixels. We will observe in the experimental results that the use of this loss function during training improves the test time prediction accuracy, even when measuring in terms of Hamming loss.


(a) An 8-connected neighborhood is used in the construction of the loss function.

$$E = -\overbrace{\begin{pmatrix} w_{00} & w_{01} \\ w_{10} & w_{11} \end{pmatrix}}^{\text{inference pairwise potential}} - \underbrace{\begin{pmatrix} 0 & \gamma \\ 0 & 0 \end{pmatrix}}_{\text{loss pairwise potential}}$$

(b) Pairwise potential construction for an edge with $y^k = +1$ and $y^l = -1$ following the loss function in Equation (3.3.1).

Figure 3.1: Non-submodularity of the joint loss augmented inference procedure using the same mapping to a set function for inference and loss functions. The inference procedure can be solved by graph cuts when the sum of the diagonal elements of E is less than the sum of the off diagonal elements. While it is enforced during optimization that $w_{00} + w_{11} - w_{01} - w_{10} \geq 0$, the presence of γ in the off diagonal, the exact position depending on the value of y, removes the guarantee of a resulting submodular minimization problem.

3.3.2 Non-submodularity of the LAI

It may appear at first glance that the structure of this loss function is aligned with that of the inference, and that we can therefore jointly optimize the loss augmented inference with a single graph cuts procedure. Indeed, the loss function is isomorphic to a supermodular set function, and the inference is isomorphic to a supermodular set function, both of which can be solved by graph cuts. However, the isomorphisms are not the same. The loss function maps to a set function by considering the set of pixels that are incorrectly labeled, while the inference maps to a set function by considering the set of pixels that are labeled as foreground.

We show in detail how the presence of γ in Equation (3.3.1) removes the submodularity guarantee. Given the loss augmented inference in Equation (3.2.4) with the joint feature function in Equation (3.2.3), as well as our supermodular loss in Equation (3.3.1), we can expand Equation (3.2.4) as:

$$\arg\min_{\tilde{y}} -\langle \begin{pmatrix} w_u \\ w_p \end{pmatrix}, \begin{pmatrix} \sum_{j=1}^{p} \phi_u(x, y^j) \\ \sum_{(k,l)\in\mathcal{E}} \phi_p(x, y^k, y^l) \end{pmatrix} \rangle - \Delta(y, \tilde{y}) \tag{3.3.2}$$

Denote E_u the unary potential, E_p the pairwise potential for this loss augmented inference. Here we try to write the problem as one graph cut problem. First we have the combined unary potential:

$$E_u(y^j) = -\langle w_u, \phi_u(x, y^j)\rangle - [y^j \neq \tilde{y}^j]. \tag{3.3.3}$$

Then we have the combined pairwise potential:

$$E_p(y^k, y^l) = -\begin{pmatrix} w_{00} & w_{01} \\ w_{10} & w_{11} \end{pmatrix} \odot \begin{pmatrix} [y^k = y^l = -1] & [\tilde{y}^k = -1 \wedge \tilde{y}^l = +1] \\ [y^k = +1 \wedge y^l = -1] & [\tilde{y}^k = \tilde{y}^l = +1] \end{pmatrix} \tag{3.3.4}$$

$$- \gamma \times \begin{pmatrix} [(y^k = y^l = -1) \wedge (\tilde{y}^k = \tilde{y}^l = +1)] & [(y^k = -1 \wedge y^l = +1) \wedge (\tilde{y}^k = +1 \wedge \tilde{y}^l = -1)] \\ [(y^k = +1 \wedge y^l = -1) \wedge (\tilde{y}^k = -1 \wedge \tilde{y}^l = +1)] & [(y^k = y^l = +1) \wedge (\tilde{y}^k = \tilde{y}^l = -1)] \end{pmatrix} \tag{3.3.5}$$

where $\odot$ is the Hadamard product. In the second term of $E_p(y^k, y^l)$, the existence of γ removes the guarantee of the submodularity of this minimization problem: $w_{00} + w_{11} \geq w_{01} - w_{10}$ is guaranteed, but we may arbitrarily add or subtract γ from either side of the inequality according to Equation (3.3.5), for different cases of the ground truth edge (y^k, y^l). This is exactly due to the fact that the loss function maps to a set function by considering the set of misperdicted pixels, while the inference maps to a set function by considering the set of pixels that are labeled as foreground, which are two different mappings.

If we apply a single mapping, the potentials of the maximization problem are no longer guaranteed to be supermodular (Figure 3.1). We therefore consider a Lagrangian based splitting method to solve the loss augmented inference problem.

3.4 ADMM Algorithm for LAI

Several Lagrangian based decomposition frameworks have been proposed such as dual decomposition and ADMM [Boyd et al., 2011], with the latter having improved convergence guarantees. We have also observed a substantial improvement in performance using ADMM over dual decomposition in our own experiments. Here we consider a splitting method to optimize the minimization of the negative of Equation (3.2.4), which is equivalent to finding the most violated constraint in cutting plane optimization:

$$\arg\min_{\tilde{y}_a, \tilde{y}_b} -\langle w, \phi(x, \tilde{y}_a)\rangle - \Delta(y, \tilde{y}_b) \quad \text{s.t. } \tilde{y}_a = \tilde{y}_b. \tag{3.4.1}$$

and we form the augmented Lagrangian as

$$\mathcal{L}(\tilde{y}_a, \tilde{y}_b, \lambda) = -\langle w, \phi(x, \tilde{y}_a)\rangle - \Delta(y, \tilde{y}_b) + \lambda^T(\tilde{y}_a - \tilde{y}_b) + \frac{\rho}{2}\|\tilde{y}_a - \tilde{y}_b\|_2^2 \tag{3.4.2}$$

where $\rho > 0$. (3.4.2) can be optimized in an iterative fashion by Algorithm 3, in a scaled form with $u = \frac{1}{\rho}\lambda$.

The saddle point of the Lagrangian will correspond to an optimal solution over a convex domain, while we are optimizing w.r.t. binary variables. Strictly speaking, we may therefore consider the linear programming (LP) relaxation of our loss augmented inference problem, followed by a rounding post-processing step. We use a standard stopping criterion as in [Boyd et al., 2011]: the primal and dual residuals must be small with an absolute criterion $\epsilon^{\text{abs}} = 10^{-4}$ and a relative criterion $\epsilon^{\text{rel}} = 10^{-2}$. In practice, we have found that

Algorithm 3 ADMM in scaled form for finding a saddle point of the Lagrangian in Equation (3.4.2)

```
1: Initialization $u^0 = 0$
2: repeat
3:     $\tilde{y}_a^{t+1} = \arg\min_{\tilde{y}_a} -\langle w, \phi(x, \tilde{y}_a)\rangle + \frac{\varrho}{2}(\|\tilde{y}_a - \tilde{y}_b^t + u^t\|_2^2)$
4:     $\tilde{y}_b^{t+1} = \arg\min_{\tilde{y}_b} -\Delta(y, \tilde{y}_b) + \frac{\varrho}{2}(\|\tilde{y}_a^{t+1} - \tilde{y}_b + u^t\|_2^2)$
5:     $u^{t+1} = u^t + (\tilde{y}_a^{t+1} - \tilde{y}_b^{t+1})$
6:     $t = t + 1$
7: until stopping criterion satisfied
```

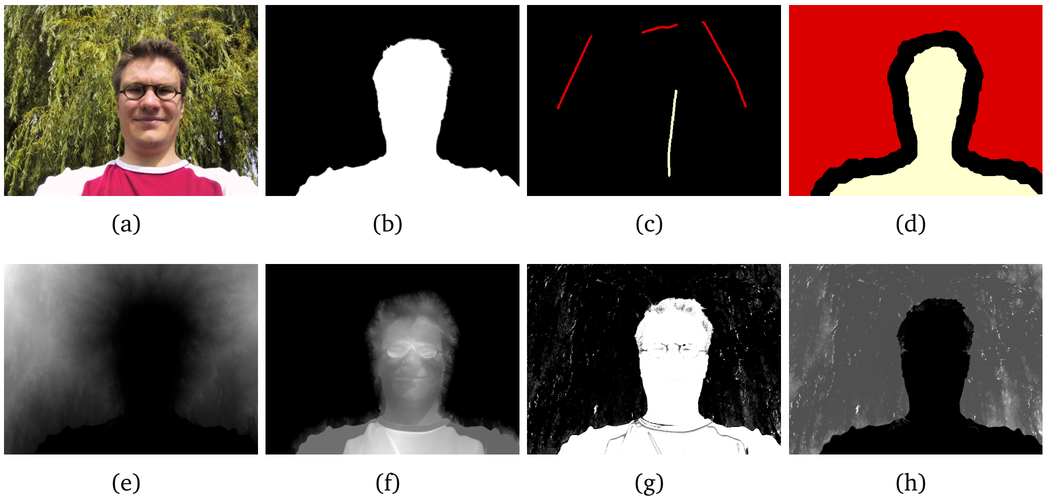


Figure 3.2: Example images from the dataset and the extracted features. 3.2(a) original RGB space image; 3.2(b) groundtruth; 3.2(c) the user-labelled seeds; 3.2(d) the extended seed region; 3.2(e) the distance features to foreground seed based on RGB space; 3.2(f) the distance features to background seed based on RGB space; 3.2(g) the GMM appearance model based on RGB space; 3.2(h) the distance features to foreground seed based on the RGB-space GMM appearance.

discretizing the quadratic terms and incorporating them into the unary potentials of the respective graph cuts problems is more computationally efficient, while yielding results that are nearly identical with exact optimization with a primal-dual gap of 0.01%. We show in the experimental results that this strategy yields results almost identical to those of an LP relaxation, while being much faster in practice.

In general, we simply need task specific solvers for Line 3 and Line 4 of Algorithm 3. These solvers need not use a single graph cut algorithm, and can therefore exploit any available structure even though it may not be present, or aligned, between the two subproblems. Although we have used this framework for the specific supermodular loss function described in the previous subsection, we note that this provides an API for the structured output SVM framework alternate to that provided by SVMstruct [Tsochantaridis et al., 2005].

$\gamma = 0.25$		Eval.		
		Δ(1e3)	0-1(1e3)	IoU
Train.	Δ	$\mathbf{6.3562 \pm 1.065}$	3.3378 ± 0.5462	0.2111 ± 0.0152
	0-1	7.8641 ± 1.0437	4.1548 ± 0.5378	0.2399 ± 0.0170

$\gamma = 0.5$		Eval.		
		Δ(1e3)	0-1(1e3)	IoU
Train.	Δ	$\mathbf{9.0483 \pm 1.3457}$	3.2801 ± 0.4687	0.2079 ± 0.0155
	0-1	11.582 ± 1.5495	4.1548 ± 0.5378	0.2399 ± 0.0170

$\gamma = 1.0$		Eval.		
		Δ(1e3)	0-1(1e3)	IoU
Train.	Δ	$\mathbf{14.908 \pm 2.4102}$	3.4145 ± 0.4108	0.2084 ± 0.0160
	0-1	19.019 ± 2.5613	4.1458 ± 0.5378	0.2399 ± 0.0170

Table 3.1: The cross comparison of average loss values (with standard error) using different loss functions during training and during testing on Grabcut dataset . Training with the same supermodular loss functions as used during testing yields the best results. Training with supermodular losses even outperform the Hamming loss in terms of evaluating by Hamming loss. A Wilcoxon sign rank test shows that training with Δ gives significantly better results in all cases ($p \leq 2 \times 10^{-3}$).

$\gamma = 0.5$		Eval.		
		Δ(1e3)	0-1(1e3)	IoU
Train.	Δ	$\mathbf{2.616 \pm 0.612}$	1.297 ± 0.224	0.169 ± 0.018
	0-1	2.885 ± 0.765	1.393 ± 0.279	0.173 ± 0.019

Table 3.2: The cross comparison of average loss values (with standard error) using different loss functions during training and during testing on IBSR dataset. (cf. comments for Table 3.1).

3.5 Experimental Results

In this section, we consider a foreground/background segmentation task. We compare the prediction using our proposed supermodular loss function with the prediction using Hamming loss. We show that: (i) our proposed splitting strategy is orders of magnitude faster than the Fujishige-Wolfe minimum norm point algorithm; (ii) our strategy yields results nearly identical to a LP-relaxation while being much faster in practice; and (iii) training with the same supermodular loss as during test time yields better performance. These results in combination demonstrate the correctness of our proposed strategy.

Datasets We use the dataset provided by [Gulshan et al., 2010; Blake et al., 2004] which contains 151 images in total, including the color images in RGB space, the ground truth foreground/background segmentation and the user-labelled seeds (see Figure 3.2(a), Figure 3.2(b), and Figure 3.2(c), respectively). As we are discriminatively training a class specific segmentation system in our experiments, we focus on the images in which the foreground objects are *people*.

Feature Extraction Following the feature extraction in [Osokin and Kohli, 2014] we enlarge the user-labelled seed region (Figure 3.2(d)) and use them as the foreground and background seeds only during the feature extraction stage. We compute in total 18 unary features: 3 RGB color space channels, 3 CIELUV space channels, the likelihood from Gaussian mixture models in RGB space and CIELUV space independently, and the distance transform features as defined in [Gulshan et al., 2010] using the two color space channels, two GMM models fitted to the enlarged foreground and background seeds independently, and two Euclidean distances fitted to foreground and background seeds. Figure 3.2(e) to Figure 3.2(h) show examples of the extracted features. We use both the software provided by [Gulshan et al., 2010] and our implementation for this feature extraction procedure.

IBSR Dataset We additionally utilise the Internet Brain Segmentation Repository (IBSR) dataset [Rohlfing, 2012], which consists of T1-weighted MR images. Images and masks have been linearly registered and cropped to $145 \times 158 \times 123$. We choose one horizontal slice within each volume and we follow the feature extraction procedure as in [Alchatzidis et al., 2014].

Training and Testing We use the ADMM splitting strategy to solve the minimization problem in Equation (3.4.1). We use the GCMex - MATLAB wrapper for the Boykov-Kolmogorov graph cuts algorithm [Fulkerson et al., 2009; Boykov and Kolmogorov, 2004; Boykov et al., 2001; Kolmogorov and Zabin, 2004] to solve the optimization problems on lines 3 and 4 in Algorithm 3 i.e. for the inference part and for the loss part separately. Results computed with different values of $\gamma > 0$ are shown in Table 3.1 and Table 3.2.

During the training stage, we use $\rho = 0.1$ for the ADMM step-size parameter. The regularization parameter C in Equation (3.2.1) is chosen by cross-validation in the range $\{10^i | -2 \leq i \leq 2\}$. We additionally train and test with Hamming loss as a comparison.

At test time, we have computed the unnormalized Hamming loss, the intersection over union loss (IoU) , and our 8-connected loss for each training scenario. We have performed several random train-test splits in order to compute error bars on the loss estimates. During testing stage, we evaluate one prediction as the average loss value for all images in the testing set. We compare different loss functions during training and during testing and measure the empirical loss values. We repeated 5 time with random splitting sets for training and testing to obtain an estimate of the average performance.

Computation Time We compare the time of one calculation of the loss augmented inference by the ADMM algorithm and by the minimum norm point algorithm [Fujishige, 2005] (MinNorm). For MinNorm, we use the implementation provided in the SFO toolbox [Krause, 2010]. Although it has been proven that in t iterations, the MinNorm returns an $O(1/t)$-approximate solution [Chakrabarty et al., 2014], the first step of this algorithm is to find a point in the submodular polytope, which alone is computationally intractable even for small 600×400 pixel images. Therefore, we measure the computation time on downsampled images, showing empirically the growth in computation as a function of image size. The running times are recorded on a machine with a 3.20GHz CPU. Similarly, a dual-decomposition baseline took orders of magnitude longer computation than the ADMM approach, following known convergence results [Boyd et al., 2011].

Results As shown in Table 3.1, training with the same supermodular loss as used for testing has achieved the best performance. Training with the supermodular loss even outperforms training with Hamming loss when measured by Hamming loss on the test set, with a reduction in error of 17.2%. We have additionally tried training with a joint graph cuts loss augmented inference using the pairwise potentials illustrated in Figure 3.1. However, due to the non-submodular potentials, the graph cuts procedure does not correctly minimize the energy resulting in incorrect cutting planes that causes optimization to fail after a small number of iterations. The performance of this system was effectively random, and we have chosen not to include these accuracy values in Table 3.1.

Qualitative segmentation results are shown in Figure 3.5 and followed by Figure 3.8, Figure 3.9 and Figure 3.10. We show also a pixelwise comparison of the prediction using 8-connected loss and using Hamming loss in Figure 3.6 and Figure 3.7. The 8-connected loss achieves better performance on the foreground/background boundary, as well as on elongated structures of the foreground object, such as the head and legs, especially when the appearance of the foreground is similar to the background.

We measure the computation time for 120 calculations of the loss augmented inference by ADMM and MinNorm on different sized images. From Figure 3.3 we can see that ADMM is always faster than the MinNorm by a substantial margin, and around 100 times faster when the problem size reaches 10^3. Figure 3.4 shows that the computing time for both ADMM and MinNorm vary linearly in log-log scale, while MinNorm has a substantially higher slope, suggesting a worse big-$\mathcal{O}$ computational complexity. We note that theoretical bounds on MinNorm are currently weak and the exact complexity is unknown [Chakrabarty et al., 2014]. Although it is immediately clear from Figures 3.3 and 3.4 that ADMM is substantially faster than the minimum norm point algorithm, we have performed Wilcoxon sign rank tests that show this difference is significant with $p < 10^{-20}$ in all settings.

We additionally ran a baseline comparing non-submodular loss augmented inference with the QPBO approach [Rother et al., 2007]. We computed pairwise energies as in Figure 3.1(a). QPBO found loss augmented energies across the dataset of $1.1 \times 10^6 \pm 3 \times 10^5$ while ADMM found loss augmented energies of $3.7 \times 10^6 \pm 8 \times 10^5$, a substantial improvement.

Comparison to LP-relaxation We also compare ADMM to an LP relaxation procedure for the loss augmented inference to determine the accuracy of our optimization in practice. For the implementation of the LP relaxation, we use the `UGM` toolbox [Schmidt, 2012]. We show in Table 3.3 the comparison between using ADMM and the LP relaxation. The first column represents the energy achieved by the loss augmented inference (Equation (3.2.4)). We observe that the (maximal) energy achieved by ADMM is almost the same as the LP relaxation: a difference of 0.4%. Columns 2–4 show the computing time for one calculation of the loss augmented inference on the downsampled images. Using a LP relaxation, the computation time is orders of magnitude slower, growing as a function of the image size. ADMM provides a more efficient strategy without loss of performance.

	$-E$	size $= 600$	size $= 1200$	size $= 2400$
ADMM	2.28 ± 0.58	0.035 ± 0.002	0.051 ± 0.002	0.864 ± 0.476
LP	2.29 ± 0.57	1.857 ± 0.128	3.946 ± 0.286	13.57 ± 1.359

Table 3.3: The comparison between ADMM and an LP relaxation for solving the loss augmented inference. The 1st column shows the optimal energy values (10^3) (Equation (3.2.4)); columns 2–4 show the computation time (s) for one calculation on downsampled images of varying size.

Square Loss In order to validate the generality, we additionally utilize another supermodular loss function as follows:

$$\Delta(y, \tilde{y}) = \left(\frac{\sum_{j=1}^{p} [y^j \neq \tilde{y}^j]}{\alpha} \right)^2 \tag{3.5.1}$$

where $\alpha > 0$ is a scale factor to prevent the value to be too large in an image scale problem. We used $\alpha = 20$ in our setting. This is a function on the misprediction set which only depends on the size of the input set. As the square function is a convex function, Δ is a supermodular loss w.r.t. the misprediction set. Then maximizing the loss itself is a supermodular maximization i.e. a submodular minimization problem, we use the `SFO` toolbox to solve it.

We can see that training with the same supermodular loss while during test time yields better performance, which validates the correctness of the ADMM splitting strategy with another loss/inference combination.

		Eval.	
		Δ	0-1
Train.	Δ	0.869 ± 0.049	8.056 ± 0.332
	0-1	0.872 ± 0.057	8.000 ± 0.404

Table 3.4: The cross comparison of average loss values (with standard error) using the supermodular loss and Hamming loss (labeled 0-1) during training and test time).

3.6 Discussion

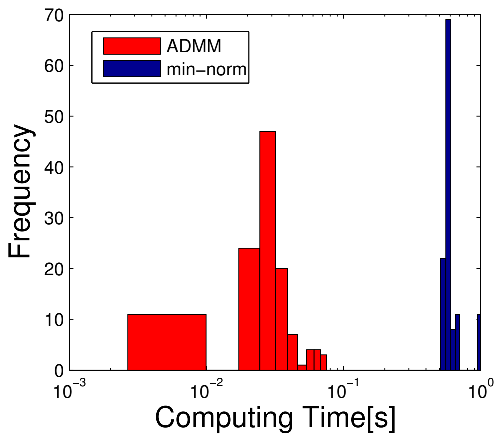
A somewhat surprising result in Table 3.1 is that training with the supermodular loss results in better performance *as measured by Hamming loss*. This has been previously observed with a different loss function by [Pletscher and Kohli, 2012; Osokin and Kohli, 2014], and indicates that in the finite sample regime a supermodular likelihood can result in better generalization performance. This holds, although the model space and regularizer were identical in both training settings. We believe that further exploration of the properties of supermodular loss functions is warranted in this regard.

Our results in terms of computation time give clear evidence for the superiority of ADMM inference when a specialized optimization procedure is available for the loss function. As shown in Figures 3.3 and 3.4, the Fujishige-Wolfe minimum norm point algorithm does not scale to typical consumer images (i.e. several megapixels), which indicates that loss functions for which a specialized optimization procedure is not available are likely infeasible for pixel level image segmentation without unprecedented improvements in general submodular minimization. Figure 3.4 shows that the log-log slope of the runtime for the min-norm point algorithm is higher than for ADMM, suggesting a worse computational compexity, at least for the loss function considered here. One may wish to employ the result that early termination of the min-norm point algorithm gives a guaranteed approximation of the exact result, but even this is infeasible for images of the size considered here. In addition, Table 3.3 suggests that ADMM provides a more efficient strategy without loss of performance compared to using an LP-relaxation. Joint graph-cuts optimization for loss augmented inference results in non-submodular pairwise potentials and graph-cuts fails to correctly minimize the joint energy. As a result, a cutting plane optimization of the structured output SVM objective fails catastrophically, and the resulting accuracy is on par with a random weight vector.

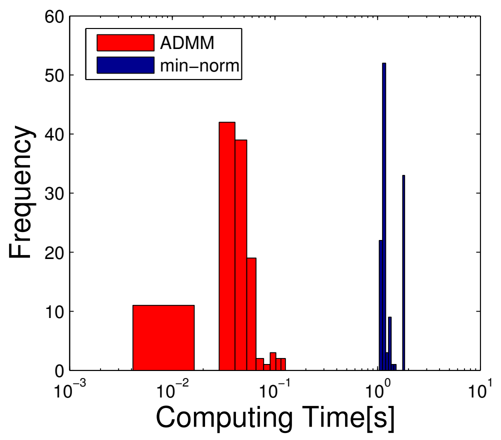
In this work, we have shown that a supermodular loss function achieves improved performance both in qualitative and quantitative terms on a binary segmentation task. We observe that a key advantage of the proposed supermodular loss over modular losses, e.g. Hamming loss, is an improved ability to find elongated regions such as heads and legs, or thin articulated structures in medical images .

Previous to our work, specialized inference procedures had to be developed for every model/loss pair, a time consuming process. By contrast, we have proposed a Lagrangian splitting technique based on ADMM to perform general loss augmented inference. We demonstrate the feasibility of the ADMM algorithm for loss augmented inference on an interactive foreground/background segmentation task, for which alternate strategies such as the Fujishige-Wolfe minimum norm point algorithm are infeasible.

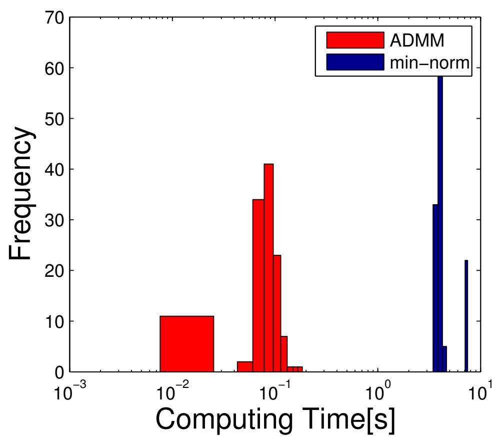
Our proposed ADMM algorithm provides a strategy to solve the loss augmented inference as two separate subproblems. This provides an alternate API for the structured output SVM framework to that of SVMstruct [Tsochantaridis et al., 2005]. We envision that this can be of use in a wide range of application settings, and an open source general purpose toolbox for this efficient segmentation framework with supermodular losses is available for download from `https://github.com/yjq8812/efficientSegmentation`



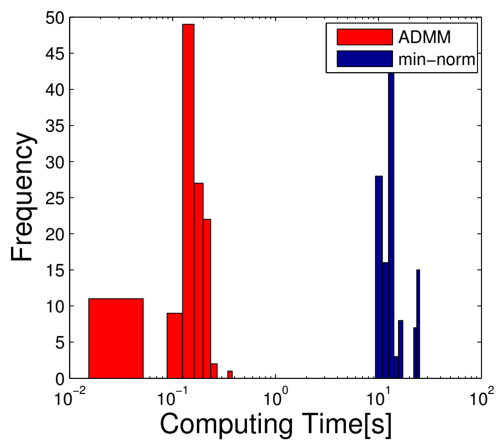
(a) Size=600



(b) Size=1200



(c) Size=2400



(d) Size=4800

Figure 3.3: The computing time for the loss augmented inference, on different problem sizes. The red histograms stands for ADMM and the blue for MinNorm. As we can see, the calculation by ADMM is always faster than by MinNorm, and there is no overlap between the computing time by the two methods.

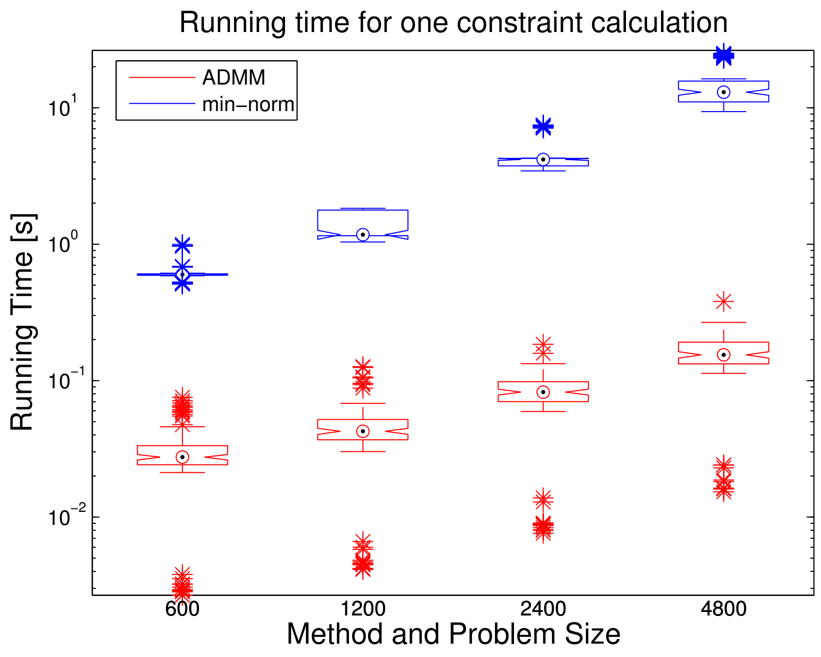
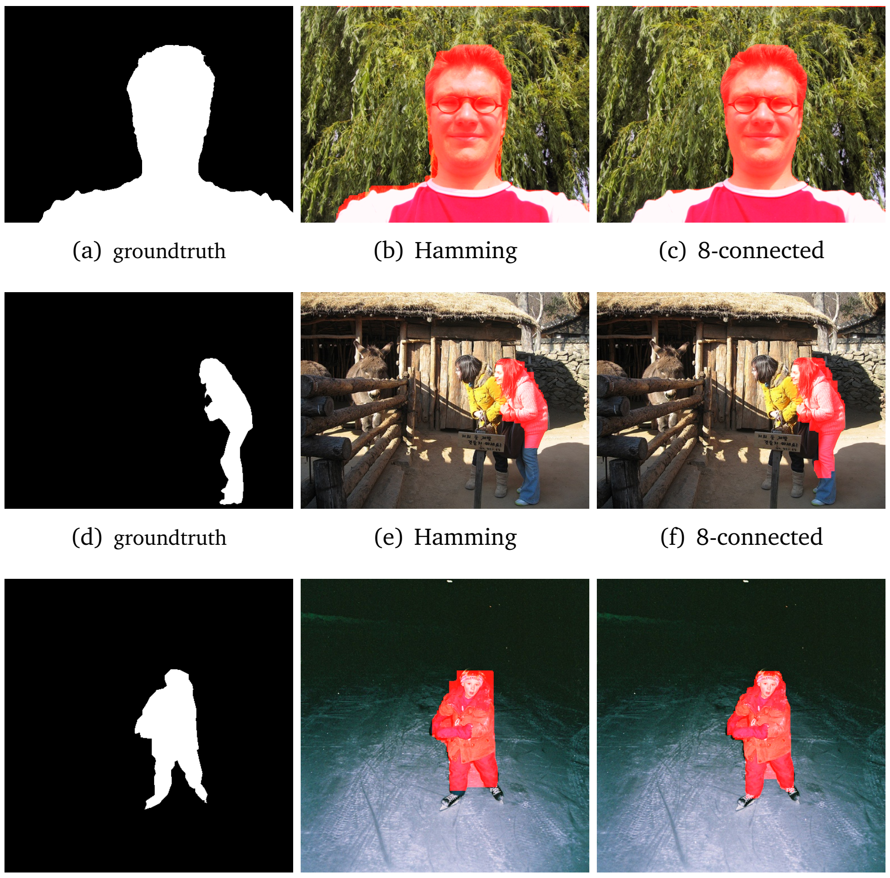


Figure 3.4: The running time increase along with the problem size. Both algorithm increase linearly in log scale while the ADMM has a time reduction from 10 times to 10^2 times along with the increase of the problem size.



(a) groundtruth (b) Hamming (c) 8-connected

(d) groundtruth (e) Hamming (f) 8-connected

(g) groundtruth (h) Hamming (i) 8-connected

Figure 3.5: The segmentation results of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs.

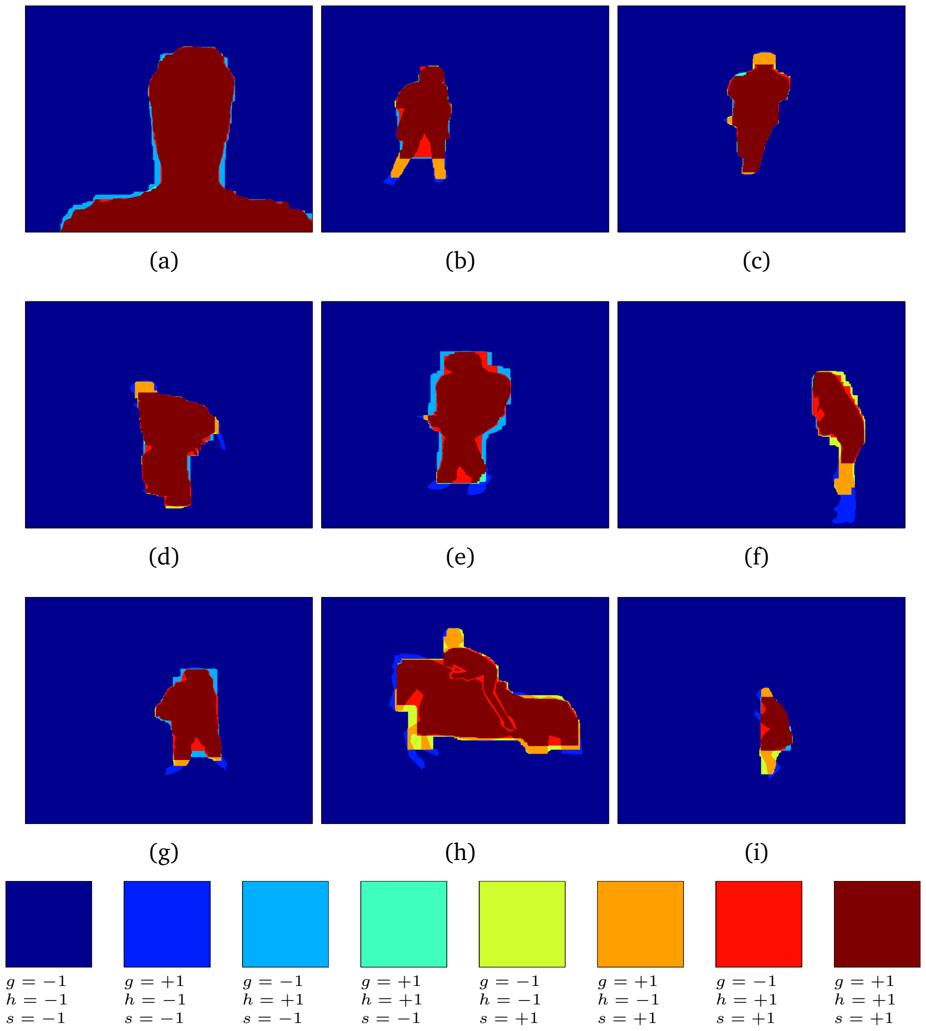


Figure 3.6: A pixelwise comparison, in the semantic segmentation task [Gulshan et al., 2010], of the ground truth (denoted g in the legend), the prediction from training with Hamming loss (denoted h), and the prediction when training with the proposed supermodular loss (denoted s). We note that there are many regions in the set of images where the supermodular loss learns to correctly predict the foreground when Hamming loss fails (orange regions corresponding to $g=+1$, $h=-1$, and $s=+1$).

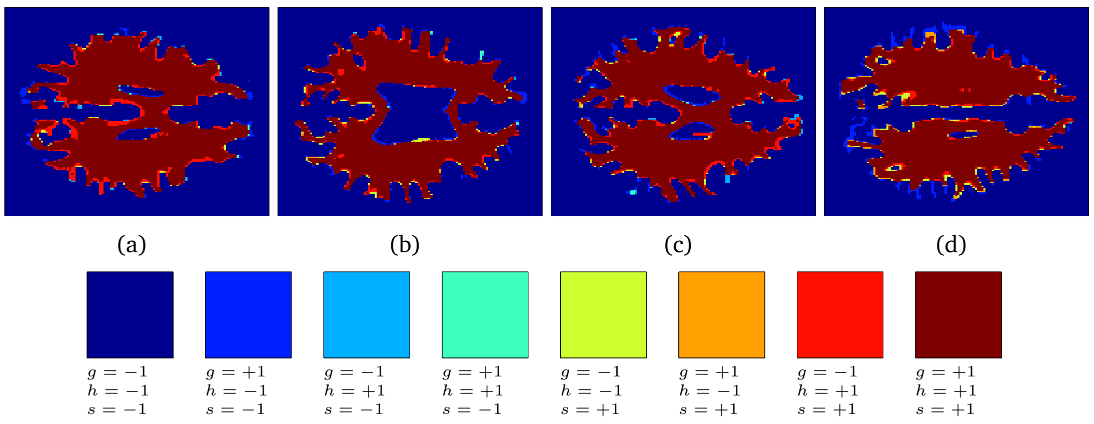
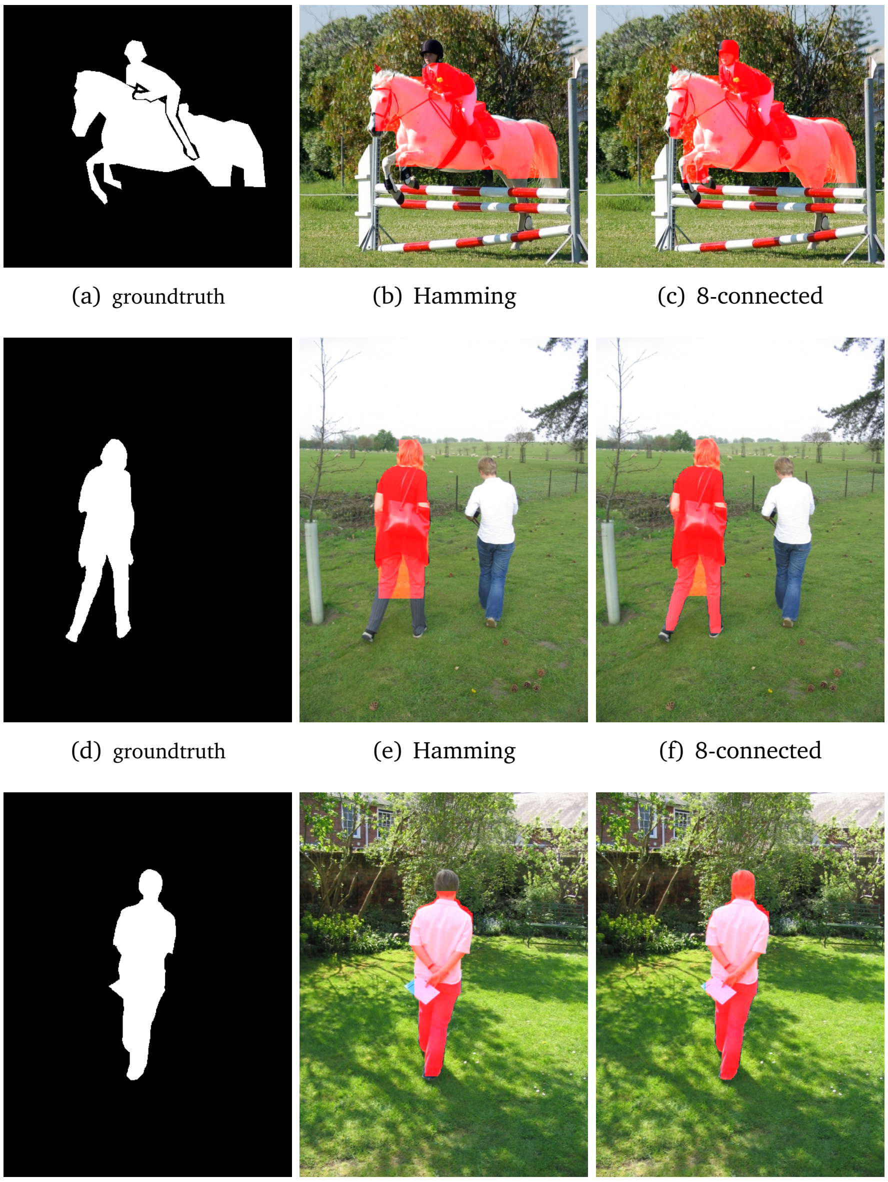


Figure 3.7: A pixelwise comparison, in the structural brain segmentation task [Rohlfing, 2012], of the ground truth (denoted g in the legend), the prediction from training with Hamming loss (denoted h), and the prediction when training with the proposed supermodular loss (denoted s). We note that there are many regions in the set of images where the supermodular loss learns to correctly predict the foreground when Hamming loss fails (orange regions corresponding to $g = +1$, $h = -1$, and $s = +1$).

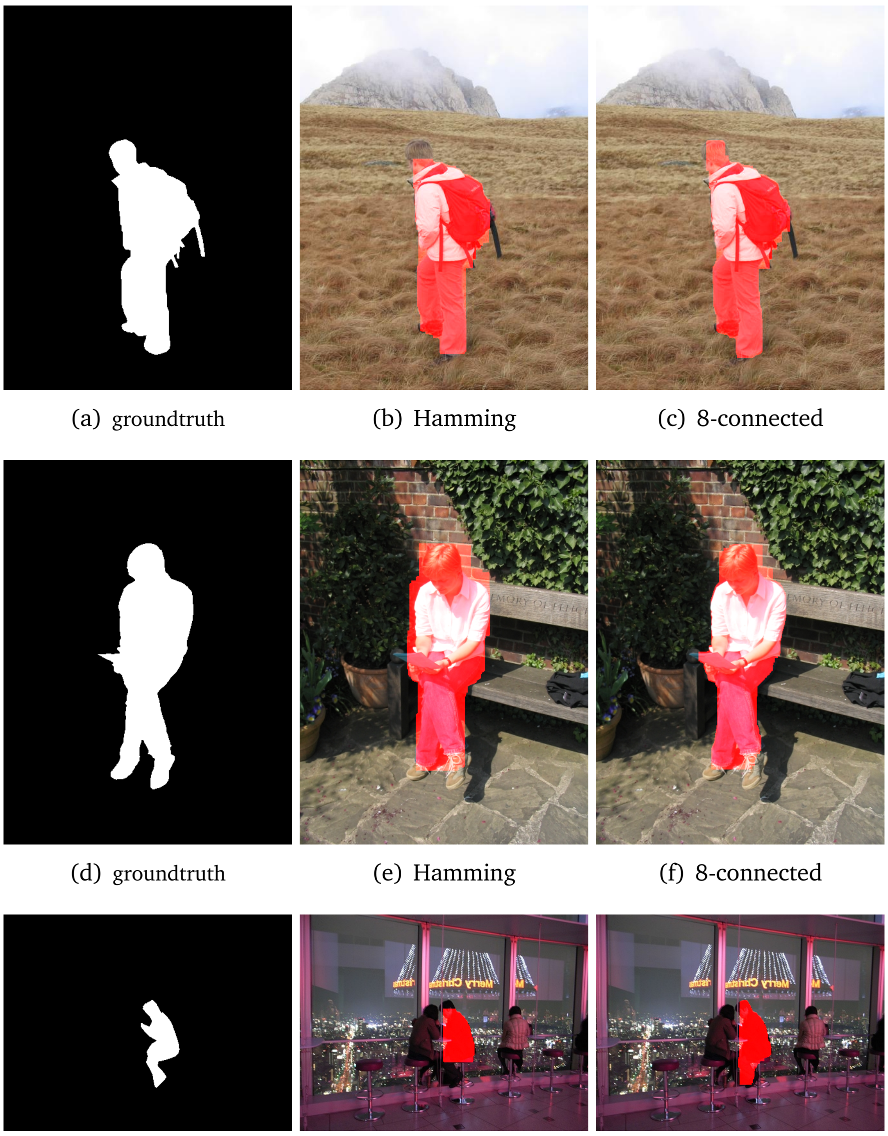


(a) groundtruth (b) Hamming (c) 8-connected

(d) groundtruth (e) Hamming (f) 8-connected

(g) groundtruth (h) Hamming (i) 8-connected

Figure 3.8: The segmentation results (continued) of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs.

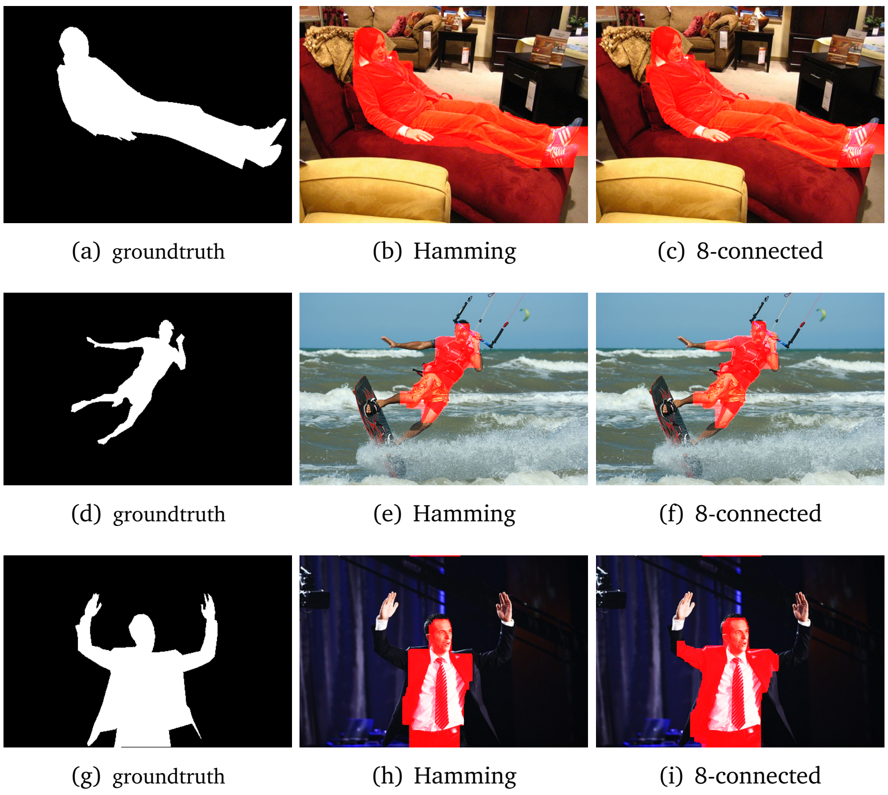


(a) groundtruth (b) Hamming (c) 8-connected

(d) groundtruth (e) Hamming (f) 8-connected

(g) groundtruth (h) Hamming (i) 8-connected

Figure 3.9: The segmentation results (continued) of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs.


(a) groundtruth (b) Hamming (c) 8-connected

(d) groundtruth (e) Hamming (f) 8-connected

(g) groundtruth (h) Hamming (i) 8-connected

Figure 3.10: The segmentation results (continued) of prediction by trained with Hamming loss (column 2 and 5) and our supermodular loss (column 4 and 6). The supermodular loss perform better on foreground object boundary than Hamming loss does, as well as it achieves better prediction on the elongated structure of the foreground object e.g. the heads and the legs.

Chapter 4

Lovász Hinge: Learning with Submodular Loss Functions

Contents

4.1 Introduction 52
4.2 Submodular Loss Functions 53
4.3 Existing Convex Surrogates 54
4.3.1 Extension 56
4.3.2 Slack Rescaling 57
4.3.3 Margin Rescaling 58
4.3.4 Complexity of Subgradient Computation 59
4.4 Lovász Hinge 60
4.4.1 Lovász Hinge 60
4.4.2 Complexity of Subgradient omputation 63
4.4.3 Visualization of Convex Surrogates 63
4.5 Hardness Result for Learning Multiple Correct Results 69
4.5.1 Learning 70
4.5.2 Supermodular losses and the Structured Output SVM 70
4.5.3 Submodular losses and the Lovász Hinge 71
4.5.4 The Inference Problem 72
4.6 Jaccard Loss 72
4.7 Experimental Results 74
4.7.1 A Synthetic Problem 74
4.7.2 PASCAL VOC Image Classification 75
4.7.3 PASCAL VOC Multilabel Prediction 77
4.7.4 MS COCO Multilabel Prediction 78
4.7.5 Empirical Results 80
4.8 Discussion 80

Overview Learning with non-modular losses is an important problem when sets of predictions are made simultaneously. The main tools for constructing convex surrogate loss functions for set prediction are margin rescaling and slack rescaling. In this chapter, we show that these strategies lead to tight convex surrogates if and only if the underlying loss function is increasing in the number of incorrect predictions. However, gradient or cutting-plane computation for these functions is NP-hard for non-supermodular loss functions. We propose instead a novel surrogate loss function for submodular losses, the Lovász hinge, which leads to $\mathcal{O}(p \log p)$ complexity with $\mathcal{O}(p)$ oracle accesses to the loss function to compute a gradient or cutting-plane. We prove that the Lovász hinge is convex and yields an extension. As a result, we have developed the first tractable convex surrogates in the literature for submodular losses. We demonstrate the utility of this novel convex surrogate through several set prediction tasks, including on the PASCAL VOC and Microsoft COCO datasets.

The work presented in this chapter is based on:

- Jiaqian Yu and Matthew B. Blaschko. The Lovász hinge: A convex surrogate for submodular losses. 2015. arXiv:1512.07797;
- Matthew B. Blaschko and Jiaqian Yu. Hardness results for structured learning and inference with multiple correct outputs. In Constructive Machine Learning Workshop at ICML, Lille, France, July 2015;
- Jiaqian Yu and Matthew B. Blaschko. Learning submodular losses with the lovász hinge. In Proceedings of the 32nd International Conference on Machine Learning, volume 37, pages 1623–1631, 2015

4.1 Introduction

In this chapter, we aim to provide a theoretical and algorithmic foundation for a novel class of learning algorithms that make feasible learning with submodular losses, an important subclass of non-modular losses that is currently infeasible with existing algorithms.

Convex surrogate loss functions are central to the practical application of empirical risk minimization. Straightforward principles have been developed for the design of convex surrogates for binary classification and regression [Bartlett et al., 2006], and in the structured output setting margin and slack rescaling are two principles for defining convex surrogates for more general output spaces [Tsochantaridis et al., 2005]. Despite the apparent flexibility of margin and slack rescaling in their ability to bound arbitrary loss functions, there are fundamental limitations to our ability to apply these methods in practice: (i) they provide only loose upper bounds to certain loss functions (cf. Propositions 4.1 & 4.2), (ii) computing a gradient or cutting plane is NP-hard for submodular loss functions, and (iii) consistency results are lacking in general [McAllester, 2007; Tewari and Bartlett, 2007]. In practice, modular losses, such as Hamming loss, are often applied to maintain tractability, although non-modular losses, such as the Jaccard loss have been applied in the structured prediction setting [Blaschko and Lampert, 2008; Nowozin, 2014]. We show in this paper that the Jaccard loss is in fact a submodular loss, and our proposed convex surrogate provides a polynomial-time tight upper bound.

In this work, we introduce an alternate principle to construct convex surrogate loss functions for submodular losses based on the Lovász extension of a set function. The Lovász extension of a submodular function is its convex closure, and has been used in other machine learning contexts e.g. [Bach, 2010; Iyer and Bilmes, 2013]. We analyze the settings in which margin and slack rescaling are tight convex surrogates by finding necessary and sufficient conditions for the surrogate function to be an extension of a set function. Although margin and slack rescaling generate extensions of *some* submodular set functions, their optimization is NP-hard. We therefore propose a novel convex surrogate for submodular functions based on the Lovász extension, which we call the Lovász hinge. In contrast to margin and slack rescaling, the Lovász hinge provides a tight convex surrogate to *all* submodular loss functions, and computation of a gradient or cutting plane can be achieved in $\mathcal{O}(p \log p)$ time with a linear number of oracle accesses to the loss function. We demonstrate empirically fast convergence of a cutting plane optimization strategy applied to the Lovász hinge, and show that optimization of a submodular loss results in lower average loss on the test set.

In Section 4.2 we introduce the notion of a submodular loss function in the context of empirical risk minimization. The Structured Output SVM is one of the most popular objectives for empirical risk minimization of interdependent outputs, and we demonstrate its properties on non-modular loss functions in Section 4.3. In Section 4.4 we introduce the Lovász hinge as well as properties involving the convexity and computational complexity. We empirically demonstrate its performance first on a synthetic problem, then on image classification and labeling tasks on Pascal VOC dataset and the Microsoft COCO dataset in Section 4.7.

4.2 Submodular Loss Functions

As reminder, in empirical risk minimization, we approximate the risk, $\mathcal{R}$ of a prediction function $\hat{h} : \mathcal{X} \mapsto \mathcal{Y}$ by an empirical sum over losses incurred on a finite sample, using e.g. an i.i.d. sampling assumption [Vapnik, 1995]:

$$\hat{\mathcal{R}}(\hat{h}) := \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i, \hat{h}(x_i)) \tag{4.2.1}$$

Central to the practical application of the empirical risk minimization principle, one must approximate, or upper bound the discrete loss function Δ with a convex surrogate. We will identify the creation of a convex surrogate for a specific loss function with an operator that maps a function with a discrete domain to one with a continuous domain. In particular, we will study the case that the discrete domain is a set of p binary predictions. In this case we denote

$$\mathcal{Y} = \{-1, +1\}^p \tag{4.2.2}$$

$$\Delta : \{-1, +1\}^p \times \{-1, +1\}^p \mapsto \mathbb{R}_+ \tag{4.2.3}$$

$$\mathbf{B}\Delta : \{-1, +1\}^p \times \mathbb{R}^p \mapsto \mathbb{R} \tag{4.2.4}$$

where $\mathbf{B}$ is an operator that constructs the surrogate loss function from Δ, and we assume

$$\hat{h}(x) = \operatorname{sign}(g(x)) \tag{4.2.5}$$

where $g : \mathcal{X} \mapsto \mathbb{R}^p$ is a parametrized prediction function to be optimized by empirical risk minimization. We always consider that Δ is isomorphic to a set function on the misprediction set:

$$\Delta(y^*, \tilde{y}) = \ell(\{j | y^j \neq \tilde{y}^j\}).$$

In the sequel when we say Δ is increasing we mean it is increasing w.r.t. the mispredicíton set $\{j|y^j \neq \tilde{y}^j\}$.

In particular in this chapter, we will focus on the function to be submodular, namely

$$\forall A, B \subseteq V,\ A \subseteq B\ v \in V \setminus B,$$
$$\ell(A \cup \{v\}) - \ell(A) \geq \ell(B \cup \{v\}) - \ell(B)$$

by Definition 2.1.

Such functions are typically increasing, though it is possible to conceive of a sensible loss function that may be non-increasing (e.g. when getting 50% recall is worse than making no prediction at all, as in the identification of cancer tissue).

A key property is the relationship between $\mathbf{B}\Delta$ and Δ. In particular, we are interested in when a given surrogate strategy $\mathbf{B}\Delta$ yields an *extension* of Δ (cf. Definition 4.1). We make this notion formal by identifying $\{-1, +1\}^p$ with a given p-dimensional unit hypercube of $\mathbb{R}^p$. We say that $\mathbf{B}\Delta(y, \cdot)$ is an extension of $\Delta(y, \cdot)$ if and only if the functions are equal over the vertices of this unit hypercube. We focus on function extensions as they ensure a tight relationship between the discrete loss and the convex surrogate.

With these notions, we now turn to an analysis of margin and slack rescaling, and show necessary and sufficient conditions for these operators to yield an extension to the underlying discrete loss function.

4.3 Existing Convex Surrogates

In this section, we analyze two existing convex surrogates namely margin rescaling and slack rescaling. We determine necessary and sufficient conditions for margin and slack rescaling to yield an extension of the underlying set loss function, and address their shortcoming in the complexity of subgradient computation.

The Structured Output SVM (SOSVM) is a popular framework in the regularized risk minimization framework [Taskar et al., 2004; Tsochantaridis et al., 2005] that aims at learning a parametrized function $h : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ over input/output pairs from which a prediction can be derived by maximizing

$$\hat{h}(x) = \arg\max_{y \in \mathcal{Y}} h(x, y; w) \tag{4.3.1}$$

The SOSVM framework assumes h to be represented by an inner product between an element of a reproducing kernel Hilbert space and some combined feature representation of inputs and outputs $\phi(x, y)$,

$$h(x, y; w) = \langle w, \phi(x, y) \rangle \tag{4.3.2}$$

although the notions of margin and slack rescaling may be applied to other function spaces, including random forests [Criminisi and Shotton, 2013] and deep networks [Bengio, 2009].

A bounded loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ quantifies the loss associated with a prediction $\tilde{y}$ while the true value is y, and is used to re-scale the constraints. As for reminder, followings are with margin-rescaling constraints and slack-rescaling constraints:

$$\min_{w,\xi} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \xi_i, \tag{4.3.3}$$

$$\text{s.t. } \forall i, \forall \tilde{y} \in \mathcal{Y} : \; \langle w, \phi(x_i, y_i) \rangle - \langle w, \phi(x_i, \tilde{y}) \rangle \geq \Delta(y_i, \tilde{y}) - \xi_i \tag{4.3.4}$$

$$\text{or } \Delta(y_i, \tilde{y}) \left(\langle w, \phi(x_i, y_i) \rangle - \langle w, \phi(x_i, \tilde{y}) \rangle \right) \geq \Delta(y, \tilde{y}) - \xi_i \tag{4.3.5}$$

respectively. A cutting plane algorithm is commonly used to solve it and one version of the approach with slack rescaling is shown in Algorithm 1 and in Algorithm 2 (cf. Chapter 2, Section 2.4)

In the sequel, we will consider the case that each $x_i \in \mathcal{X}$ is an ordered set of p elements, and that $y_i \in \mathcal{Y}$ is a binary vector in $\{-1, +1\}^p$. We consider feature maps such that

$$\langle w, \phi(x, y) \rangle = \sum_{j=1}^{p} \langle w^j, x^j \rangle y^j. \tag{4.3.6}$$

Given this family of joint feature maps, we may identify the jth dimension of g (cf. Equation (4.2.5)) with

$$g^j(x) := \langle w^j, x^j \rangle. \tag{4.3.7}$$

Therefore $\arg\max_{y \in \{-1;+1\}^p} h(x, y; w) = \text{sign}(g(x))$ and

$$h(x, y) := \langle g(x), y \rangle. \tag{4.3.8}$$

While this section is developed with a linearly parametrized function, the resulting surrogates in this work provide valid subgradients with respect to more general (non-linear) functions. The treatment of optimization with respect to these more general function classes is left to future work.

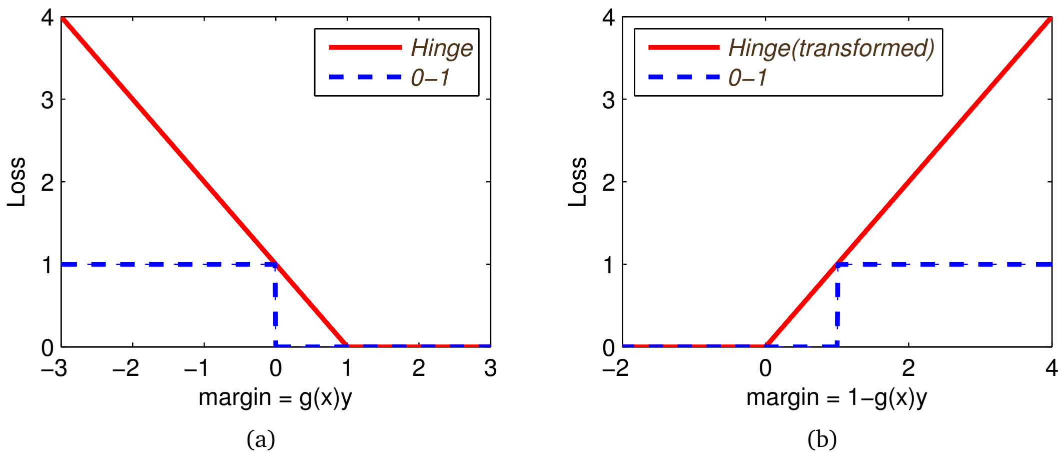


Figure 4.1: Figure 4.1(a) shows the hinge loss as a function of $g^j(x)y^j$ and Figure 4.1(b) shows the transformed mapping as a function of $1 - g^j(x)y^j$

4.3.1 Extension

In order to analyse whether an operator $\mathbf{B}$ yields extensions of Δ, we construct a mapping to a p-dimensional vector space using following definition:

Definition 4.1. *A convex surrogate function* $\mathbf{B}\Delta$ *is an extension if for all* $y \in \mathcal{Y}$,

$$\mathbf{B}\Delta(y, g(x)) = \Delta(y, \operatorname{sign}(g(x))) \tag{4.3.9}$$

on the vertices ($[u]^j \in \{0,1\}$) *of the 0-1 unit cube under the mapping to* $\mathbb{R}^p$ *(cf. Figure 4.2):*

$$j = \{1, \dots, p\}, \quad [u]^j = 1 - g^j(x)y^j. \tag{4.3.10}$$

This mapping is inspired by the hinge loss mapping, whereas it aims at putting the point of zero-loss the at the origin and Δ at the vertex of the unitcube(cf. Figure. 4.1). We note that this definition is a natural generalization of the usual notion of convex surrogates for binary prediction as tight upper bounds on the 0-1 step loss (cf. [Hastie et al., 2009, Figure 10.4]).

In the sequel, we will use the notation for $l : \mathcal{P}(V) \mapsto \mathbb{R}$ as in Equation (3.2.5). Note that at the vertices, $\Delta(y, \cdot)$ has the following values:

$$\ell(\emptyset) \text{ at } \mathbf{0}_p \tag{4.3.11}$$

$$\ell(\mathbf{I}) \text{ at } \{v | v \in \{0,1\}^p, v_i = 1 \Leftrightarrow i \in \mathbf{I}\} \tag{4.3.12}$$

We call (4.3.12) the value of ℓ at *the vertex* $\mathbf{I}$.

We denote the operators for margin and slack rescaling that map the loss function to its convex surrogates $\mathbf{M}$ and $\mathbf{S}$, respectively. These operators have the same signature as $\mathbf{B}$ in

Equation (4.2.4).

$$\mathbf{M}\Delta(y, g(x)) := \max_{\tilde{y}\in\mathcal{Y}} \Delta(y, \tilde{y}) + \langle g(x), \tilde{y}\rangle - \langle g(x), y\rangle \tag{4.3.13}$$

$$\mathbf{S}\Delta(y, g(x)) := \max_{\tilde{y}\in\mathcal{Y}} \Delta(y, \tilde{y})\big(1 + \langle g(x), \tilde{y}\rangle - \langle g(x), y\rangle\big) \tag{4.3.14}$$

respectively.

4.3.2 Slack Rescaling

Proposition 4.1. $\mathbf{S}\Delta(y, \cdot)$ *is an extension of a set function* $\Delta(y, \cdot)$ *iff* $\Delta(y, \cdot)$ *is an increasing function.*

Proof. First we demonstrate the necessity.

Given $\mathbf{S}\Delta(y, \cdot)$ an extension of $\Delta(y, \cdot)$, we will study on whether $\Delta(y, \cdot)$ is an increasing function. More explicitly, there are two cases to determine the values of $\mathbf{S}\Delta(y, \cdot)$.

First case when $u = \mathbf{0}$ (Equation (4.3.10)) $\Delta(y, g(x)) = \ell(\emptyset)$ according to Equation (3.2.5). Given $\mathbf{S}\Delta(y, g(x))$ yields an extension, by Definition 4.1, $\mathbf{S}\Delta(y, g(x)) = \Delta(y, \text{sign}(g(x)))$ at the vertices as in Equation (4.3.11) and Equation (4.3.12). Note that it is trivial when $u = \mathbf{0}$ so the first case is always true for arbitrary (including increasing) ℓ.

Second case when $u \in \mathbb{R}^p \setminus \{\mathbf{0}\}$, let $\mathbf{I} = \{i | u^i \neq 0\}$, then according to Equation (4.3.14), $\mathbf{S}\Delta(y, g(x))$ takes the value of the following equation:

$$\max_{\mathbf{I}\in\mathcal{P}(V)} \ell(\mathbf{I})\big(1 - \sum_{i\in\mathbf{I}} g^i(x) y^i\big). \tag{4.3.15}$$

Considering the second case when $\mathbf{S}\Delta(y, g(x))$ is equal to Equation (4.3.15), denote

$$\mathbf{I}_2 := \arg\max_{\mathbf{I}\in\mathcal{P}(V)} \ell(\mathbf{I})\big(1 - \sum_{i\in\mathbf{I}} g^i(x) y^i\big). \tag{4.3.16}$$

As $\mathbf{S}\Delta(y, g(x))$ is an extension, $\mathbf{S}\Delta(y, g(x)) = \Delta(y, \text{sign}(g(x)))$ at the vertex $\mathbf{I}_2$ which gives us:

$$\forall \mathbf{I}_1 \in \mathcal{P}(V) \setminus \{\emptyset\},\ \ell(\mathbf{I}_2) \geq \ell(\mathbf{I}_1)\,(1 - |(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1|) \tag{4.3.17}$$

This leads to two cases,

1. if $|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1| = 0$, namely $(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1 = \emptyset$, which implies $\mathbf{I}_1 \subseteq \mathbf{I}_2$, then from Equation (4.3.17) we get $\ell(\mathbf{I}_2) \geq \ell(\mathbf{I}_1)$. This implies that ℓ is increasing i.e. Δ are increasing;

2. if $|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1| \geq 1$, this means the rhs of Equation (4.3.17) is non-positive, then it is redundant with $\ell(\mathbf{I}_2) \geq 0$ which is always true.

To conclude, given $\mathbf{S}\Delta(y,\cdot)$ is an extension of a set function $\Delta(y,\cdot)$, it is always the case that Δ is increasing.

To demonstrate the sufficiency, we need to show Equation (4.3.17) is always true if ℓ is increasing.

First, if we have $|(V\setminus \mathbf{I}_2)\cap \mathbf{I}_1| = 0$ which implies $(V\setminus \mathbf{I}_2)\cap \mathbf{I}_1 = \emptyset$ then $\mathbf{I}_1 \subseteq \mathbf{I}_2$, thus $\ell(\mathbf{I}_2) \geq \ell(\mathbf{I}_1)$ given ℓ is increasing.

Then if we have $|(V\setminus \mathbf{I}_2)\cap \mathbf{I}_1| \geq 1$, (4.3.17) always holds even for arbitrary ℓ.

By conclusion, we have that $\mathbf{S}\Delta(y,g(x)) = \Delta(y,\operatorname{sign}(g(x)))$ at the vertices when Equation (4.3.12) holds. As for the case of Equation (4.3.11), it is trivial as $u = \mathbf{0}$. So $\mathbf{S}\Delta(y,\cdot)$ yields a extension of $\Delta(y,\cdot)$ if $\Delta(y,\cdot)$ is increasing. □

4.3.3 Margin Rescaling

It is a necessary, but not sufficient condition that $\Delta(y,\tilde{y})$ be increasing for margin rescaling to yield an extension. However, we note that for all increasing $\Delta(y,\tilde{y})$ there exists a positive scaling $\gamma\in\mathbb{R}$ such that margin rescaling yields an extension. This is an important result for regularized risk minimization as we may simply rescale Δ to guarantee that margin rescaling yields an extension, and simultaneously scale the regularization parameter such that the relative contribution of the regularizer and loss is unchanged at the vertices of the unit cube.

Proposition 4.2. *For all increasing set functions ℓ such that $\exists y$ for which $\mathbf{M}\Delta(y,\cdot)$ is not an extension of $\Delta(y,\cdot)$, we can always find a positive scale factor γ specific to ℓ such that margin rescaling yields an extension. We denote $\mathbf{M}\gamma\Delta$ and $\gamma\Delta$ as the rescaled functions.*

Proof. Similar to Proposition 4.1, we analyse two cases to determine the values of $\mathbf{M}\gamma\Delta(y,g(x))$:

1. if $u=\mathbf{0}$, $\mathbf{M}\gamma\Delta(y,g(x)) = \gamma\ell(\emptyset)$ where u is defined as in Equation (4.3.10). It is typically the case that $\ell(\emptyset)=0$, but this is not a technical requirement.

2. if $u\neq\mathbf{0}$, let $\mathbf{I}=\{i|u^i\neq 0\}$, then $\mathbf{M}\gamma\Delta(y,g(x))$ takes the value of the following equation:
$$\max_{\mathbf{I}\in\mathcal{P}(V)} \gamma\ell(\mathbf{I}) - \sum_{i\in\mathbf{I}} g^i(x)y^i \tag{4.3.18}$$

To satisfy Definition 4.1, we must find a $\gamma>0$ such that $\mathbf{M}\gamma\Delta(y,g(x)) = \gamma\Delta(y,\operatorname{sign}(g(x)))$ at the vertices. Note that it is trivial when $u=\mathbf{0}$ so the first case is true for arbitrary $\gamma>0$.

For the second case as in Equation (4.3.18), let $\mathbf{I}_2 = \arg\max_{\mathbf{I}\in\mathcal{P}(V)} \big(\ell(\mathbf{I}) - \sum_{i\in\mathbf{I}} g^i(x)y^i\big)$. We have $\mathbf{M}\gamma\Delta(y,g(x)) = \Delta(y,\operatorname{sign}(g(x)))$ at the vertices $\mathbf{I}_2$ according to the extension. The scale factor should satisfy:

$$\forall \mathbf{I}_1\in\mathcal{P}(V)\setminus\{\emptyset\},\ \gamma\big(\ell(\mathbf{I}_2)-\ell(\mathbf{I}_1)\big) \geq -|(V\setminus\mathbf{I}_2)\cap\mathbf{I}_1| \tag{4.3.19}$$

which leads to the following cases:

1. if $|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1| = 0$, we have $(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1 = \emptyset$, which implies $\mathbf{I}_1 \subseteq \mathbf{I}_2$. Equation (4.3.19) reduces to
$$\gamma\big(\ell(\mathbf{I}_2) - \ell(\mathbf{I}_1)\big) \geq 0 \tag{4.3.20}$$
and ℓ is an increasing function so $\ell(\mathbf{I}_1) \leq \ell(\mathbf{I}_2)$ and Equation (4.3.20) is always true as $\gamma > 0$.

2. if $|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1| \neq 0$, we need to discuss the relationship between $\ell(\mathbf{I}_1)$ and $\ell(\mathbf{I}_2)$:
   (a) if $\ell(\mathbf{I}_2) = \ell(\mathbf{I}_1)$, then Equation (4.3.20) becomes $0 \geq -|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1|$, for which the rhs is negative so it is always true.
   (b) if $\ell(\mathbf{I}_2) > \ell(\mathbf{I}_1)$, then
   $$\gamma \geq \frac{-|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1|}{\ell(\mathbf{I}_2) - \ell(\mathbf{I}_1)} \tag{4.3.21}$$
   for which the rhs is negative so it is redundant with $\gamma > 0$.
   (c) if $\ell(\mathbf{I}_2) < \ell(\mathbf{I}_1)$, then
   $$\gamma \leq \frac{-|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1|}{\ell(\mathbf{I}_2) - \ell(\mathbf{I}_1)} \tag{4.3.22}$$
   for which the rhs is *strictly* positive so it becomes an upper bound on γ .

In summary the scale factor γ should satisfy the following constraint for an increasing loss function ℓ:
$$\forall \mathbf{I}_1, \mathbf{I}_2 \in \mathcal{P}(V) \setminus \{\emptyset\}, 0 < \gamma \leq \frac{-|(V \setminus \mathbf{I}_2) \cap \mathbf{I}_1|}{\ell(\mathbf{I}_2) - \ell(\mathbf{I}_1)}$$
Finally, we note that the rightmost ratio is always strictly positive. □

4.3.4 Complexity of Subgradient Computation

Although we have proven that slack and margin rescaling yield extensions to the underlying discrete loss under fairly general conditions, their key shortcoming is in the complexity of the computation of subgradients for submodular losses. The subgradient computation for slack and margin rescaling requires the computation of
$$\arg\max_{\tilde{y}} \Delta(y, \tilde{y})(1 + h(x, \tilde{y}) - h(x, y)) \tag{4.3.23}$$
and
$$\arg\max_{\tilde{y}} \Delta(y, \tilde{y}) + h(x, \tilde{y}), \tag{4.3.24}$$
respectively. The $\arg\max$ in both of these functions corresponds to a non-submodular minimization for submodular loss functions. This computation is NP-hard, and such loss functions are not feasible with these existing methods in practice. Furthermore, approximate inference, e.g. based on [Nemhauser et al., 1978], leads to poor convergence when used to train a structured output SVM resulting in a high error rate (cf. Section 4.7, Tables 4.1-4.4). We therefore introduce the Lovász hinge as an alternative operator to construct feasible convex surrogates for submodular losses.

4.4 Lovász Hinge

We now develop our convex surrogate for submodular loss functions, which is based on the Lovász extension. In the sequel, the notion of permutations is important. We denote a permutation of p elements by $\pi = (\pi_1, \pi_2, \dots, \pi_p)$, where $\pi_i \in \{1, 2, \dots p\}$ and $\pi_i \neq \pi_j,\ \forall i \neq j$. Furthermore, for a vector $s \in \mathbb{R}^p$, we will be interested in the permutation π that sorts the elements of s in decreasing order, i.e. $s^{\pi_1} \geq s^{\pi_2} \geq \dots \geq s^{\pi_p}$. Without loss of generality, we will index the base set of a set function ℓ by integer values, i.e. $V = \{1, 2, \dots, p\}$, so that for a permutation π, $\ell(\{\pi_1, \pi_2, \dots, \pi_i\})$ is well defined.

4.4.1 Lovász Hinge

We propose our novel convex surrogate for submodular functions increasing and non-increasing in Definition 4.2 and Definition 4.3, respectively.[1]

Definition 4.2 (Lovász hinge for submodular increasing functions)**.** *For ℓ submodular and* increasing*, the Lovász hinge* $\mathbf{L}$ *is defined as the unique operator such that,*

$$\mathbf{L}\Delta(y^*, g(x)) := \max_{\pi} \sum_{j=1}^{p} (s^{\pi_j})_+ \left(\ell\left(\{\pi_1, \cdots, \pi_j\}\right) - \ell\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)\right) \tag{4.4.1}$$

where $(\cdot)_+ = \max(\cdot, 0)$*,* π *is a permutation,*

$$s^{\pi_j} = 1 - g^{\pi_j}(x) y^{*\pi_j}, \tag{4.4.2}$$

and $g^{\pi_j}(x)$ *is the* π_j*th dimension of* $g(x)$ *(cf. Equation* (4.3.7)*).*

Definition 4.3 (Lovász hinge for submodular non-increasing functions)**.** *For ℓ submodular and* non-increasing*, the Lovász hinge,* $\mathbf{L}$*, is defined as the unique operator such that*

$$\mathbf{L}\Delta(y, g(x)) := \left(\max_{\pi} \sum_{j=1}^{p} s^{\pi_j} \left(\ell\left(\{\pi_1, \cdots, \pi_j\}\right) - \ell\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)\right) \right)_+ \tag{4.4.3}$$

with the same π*,* s^{π_j}*,* $g^{\pi_j}(x)$ *and* $(\cdot)_+$ *defined as in Definition 4.2.*

Note that the Lovász hinge is itself an extension of the Lovász extension (defined over the p-dimensional unit cube) to $\mathbb{R}^p$. For ℓ being submodular increasing, we threshold each negative component of s i.e. $s^{\pi_j} = 1 - g^{\pi_j}(x) y^{\pi_j}$ to zero. As ℓ being increasing, the components $\ell\left(\{\pi_1, \cdots, \pi_j\}\right) - \ell\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)$ will always be non-negative. Then $\mathbf{L}\Delta(y, g(x))$ in Defintion 4.2 is always non-negative. While in Definition 4.3, we don't apply the thresholding strategy on the components of s, but instead on the entire formulation.

We show in the following propositions that these two definitions yield surrogates that are indeed convex and extensions of the corresponding loss functions.

[1] Source code is available for download at `https://github.com/yjq8812/lovaszhinge`.

Proposition 4.3. *For a submodular non-negative loss function ℓ, the Lovász hinge as in Definition 4.3 is convex and yields an extension of Δ.*

Proof. We first demonstrate the "convex" part. It is clear that taking a maximum over linear functions is convex. The thresholding to non-negative values similarly maintains convexity.

We then demonstrate the "extension" part. From the definition of extension as in Equation (4.3.10) as well as the notation concerning the vertices of the unit cube as in Equation (4.3.12), for the values of s on the vertices $\mathbf{I}$, we have explicitly

$$s^j = \begin{cases} 1, & \forall j \in \mathbf{I} \\ 0, & \forall j \in V \setminus \mathbf{I} \end{cases} \tag{4.4.4}$$

As a result, the permutation π that sorts the components of s decreasing on the vertices is actually

$$\{\pi_1, \cdots, \pi_k, \pi_{k+1}, \cdots, \pi_p\},\ k = |\mathbf{I}|.$$

with $\{\pi_1, \cdots, \pi_k\}$ a permutation of $\{j | j \in \mathbf{I}\}$ and $\{\pi_{k+1}, \cdots, \pi_p\}$ a permutation of $\{j | j \in V \setminus \mathbf{I}\}$.

We then reformulate the inside part of rhs of Equation (4.4.3) as:

$$\begin{aligned} &\sum_{j=1}^{p} s^{\pi_j} \left(\ell\left(\{\pi_1, \cdots, \pi_j\}\right) - \ell\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)\right) \\ &= \sum_{j=1}^{k} 1 \times \left(\ell\left(\{\pi_1, \cdots, \pi_j\}\right) - \ell\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)\right) + 0 \\ &= \ell\left(\{\pi_1, \cdots, \pi_k\}\right) \\ &= \ell(\mathbf{I}) \end{aligned} \tag{4.4.5}$$

Then for non-negative ℓ, thresholding is redundant. As a consequence we have

$$\mathbf{L}\Delta(y, g(x)) = (\ell(\mathbf{I}))_+ = \Delta(y, g(x)) \tag{4.4.6}$$

which validates that $\mathbf{L}\Delta(y, g(x))$ yields an extension of $\Delta(y, g(x))$. □

Proposition 4.4. *For a submodular increasing loss function ℓ , the Lovász hinge as in Definition 4.2 is convex and yields an extension of Δ.*

Proof. We first demonstrate the "convex" part. When ℓ is increasing, $\mu^{\pi_j} = \ell\left(\{\pi_1, \cdots, \pi_j\}\right) - \ell\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)$ will always be positive for all $\pi_j \in \{1, 2, \cdots, p\}$. So it's obvious that $\mu^{\pi_j} \times \max(0, s^{\pi_j})$ will always be convex with respect to $s^{\pi_j} \in \mathbb{R}$ for any given $\mu^{\pi_j} \geq 0$. As the convex function set is closed under the operation of addition, the Lovász hinge is convex in $\mathbb{R}^p$.

We note that on the vertex $\mathbf{I}$, we have $s \in \{0, 1\}^p$ as in Equation (4.4.4), then the positive threshold on the components of s^j can be removed. With the same procedure as in Equation (4.4.5), $\mathbf{L}\Delta(y, g(x))$ yields an extension of $\Delta(y, g(x))$. □

Although Definition 4.3 is convex and yields an extension for both increasing and non-increasing losses, we rather apply Definition 4.2 for increasing loss functions. This will ensure the Lovász hinge is an analogue to a standard hinge loss in the special case of a symmetric modular loss. We formally state this as the following proposition:

Proposition 4.5. *For a submodular increasing loss function ℓ, the Lovász hinge as in Definition 4.2 thresholding negative s^{π_j} to zero, coincides with an SVM (i.e. additive hinge loss) in the case of Hamming loss.*

Proof. The hinge loss for a set of p training samples is defined to be

$$\ell_{\text{hinge}}(y, g(x)) := \sum_{i=1}^{p} \left(1 - g(x^i)y^i)\right)_+ . \tag{4.4.7}$$

Following the previous notation, we will interpret $g(x^i) = g^i(x)$. For a modular loss, the Lovász hinge in Equation (4.4.1) simplifies to

$$\begin{aligned} \mathbf{L}\Delta(y, g(x)) &= \max_{\pi} \sum_{j=1}^{p} (s^{\pi_j})_+ \ell(\{\pi_j\}) \qquad (4.4.8)\\ &= \sum_{j=1}^{p} \left(1 - g^j(x)y^j\right)_+ \ell(\{j\}). \qquad (4.4.9) \end{aligned}$$

For the Hamming loss $\ell(\{j\}) = 1, \forall j$ and $\mathbf{L}\Delta(y, g(x)) = \ell_{\text{hinge}}(y, g(x))$. □

Lemma 4.1. *The convex closure of a set function ℓ is the largest function $\ell_c : [0,1]^p \mapsto \mathbb{R} \cup \{+\infty\}$ such that (a) ℓ_c is convex and (b) for all $A \subseteq V, l_c(1_A) \leq l(A)$, where 1_A is a binary vector such that the ith element is one iff $i \in A$ and zero otherwise. The Lovász extension of a submodular function ℓ coincides with its convex closure [Bach, 2013; Choquet, 1953].*

Proposition 4.6. *The Lovász hinge has an equal or higher value than slack and margin rescaling inside the unit cube.*

Proof. By Lemma 4.1 it is clear that the Lovász hinge inside the unit cube is the maximal convex extension, and therefore slack and margin rescaling must have equal or lesser values. □

Corollary 4.1. *Slack and margin rescaling may have additional inflection points inside the unit cube that are not present in the Lovász hinge. This is a consequence of Proposition 4.6 and the fact that all three are piecewise linear extensions, and thus have identical values on the vertices of the unit cube. One can also visualize this in Figure 4.2.*

Another reason that we use a different threshold strategy on s for increasing or non-increasing losses is that we cannot guarantee that the Lovász hinge is always convex if we threshold each negative component s^{π_j} to zero for non-increasing losses.

Proposition 4.7. *For a submodular non-increasing loss function ℓ , the Lovász hinge is* not *convex if we threshold each negative component s^{π_j} to zero as in Definition 4.2.*

Proof. If ℓ is non-increasing, there exists at least one π and j such that

$$\mu^{\pi_j} := l\left(\{\pi_1, \cdots, \pi_j\}\right) - l\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)$$

is strictly negative. The partial derivative of $\mathbf{L}\Delta(y, g(x))$ w.r.t. s^{π_j} is

$$\frac{\partial \mathbf{L}\Delta(y, g(x))}{\partial s^{\pi_j}} = \begin{cases} 0 & \text{if } s^{\pi_j} < 0 \\ \mu^{\pi_j} & \text{if } s^{\pi_j} > 0. \end{cases} \tag{4.4.10}$$

As $\mu^{\pi_j} < 0$ by assumption, we have that the partial derivative at $s^{\pi_j} < 0$ is larger than the partial derivative at $s^{\pi_j} > 0$, and the loss surface cannot therefore be convex. □

Figure 4.6(a) and Figure 4.6(b) show an example of the loss surface when ℓ is non-increasing. We can see that at the non-increasing element leads to a negative subgradient at one side of a vertex, while on its other side the subgradient is zero due to the fact that we still apply the thresholding strategy. Thus it leads to a non-convex surface.

4.4.2 Complexity of Subgradient omputation

We explicitly present the cutting plane algorithm for solving the max-margin problem as in Equation (4.3.3) in Algorithm 4. The novelties are: (i) in Line 5 we calculate the upper bound on the empirical loss by the Lovász hinge; (ii) in Line 6 we calculate the loss gradient by the computation relating to the permutation π instead of to all possible outputs $\tilde{y}$.

As the computation of a cutting plane or loss gradient is precisely the same procedure as computing a value of the Lovász extension, we have the same computational complexity, which is $\mathcal{O}(p \log p)$ to sort the p coefficients, followed by $\mathcal{O}(p)$ oracle accesses to the loss function [Lovász, 1983]. This is precisely an application of the greedy algorithm to optimize a linear program over the submodular polytope as shown in Proposition 2.11. In our implementation, we have employed a one-slack cutting-plane optimization with ℓ_2 regularization analogous to [Joachims et al., 2009]. We observe empirical convergence of the primal-dual gap at a rate comparable to that of a structured output SVM (Figure 5.5).

4.4.3 Visualization of Convex Surrogates

For visualization of the loss surfaces, in this section we consider a simple binary classification problem with two elements with non-modular loss functions :

$$\mathcal{X} := \mathbb{R}^{d\times 2} \qquad \mathcal{Y} := \{-1, +1\}^2$$

Then with different values for $\ell(\emptyset)$, $\ell(\{1\})$, $\ell(\{2\})$ and $\ell(\{1, 2\})$, we can have different modularity or monotonicity properties of the function ℓ which then defines Δ. We illustrate the Lovász hinge, slack and margin rescaling for the following cases:

Algorithm 4 Cutting plane algorithm for solving the problem in Equation (4.3.3), with Definition 4.3 – the Lovász hinge.

```
1: Input: $(x_1, y_1), \cdots, (x_n, y_n), C, \epsilon$
2: $S^i = \emptyset, \forall i = 1, \cdots, n$
3: repeat
4:     for $i = 1, \cdots, n$ do
5:         $H(y_i, \pi) = \sum_{j=1}^{p} s_i^{\pi_j} \left( l\left(\{\pi_1, \cdots, \pi_j\}\right) - l\left(\{\pi_1, \cdots, \pi_{j-1}\}\right)\right)$,
           where $s^{\pi_j} = 1 - g^{\pi_j}(x_i) y_i^{\pi_j}$
6:         $\hat{\pi} = \arg\max_{\pi} H(y_i, \pi)$
           % find the most violated constraint by sorting the $p$ elements of $s$
7:         $\xi^i = \max\{0, H(y_i, \hat{\pi})\}$
8:         if $H(y_i, \hat{\pi}) > \xi^i + \epsilon$ then
9:             $S^i := S^i \cup \{\hat{\pi}\}$
10:            $w \leftarrow$ optimize Equation (4.3.3) with constraints defined by $\cup_i S^i$
11:        end if
12:    end for
13: until no $S^i$ has changed during iteration
14: return $(w, \xi)$
```

(i) submodular increasing:

$$\ell(\emptyset) = 0,\ l(\{1\}) = l(\{2\}) = 1,\ l(\{1,2\}) = 1.2$$

(ii) submodular non-increasing:

$$\ell(\emptyset) = 0,\ l(\{1\}) = l(\{2\}) = 1,\ l(\{1,2\}) = 0.6$$

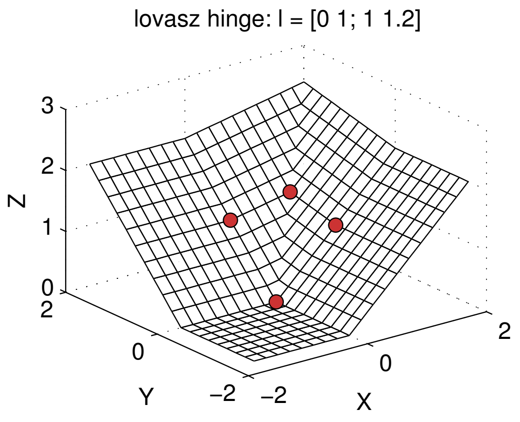
(iii) supermodular increasing:

$$\ell(\emptyset) = 0,\ l(\{1\}) = l(\{2\}) = 1,\ l(\{1,2\}) = 2.6$$

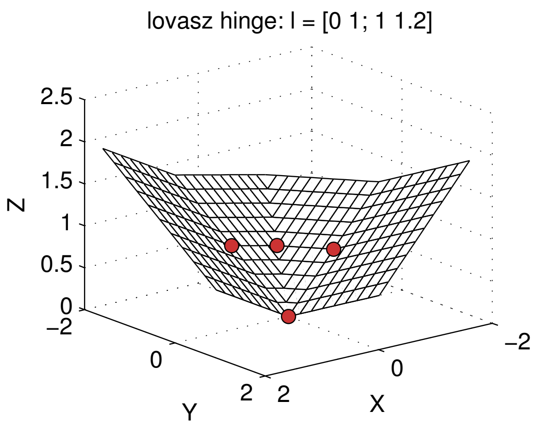
In Figure 4.2 to Figure 4.5, the x axis represents the value of s^1, the y axis represents the value of s^2 in Equation (4.4.2), and the z axis is the convex loss function given by Equation (4.3.13), Equation (4.3.14), and Definition 4.2 and 4.3 for different loss functions. We plot the values of ℓ as solid dots at the vertices of the hypercube.

We observe first that all surfaces are convex. For the Lovász hinge with a submodular increasing function in Figure 4.2(a) and Figure 4.2(b), the thresholding strategy adds two hyperplanes on the left and on the right, while convexity is maintained.

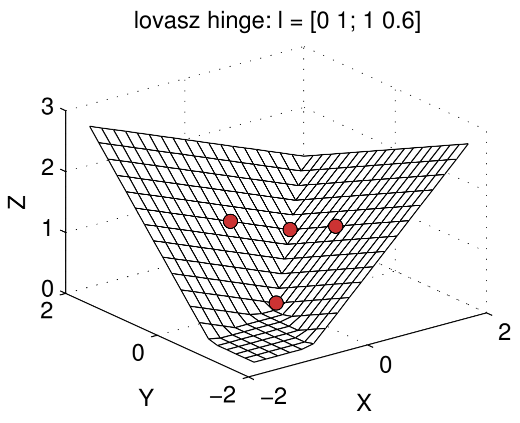
We observe in Figure 4.2 that all the solid dots corresponding to the discrete loss function values *touch* the surfaces of the Lovász hinge with both submodular increasing loss function and non-increasing loss function, which empirically validates that the surrogates are extensions of the discrete loss.



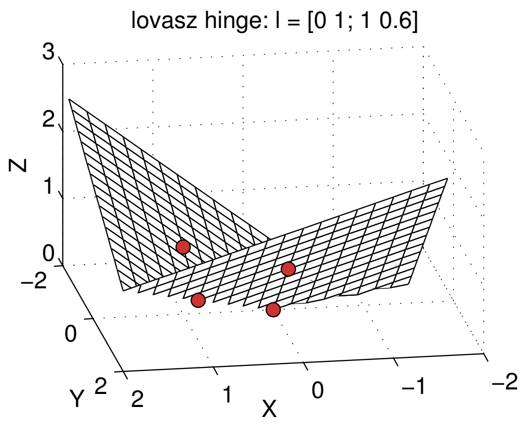
(a) **L** with submodular increasing ℓ



(b) **L** with submodular increasing ℓ

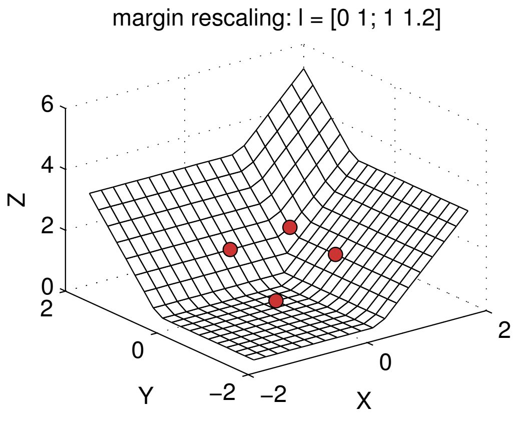


(c) **L** with submodular non-increasing ℓ

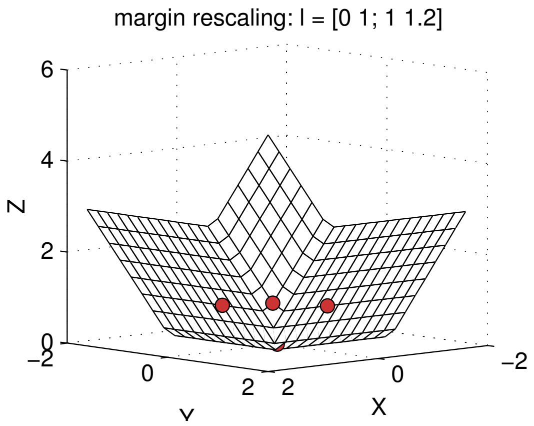


(d) **L** with submodular non-increasing ℓ

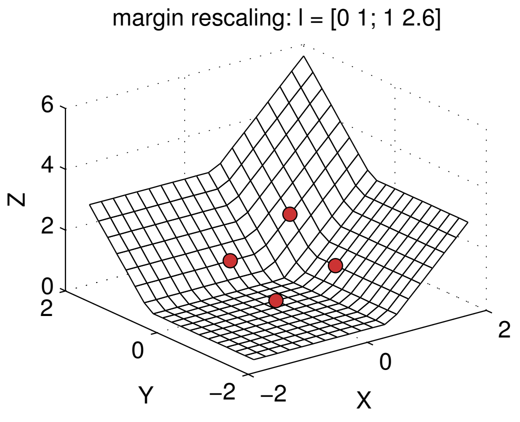
Figure 4.2: The Lovász hinge surfaces with a submodular increasing loss and a submodular non-increasing loss from different views; the x and y axes represent the value of s_i^1 and s_i^2 in Equation (4.4.2); the z axis represents the value of the convex surrogate; the solid red dots represent the values of ℓ at the vertices of the unit hypercube. The convex surrogate strategies yield extensions of the discrete loss as the red dots touch the surfaces.



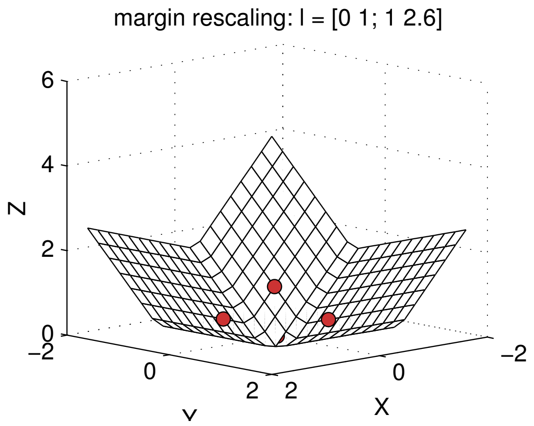
(a) $\mathbf{M}$ with submodular increasing ℓ



(b) $\mathbf{M}$ with submodular increasing ℓ

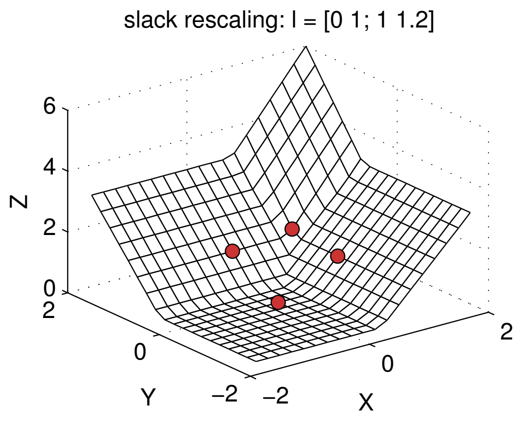


(c) $\mathbf{M}$ with supermodular increasing ℓ

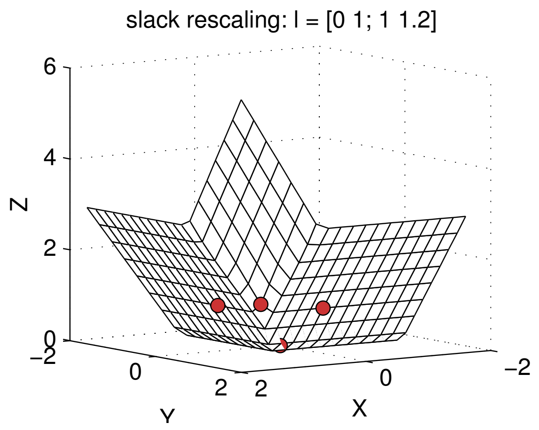


(d) $\mathbf{M}$ with supermodular increasing ℓ

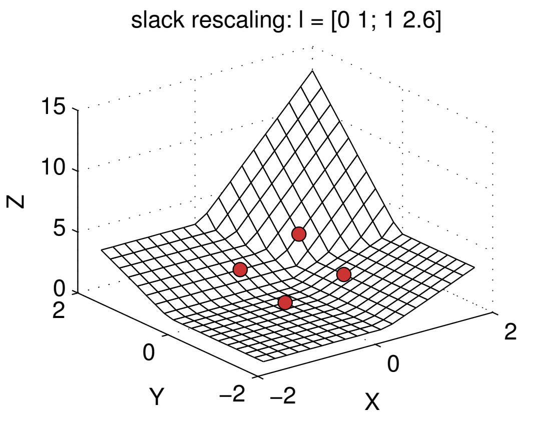
Figure 4.3: Margin rescaling surfaces with a submodular increasing loss and a supermodular increasing loss from different views. The convex surrogate strategies yield extensions of the discrete loss as the red dots touch the surfaces.



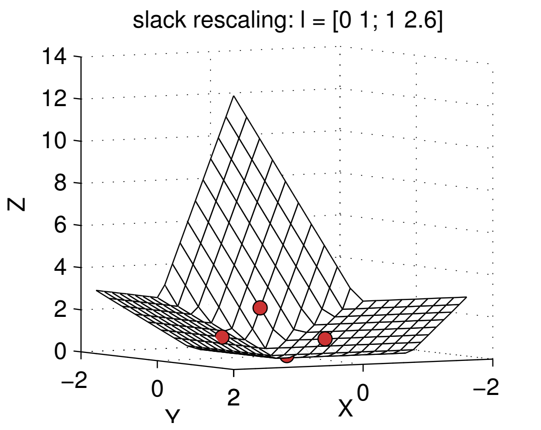
(a) **S** with submodular increasing ℓ



(b) **S** with submodular increasing ℓ

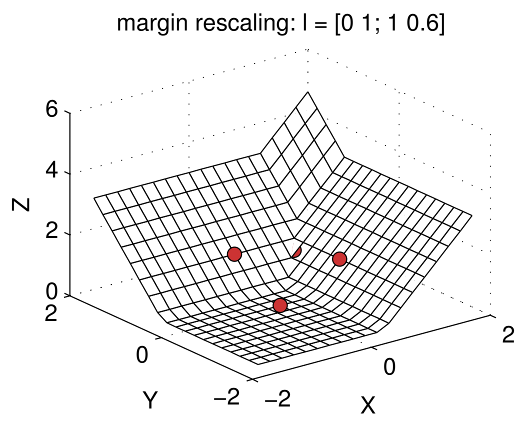


(c) **S** with supermodular increasing ℓ

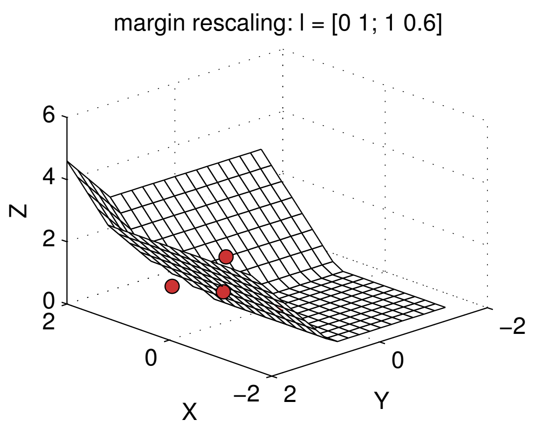


(d) **S** with supermodular increasing ℓ

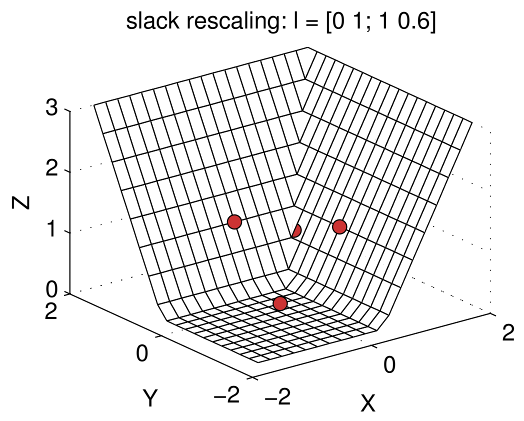
Figure 4.4: Slack rescaling surfaces with a submodular increasing loss and a supermodular increasing loss from different views. The convex surrogate strategies yield extensions of the discrete loss as the red dots touch the surfaces.



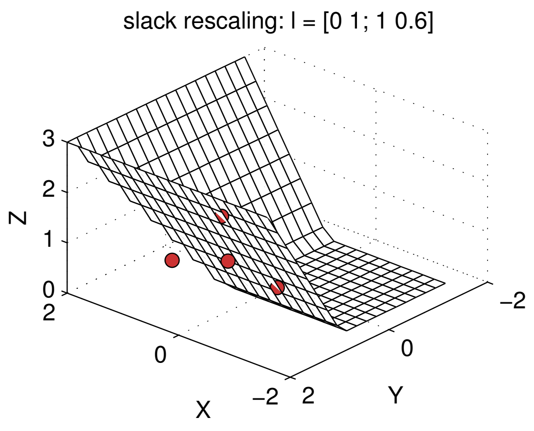
(a) **M** with submodular non-increasing ℓ



(b) **M** with submodular non-increasing ℓ



(c) **S** with submodular non-increasing ℓ



(d) **S** with submodular non-increasing ℓ

Figure 4.5: Magin rescaling surfaces and slack rescaling surfaces with a submodular non-increasing loss from different views. The convex surrogate strategies fail to yield extensions of the discrete loss as the red dots are below the surfaces.

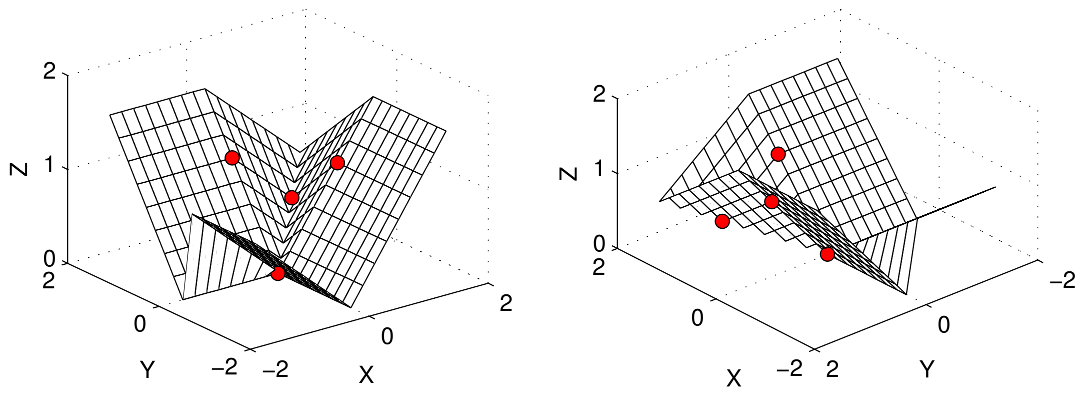


Figure 4.6: Lovász hinge with submodular non-increasing ℓ while thresholding negative components of s is still applied (cf. the caption of Figure 4.2 for the axes notation). Although the red dots touch the surface, the surface is no longer convex due to the thresholding strategy.

However, for slack rescaling and margin rescaling (while having a proper positive scale factor γ), the dots coincide with the surfaces only for submodular increasing and supermodular increasing ℓ (Figure 4.3 to Figure 4.4); margin rescaling and slack rescaling fail to yield extensions of submodular non-increasing losses as in Figure 4.5, as the red dots are below the surfaces.

Here we set ℓ as symmetric functions, while the extensions can be also validated for asymmetric increasing set functions.

We additionally plot the Lovász hinge with a non-increasing function while the thresholding strategy on s is still applied in Figure 4.6. Compared to Figure 4.2(c) and Figure 4.2(d), convexity is lost due to the thresholding strategy on negative components of s.

4.5 Hardness Result for Learning Multiple Correct Results

Many domains of structured prediction contain multiple correct outputs. In computer vision, multiple correct object detections may be present in an image [Blaschko, 2011]. In text summarization, multiple paragraphs may be considered equally good summaries of a document [?]. In protein structure prediction, a molecule may have multiple possible configurations [?]. In this work, we show that the presence of multiple correct outputs leads to intractable computational problems in many common settings for which a single correct output leads to tractable problems.

In this section, we show three main hardness results for structured prediction with multiple correct outputs: (i) regularized risk minimization with a supermodular loss function is tractable with existing learning frameworks for a single correct output but NP-hard for multiple correct outputs (Proposition 4.8), (ii) regularized risk minimization with a submodular loss function is tractable with existing learning frameworks for a single correct output but NP-hard for multiple correct outputs (Proposition 4.9), and (iii) test time inference that is polynomial time solvable for a single correct output is NP-hard for multiple correct outputs when a diversity penalty is included (Proposition 4.10). These results suggest the use of alternative learning and approximate inference schemes when multiple correct outputs are present during training and/or testing.

These results give substantial evidence that structured learning and inference with multiple correct outputs is fundamentally harder than when only a single output is considered correct. This points to two potentially productive directions of inquiry: (i) the exploration of tractable approximations to the NP-hard learning and inference problems, and (ii) the derivation of novel convex surrogates and sufficient conditions for polynomial time learning and inference that are applicable in practice to problems of interest.

4.5.1 Learning

In structured output learning, we assume that a task specific loss function is given that measures the disagreement between a prediction and a ground truth element. We denote a ground truth instance as $y \in \mathcal{Y}$ and the (incorrect) prediction $\tilde{y}$. We will distinguish between a loss function in which a single correct output is to be predicted, and a loss function in which p multiple correct outputs $Y \in \mathcal{Y}^p$ are possible:

$$\Delta_{\text{single}} : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R} \tag{4.5.1}$$

$$\Delta_{\text{multiple}} : \mathcal{Y}^p \times \mathcal{Y} \mapsto \mathbb{R}. \tag{4.5.2}$$

Specifically, we will assume that when there are multiple outputs we will take

$$\Delta_{\text{multiple}}(Y, \tilde{y}) := \min_{y \in Y} \Delta_{\text{single}}(y, \tilde{y}) \tag{4.5.3}$$

so that we require a prediction to be close to one of the ground truth outputs. We focus on two feasible families of loss functions for Δ_{single} for which convex surrogates have been developed: supermodular loss functions and submodular loss functions.

4.5.2 Supermodular losses and the Structured Output SVM

We see in the the previous section that it is a necessary, but not sufficient condition that the loss function $\Delta(y, \tilde{y})$ be increasing for margin rescaling to yield an extension. However, for all increasing $\Delta(y, \tilde{y})$ there exists a positive scaling $\gamma \in \mathbb{R}$ such that margin rescaling yields an extension. These results indicate that both slack and margin rescaling can be used to construct tight convex surrogates to increasing loss functions.

In the event that we have multiple correct outputs, we construct the loss imputed to the SOSVM by taking the minimum loss over all possible correct outputs. The resulting loss remains increasing, and therefore the resulting convex surrogate is tight, a positive result from a statistical perspective. However, multiple correct outputs may mean that computation of the subgradient of the convex surrogate is NP-hard. We show this making use of the following lemma:

Proposition 4.8. *Computation of a subgradient of the slack and margin rescaling convex surrogates is NP-hard when there are multiple correct outputs.*

Proof. Slack and margin rescaling are computationally feasible only when Δ is supermodular (Chapter 4.3). This is because subgradient computation requires solving

$$\arg\max_{\tilde{y}} \Delta(y, \tilde{y})(1 + \langle w, \phi(x, \tilde{y}) - \phi(x, y)\rangle) \tag{4.5.4}$$

and

$$\arg\max_{\tilde{y}} \Delta(y, \tilde{y}) + \langle w, \phi(x, \tilde{y})\rangle \tag{4.5.5}$$

for slack and margin rescaling, respectively. The minimum over supermodular functions is the negative of the maximum over submodular functions, and submodular functions are not closed under maximization (see Theorem 2.13 in Chapter 2). Finally we note that the $\arg\max$ in Equations (4.5.4) and (4.5.5) will therefore be taken over non-supermodular functions, which is NP-hard in general. □

Consequently, even if we have a tractable loss augmented inference problem for a single correct output with a supermodular loss, the presence of multiple correct outputs will lead to NP-hard loss augmented inference problems.

4.5.3 Submodular losses and the Lovász Hinge

We now turn to submodular loss functions. These result in NP-hard subgradient computation for SOSVMs, so Chapter 4.4 introduced an alternative convex surrogate based on the Lovász extension of a submodular set function, called the Lovász hinge. This convex surrogate generalizes hinge loss to multiple outputs when the loss function is submodular. Subgradient computation has a computational complexity of $\mathcal{O}(|y| \log |y|)$, where $|y|$ is the size of a single instance $y \in \mathcal{Y}$. We note that this convex surrogate (written as the maximum over linear constraints) is only tight when Δ remains submodular. Analogous to Proposition 4.8, we now show that multiple correct outputs results in a loss function Δ that is not guaranteed to be submodular, even if the loss function for a single correct output is submodular. We make use of the following result:

Proposition 4.9. *Neither the Lovász hinge nor the structured output SVM provide a polynomial time tight convex surrogate to a submodular loss function when there are multiple correct outputs.*

Proof. Application of Equation (4.5.3) combined with the fact that submodularity is not closed under minimization (see Theorem 2.14 in Chapter 2) indicates that Δ_{multiple} is not submodular in general, even if Δ_{single} is. As Δ_{multiple} is neither submodular nor supermodular, neither the Lovász hinge nor the SOSVM yield polynomial time tight convex surrogates. □

4.5.4 The Inference Problem

In the previous section, we have shown that learning with both submodular and supermodular loss functions leads to NP-hard computation in order to compute subgradients of existing convex surrogate loss functions when there are multiple correct outputs. In this section, we further show a result due to Blaschko [2011] that test time inference becomes NP-hard in the presence of multiple correct outputs when a diversity penalty is included in the inference procedure.

Proposition 4.10. *Let $g(x, y)$ be a compatibility function for a structured prediction problem (e.g. $\langle w, \phi(x, y)\rangle$ from a SOSVM). The prediction of a set of $p \geq 2$ outputs with a diversity penalty is NP-hard in general:*

$$\arg\max_{Y \in \mathcal{Y}^p} \sum_{y \in Y} g(x, y) - \mathbf{\Omega}(Y) \tag{4.5.6}$$

where

$$\mathbf{\Omega}(\mathbf{Y}) = \sum_{i \neq j} \Omega(y_i, y_j) + \sum_{c \in \mathcal{C}} \Omega_c(y_c), \tag{4.5.7}$$

$\mathcal{C}$ is the set of higher order cliques in the penalty term (possibly $\mathcal{C} = \emptyset$), and Ω_c is supermodular for all $c \in \mathcal{C}$.

Proof. Section 3 of [Blaschko, 2011] shows that Equation (4.5.7) is supermodular for $\Omega \geq 0$. Consequently, the optimization in Equation (4.5.6) corresponds with a non-submodular minimization and is NP-hard. □

4.6 Jaccard Loss

The Jaccard index score is a popular measure for comparing the similarity between two sample sets, widely applied in diverse prediction problems such as structured output prediction [Blaschko and Lampert, 2008], social network prediction [Liben-Nowell and Kleinberg, 2007] and image segmentation [Unnikrishnan et al., 2007]. It is used in the evaluation of popular computer vision challenges, such as PASCAL VOC [Everingham et al., 2010] and ImageNet [Russakovsky et al., 2014, Sec. 4.2]. In this section, we introduce the Jaccard loss based on the Jaccard index score, and we prove that the Jaccard loss is a submodular function with respect to the set of mispredicted elements.

We will use $P_y, P_{\tilde{y}} \subseteq V$ to denote sets of positive predictions. We define the Jaccard loss to be [Blaschko and Lampert, 2008]

$$\Delta_J(y, \tilde{y}) := 1 - \frac{|P_y \cap P_{\tilde{y}}|}{|P_y \cup P_{\tilde{y}}|} \tag{4.6.1}$$

We now show that this is submodular under the isomorphism $(y, \tilde{y}) \to A := \{i | y^i \neq \tilde{y}^i\}$, $\Delta_J(y, \tilde{y}) \cong \ell(A)$. We will use the diminishing returns definition of submodularity as in Definition 2.1. We will denote $m := |P_y| > 0$, $p := |P_{\tilde{y}} \setminus P_y|$, and $n := |P_y \setminus P_{\tilde{y}}|$. With this notation, we have that

$$\Delta_J(y, \tilde{y}) = 1 - \frac{m - n}{m + p} \tag{4.6.2}$$

For a fixed groundtruth y, we have for two sets of mispredictions A and B

$$\text{if } B \subseteq A \text{, then } \; n_B \leq n_A, \; p_B \leq p_A \tag{4.6.3}$$

We first prove two lemmas about the submodularity of the loss function restricted to additional false positives or false negatives.

Lemma 4.2. *Δ_J restricted to marginal false negatives is submodular.*

Proof. If i is an extra false negative,

$$\begin{aligned}\Delta_J(A \cup \{i\}) &= \Delta_J(n_A + 1, p_A) \\ &= \frac{n_A + p_A + 1}{m + p_A} \\ &= \Delta_J(A) + \frac{1}{m + p_A}\end{aligned} \tag{4.6.4}$$

Then we have

$$\begin{aligned}\Delta_J(A \cup \{i\}) - \Delta_J(A) &= \frac{1}{m + p_A} \\ &\leq \frac{1}{m + p_B} \\ &= \Delta_J(B \cup \{i\}) - \Delta_J(B)\end{aligned} \tag{4.6.5}$$

which complies with the definition of submodularity. □

Lemma 4.3. *Δ_J restricted to marginal false positives is submodular.*

Proof. If i is an extra false positive, with the similar procedure as previous, we have

$$\begin{aligned}\Delta_J(A \cup \{i\}) &= \Delta_J(n_A, p_A + 1) \\ &= \frac{n_A + p_A + 1}{m + p_A + 1} \qquad (4.6.6)\\ \Delta_J(A \cup \{i\}) - \Delta_J(A) &= \frac{n_A + p_A + 1}{m + p_A + 1} - \frac{n_A + p_A}{m + p_A} \\ &= \frac{|A| - n_A}{(m + p_A + 1)(m + p_A)} \\ &\leq \frac{|A| - n_B}{(m + p_B + 1)(m + p_B)} \\ &= \Delta_J(B \cup \{i\}) - \Delta_J(B) \qquad (4.6.7)\end{aligned}$$

which also complies with the definition of submodularity. □

Proposition 4.11. Δ_J *is submodular.*

Proof. Lemmas 4.2 and 4.3 cover mutually exclusive cases whose union covers all possible marginal mistakes. As the diminishing returns property of submodularity (Equation (2.1.2)) holds in both cases, it also holds for the union. □

4.7 Experimental Results

We validate the Lovász hinge on a number of different experimental settings. In Section 4.7.1, we demonstrate a synthetic problem motivated by an early detection task. Next, we show that the Lovász hinge can be employed to improve image classification measured by the Jaccard loss on the PASCAL VOC dataset in Section 4.7.2. Multi-label prediction with submodular losses is demonstrated on the PASCAL VOC dataset in Section 4.7.3, and on the MS COCO dataset in Section 4.7.4. A summary of results is given in Section 4.7.5.

4.7.1 A Synthetic Problem

We designed a synthetic problem motivated by early detection in a sequence of observations. As shown in Figure 5.2, each star represents one sample in a *bag* and $p = 15$ samples form one bag. In each bag, the samples are arranged in a chronological order, with samples appearing earlier drawn from a different distribution than later samples. The red and magenta dots represent the early and late positive samples, respectively. The blue dots represent the negative samples. For the negative samples, there is no change in distribution between early and late samples, while the distribution of positive samples changes with time (e.g. as in the evolution of cancer from early to late stage).

We define the loss function as

$$\Delta_1(y, \tilde{y}) = \sum_{i=1}^{p} \gamma_i l_{sub}(\mathbf{I}_i) \qquad (4.7.1)$$

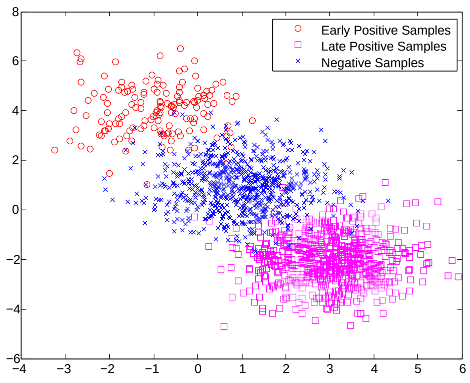


Figure 4.7: The disributions of samples for a synthetic binary classification problem motivated by the problem of early detection in a temporal sequence. As in, e.g. disease evolution, the distribution of early stage samples differs from that of late stage samples.

where $\mathbf{I}_i = \{j | j \leq i, \tilde{y}^j \neq y^j\}$. Let $\gamma_i := e^{-i}\ \forall i$. By this formulation, we penalize early mispredictions more than late mispredictions. This is a realistic setting in many situations where, e.g. early detection of a disease leads to better patient outcomes. The loss l_{sub} measures the misprediction rate up to the current time while being limited by an upper bound:

$$l_{sub}(\mathbf{I}_i) := \min\left(|\mathbf{I}_i|,\ \frac{i}{2}\right). \tag{4.7.2}$$

As Δ is a positively weighted sum of submodular losses, it is submodular. We additionally train and test with the 0-1 loss, which is equivalent to an SVM and Hamming loss in the Lovász hinge. We use different losses during training and during testing, then we measure the empirical loss values of one prediction as the average loss value for all images shown in Table. 4.1. We have trained on 1000 bags and tested on 5000 bags. **M** and **S** denote the use of the submodular loss with margin and slack rescaling, respectively. As this optimization is intractable, we have employed the approximate optimization procedure of [Nemhauser et al., 1978].

As predicted by theory, training with the same loss function as used during testing yields the best results. Slack and margin rescaling fail due to the necessity of approximate inference, which results in a poor discriminant function. By contrast, the Lovász hinge yields the best performance on the submodular loss.

4.7.2 PASCAL VOC Image Classification

We consider an image classification task on the Pascal VOC dataset [Everingham et al., 2010]. This challenging dataset contains around 10,000 images of 20 classes including animals, handmade and natural objects such as *person*, *bird*, *aeroplane*, etc. In the training set, which contains 5,011 images of different categories, the number of positive samples varies largely, we evaluate the prediction on the entire training/testing set with the Jaccard

		Test	
		Δ_1	0-1
Train	**L**	$\mathbf{0.100 \pm 0.001}$	7.42 ± 0.01
	0-1	0.166 ± 0.001	$\mathbf{2.87 \pm 0.02}$
	S	0.144 ± 0.001	7.69 ± 0.01
	M	0.154 ± 0.001	3.01 ± 0.01

Table 4.1: For the synthetic problem, the cross comparison of average loss values (with standard error) using different convex surrogates for the submodular loss in Equation (4.7.1) during training, 0-1 loss for comparison, and testing with different losses.

		aeroplane		bicycle		bird		boat		bottle	
Test time		Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1
Train	**L**	0.310	0.014	0.436	0.025	0.371	0.025	0.416	0.017	0.761	0.062
	0-1	0.310	0.014	0.466	0.027	0.399	0.025	0.483	0.020	0.917	0.045
		bus		car		cat		chair		cow	
Test time		Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1
Train	**L**	0.539	0.026	0.399	0.076	0.426	0.035	0.687	0.121	0.640	0.027
	0-1	0.661	0.025	0.397	0.068	0.469	0.034	0.744	0.090	0.892	0.023
		diningtable		dog		horse		motorbike		person	
Test time		Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1
Train	**L**	0.664	0.055	0.475	0.054	0.405	0.029	0.439	0.025	0.347	0.180
	0-1	0.695	0.043	0.564	0.050	0.451	0.032	0.493	0.027	0.325	0.151
		pottedplant		sheep		sofa		train		tvmonitor	
Test time		Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1	Δ_J	0-1
Train	**L**	0.747	0.058	0.580	0.017	0.710	0.062	0.306	0.018	0.588	0.042
	0-1	0.846	0.043	0.611	0.016	0.730	0.062	0.325	0.018	0.690	0.036

Table 4.2: For the VOC image classification task, the cross comparison of average loss values using the Jaccard loss as well as 0-1 loss during training and during testing for the 20 categories.

loss as in Equation (4.6.1).

We use Overfeat [Sermanet et al., 2014] to extract image features following the procedure described in [Razavian et al., 2014]. Overfeat has been trained for the image classification task on ImageNet ILSVRC 2013, and has achieved good performance on a range of image classification problems.

We compare learning with the Jaccard loss with the Lovász hinge with learning an SVM (labeled 0-1 in the results tables). The empirical error values using different loss function during test time are shown in Table 4.2.

We can see from the table that almost for all the categories, training with Jaccard loss using Lovász hinge yields lower empirical error values compared to training an Hamming loss, when the Jaccard loss is used at test time. This result is as expected by the empirical risk minimization principle, which empirically validates the correctness of Lovász hinge.

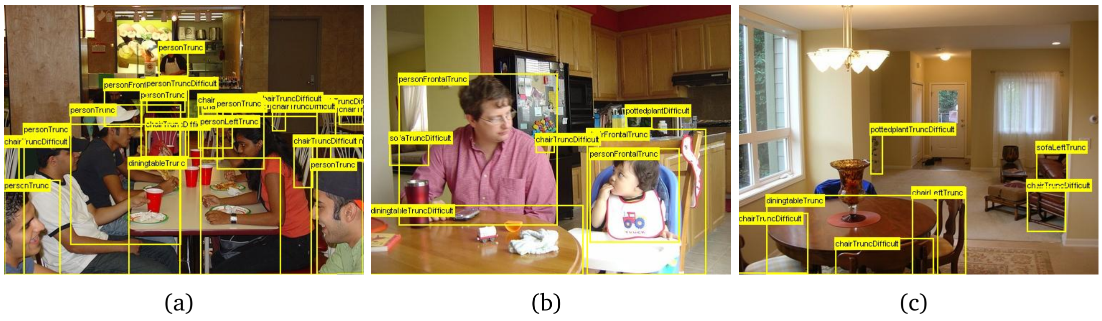


Figure 4.8: Example images from the PASCAL VOC dataset. Figure 4.8(a) and Figure 4.8(b) contain three categories: *people*, *table* and *chair*. Figure 4.8(c) contains *table* and *chair*.

4.7.3 PASCAL VOC Multilabel Prediction

We consider next a multi-label prediction task on the Pascal VOC dataset [Everingham et al., 2010], in which multiple labels need to be predicted simultaneously and for which a submodular loss over labels is to be minimized. Figure 4.8 shows example images from the dataset, including three categories i.e. *people*, *tables* and *chairs*. If a subsequent prediction task focuses on detecting "*people sitting around a table*", the initial multilabel prediction should emphasize that all labels must be correct *within a given image*. This contrasts with a traditional multi-label prediction task in which a loss function decomposes over the individual predictions. The misprediction of a single label, e.g. *person*, will preclude the chance to predict correctly the combination of all three labels. This corresponds exactly to the property of diminishing returns of a submodular function.

While using classic modular losses such as 0-1 loss, the classifier is trained to minimize the sum of incorrect predictions, so the complex interaction between label mispredictions is not considered. In this work, we use a new submodular loss function and apply the Lovász hinge to enable efficient convex risk minimization. In his experiment, we choose the most common combination of three categories: *person*, *chairs* and *dining table*. Objects labeled as *difficult* for object classification are not used in our experiments.

For the experiments, we repeatedly sample sets of images containing one, two, and three categories. The training/validation set and testing set have the same distribution. For a single iteration, we sample 480 images for the training/validation set including 30 images with all labels, and 150 images each for sets containing zero, one, or two of the target labels. More than 5,000 images from the entire dataset are sampled at least once as we repeat the experiments with random samplings to compute statistical significance. In this task, we first define the submodular loss function as follows:

$$\Delta_2(y, \tilde{y}) := \min\left(l_{\max}, \left\langle \beta, \frac{1 - y \odot \tilde{y}}{2} \right\rangle\right) \tag{4.7.3}$$

where $\odot$ is the Hadamard product, $l_{\max}$ is the maximal risk value, $\beta > 0$ is a coefficient vector of size p that accounts for the relative importance of each category label. $l_{\max} < \|\beta\|_1$

		Test	
		Δ_2	0-1
Train	**L**	$\mathbf{0.4371 \pm 0.0045}$	0.9329 ± 0.0097
	0-1	0.5091 ± 0.0023	$\mathbf{0.8320 \pm 0.0074}$
	S	0.4927 ± 0.0067	0.8731 ± 0.0073
	M	0.4437 ± 0.0034	1.0010 ± 0.0080
		Test	
		Δ_3	0-1
Train	**L**	$\mathbf{0.9447 \pm 0.0069}$	0.8786 ± 0.0077
	0-1	0.9877 ± 0.0044	$\mathbf{0.8173 \pm 0.0061}$
	S	0.9784 ± 0.0052	0.8337 ± 0.0054
	M	0.9718 ± 0.0041	0.9425 ± 0.0054

Table 4.3: For the VOC multilabel prediction task, the cross comparison of average loss values (with standard error) using submodular loss as in Equation (4.7.3) and Equation (4.7.4), as well as 0-1 loss during training and during testing.

ensures the function is strictly submodular. This function is an analogue to the submodular increasing function in Sec. 4.4.3. In this VOC experiments, we order the category labels as *person*, *dining table* and *chair*. We set $l_{\max} = 1.3$ and $\beta = [1\ 0.5\ 0.2]$.

We have additionally carried out experiments with another submodular loss function:

$$\Delta_3(y, \tilde{y}) := 1 - \exp\left(-|\mathbf{I}|\right) + \left\langle \beta, \frac{1 - y \odot \tilde{y}}{2} \right\rangle \tag{4.7.4}$$

where $\mathbf{I} = \{j | \tilde{y}^j \neq y^j\}$ is the set of mispredicted labels. The first part, $1 - \exp\left(-|\mathbf{I}|\right)$, is a concave function depending only on the size of $\mathbf{I}$, as a consequence it is a submodular function; the second part is a modular function that penalizes labels proportionate to the coefficient vector β as above. The results with this submodular loss are shown in Table 4.3. We additionally train and test with the 0-1 loss, which is equivalent to an SVM.

We compare different losses employed during training and during testing. $\mathbf{M}$ and $\mathbf{S}$ denote the use of the submodular loss with margin and slack rescaling, respectively. The empirical results with this submodular loss are shown in Table 4.3.

4.7.4 MS COCO Multilabel Prediction

In this section, we consider the multi-label prediction task on Microsoft COCO dataset. On detecting a *dinner scene*, the initial multi-label prediction should emphasize that all labels must be correct *within a given image*, while MS COCO provides more relating categories e.g. *people*, *dinning tables*, *forks* or *cups*. Submodular loss functions coincides with the fact that misprediction of a single label, e.g. *person*, will preclude the chance to predict correctly the combination of all labels. Figure 4.9 shows example images from the Microsoft COCO dataset [Lin et al., 2014].

Figure 4.9: Examples from the Microsoft COCO dataset. Figure 4.9(a) contains all the categories of interest (cf. Section 4.7); Figure 4.9(b) contains *dining table*, *fork* and *cup*; Figure 4.9(c) is not a dining scene but contains *people*.

The Microsoft COCO dataset [Lin et al., 2014] is an image recognition, segmentation, and captioning dataset. It contains more than 70 categories, more than 300,000 images and around 5 captions per image. We have used frequent itemset mining [Uno et al., 2004] to determine the most common combination of categories in an image. For sets of size 6, these are: *person*, *cup*, *fork*, *knife*, *chair* and *dining table*.

For the experiments, we repeatedly sample sets of images containing k ($k = 0, 1, 2, \cdots, 6$) categories. The training/validation set and testing set have the same distribution. For a single iteration, we sample 1050 images for the training/validation set including 150 images each for sets containing k ($k = 0, 1, 2, \cdots, 6$) of the target labels. More than 12,000 images from the entire dataset are sampled at least once as we repeat the experiments to compute statistical significance.

In this task, we first use a submodular loss function as follows:

$$\Delta_4(y, \tilde{y}) := 1 - \exp\left(-\alpha|\mathbf{I}|\right) \tag{4.7.5}$$

where $\mathbf{I} = \{j | y^j \neq \tilde{y}^j\}$ is the set of mispredicted labels. $1 - \exp\left(-|\mathbf{I}|\right)$, is a concave function depending only on the size of $\mathbf{I}$, as a consequence it is a submodular function, α is a positive coefficient that effect the increasing rate of the concave function thus the submodularity of the set function. In the experiment we set $\alpha = 1$ (cf. Table 4.4).

We have also carried out experiments with the submodular loss function used in VOC experiments:

$$\Delta_5(y, \tilde{y}) := 1 - \exp\left(-|\mathbf{I}|\right) + \left\langle \beta, \frac{1 - y \odot \tilde{y}}{2} \right\rangle \tag{4.7.6}$$

where we set $\beta = [1\ 0.8\ 0.7\ 0.6\ 0.5\ 0.4]^T$ according to the size of the object that we order the category labels as *person*, *dining table*, *chair*, *cup*, *fork* and *knife* (cf. Table 4.4).

cluster-label loss In addition, we use another submodular loss, which is a concave over modular function, as follows,

$$\Delta_6(y, \tilde{y}) := \sqrt{m(\mathbf{I})} \tag{4.7.7}$$

where m is a modular function for which the values on each element is defined as

$$m(\{j\}) = \sum_{A \in F} [j \in A]. \tag{4.7.8}$$

where the F is the set of frequent itemsets that has been pre-calculated among training samples. This loss gives a higher penalty for mispredicting clustered labels that frequently co-occur in the training set (cf. Table 4.4). While the diminishing returns property of submodularity yields the correct cluster semantics: if we already have a large set of mispredctions, the joint prediction is already poor and an additional misprediction has a lower marginal cost.

We compare different losses employed during training and during testing. We also train and test with the 0-1 loss, which is equivalent to an SVM. $\mathbf{M}$ and $\mathbf{S}$ denote the use of the submodular loss with margin and slack rescaling, respectively. As this optimization is NP-hard, we have employed the simple application of the greedy approach as is common in (non-monotone) submodular maximization (e.g. [Krause and Golovin, 2014]).

We repeated each experiment 10 times with random sampling in order to obtain an estimate of the average performance. Table 4.4 shows the cross comparison of average loss values (with standard error) using different loss functions during training and during testing for the COCO dataset.

4.7.5 Empirical Results

From the empirical results in Table 4.1, Table 4.2, Table 4.3 and Table 4.4, we can see that training with the same submodular loss functions as used during testing yields the best results. Slack and margin rescaling fail due to the necessity of approximate inference, which results in a poor discriminant function. By contrast, the Lovász hinge yields the best performance when the submodular loss is used to evaluate the test predictions. We do not expect that optimizing the submodular loss should give the best performance when the 0-1 loss is used to evaluate the test predictions. Indeed in this case, the Lovász hinge trained on 0-1 loss corresponds with the best performing system.

Figure 4.10(a) and Figure 4.10(b) show for the two submodular functions the primal-dual gap as a function of the number of cutting-plane iterations using the Lovász hinge with submodular loss, as well as for a SVM (labeled 0-1), and margin and slack rescaling (labeled $\mathbf{M}$ and $\mathbf{S}$). This demonstrates that the empirical convergence of the Lovász hinge is at a rate comparable to an SVM, and is feasible to optimize in practice for real-world problems.

4.8 Discussion

In this work, we have introduced a novel convex surrogate loss function, the Lovász hinge, which makes tractable for the first time learning with submodular loss functions. In contrast

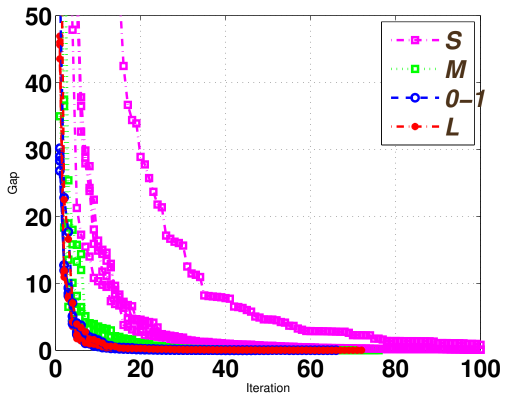
		Test	
		Δ_4	0-1
Train	**L**	$\mathbf{0.5567 \pm 0.0057}$	1.4295 ± 0.0194
	0-1	0.5729 ± 0.0060	$\mathbf{1.3924 \pm 0.0195}$
	S	0.5875 ± 0.0065	1.4940 ± 0.0233
	M	0.5820 ± 0.0063	1.4802 ± 0.0233
		Test	
		Δ_5	0-1
Train	**L**	$\mathbf{1.3767 \pm 0.0143}$	1.3003 ± 0.0176
	0-1	1.3813 ± 0.0135	$\mathbf{1.2975 \pm 0.0152}$
	S	1.4711 ± 0.0153	1.3832 ± 0.0156
	M	1.4811 ± 0.0117	1.4016 ± 0.0136
		Test	
		Δ_6	0-1
Train	**L**	$\mathbf{0.6141 \pm 0.0048}$	1.2504 ± 0.0106
	0-1	0.6224 ± 0.0048	$\mathbf{1.2461 \pm 0.0115}$
	S	0.6700 ± 0.0051	1.3789 ± 0.0150
	M	0.6723 ± 0.0051	1.3787 ± 0.0123

Table 4.4: For the MS COCO prediction task, the comparison of average loss values (with standard error) using the submodular losses as in Equation (4.7.5), Equation (4.7.6) and Equation (4.7.7), as well as 0-1 loss during training and during testing.

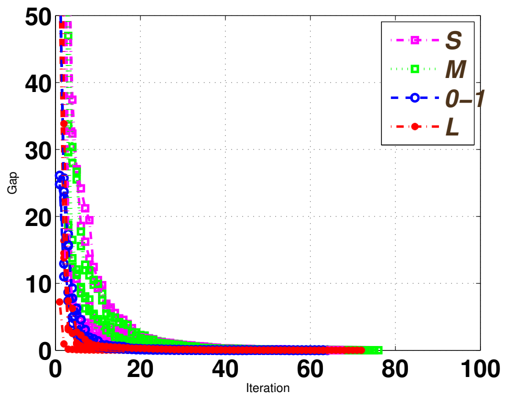
to margin and slack rescaling, computation of the gradient or cutting plane can be achieved in $\mathcal{O}(p \log p)$ time. Margin and slack rescaling are NP-hard to optimize in this case, and furthermore deviate from the convex closure of the loss function (4.1).

We have proven necessary and sufficient conditions for margin and slack rescaling to yield tight convex surrogates to a discrete loss function. These conditions are that the discrete loss be a (properly scaled) increasing function. However, it may be of interest to consider non-increasing functions in some domains. The Lovász hinge can be applied also to non-increasing functions.

We have demonstrated the correctness and utility of the Lovász hinge on different tasks. We have shown that training by minimization of the Lovász hinge applied to multiple submodular loss functions, including the popular Jaccard loss, results in a lower empirical test error than existing methods, as one would expect from a correctly defined convex surrogate. As predicted by the theory, optimizing the submodular loss with the Lovász hinge yields the best performance when evaluating performance using the same loss. Similarly optimizing Hamming loss during training yields the best performance with respect to Hamming loss at test time. Slack and margin rescaling both fail in practice as approximate inference does not yield a good approximation of the discriminant function. The causes of this have been studied in a different context in [Finley and Joachims, 2008], but are effectively due to (i) repeated approximate inference compounding errors, and (ii) erroneous early termination due to underestimation of the primal objective. We empirically observe that the Lovász



(a)



(b)

Figure 4.10: The primal-dual gap as a function of the number of cutting-plane iterations using the Lovász hinge with submodular loss, a SVM (labeled 0-1), and margin and slack rescaling with greedy inference (labeled **M** and **S**). Figure 4.10(a) for the experiment using Equation (4.7.5) and Figure 4.10(b) for Equation (4.7.6). This demonstrates that empirical convergence of the Lovász hinge is at a rate comparable to an SVM, and is feasible to optimize in practice for real-world problems.

hinge delivers much better performance by contrast, and makes no approximations in the subgradient computation. Exact inference should yield a good predictor for slack and margin rescaling, but sub-exponential optimization only exists if P=NP. Therefore, the Lovász hinge is the *only* polynomial time option in the literature for learning with such losses.

The introduction of this novel strategy for constructing convex surrogate loss functions for submodular losses points to many interesting areas for future research. Among them are then definition and characterization of useful loss functions in specific application areas. It is known that an arbitrary set function can be written as the difference of two submodular functions, which leads to the question of optimizing such functions by bounding the submodular and supermodular components by different convex surrogates. Furthermore, theoretical convergence results for a cutting plane optimization strategy are of interest. A result of this kind may imply a convergence result for cutting plane minimization of submodular functions, which is of general interest and a known open problem in submodular optimization [Fujishige, 2005; Schrijver, 2002].

Chapter 5

Convex Surrogate Operator for General Non-modular Loss Functions

Contents

5.1 Introduction **84**
5.2 A Submodular-Supermodular Decomposition **84**
5.3 A Canonical Decomposition **86**
5.4 A Convex Surrogate Operator $\mathbf{B}_\mathbf{D}$ **87**
5.4.1 Definition of $\mathbf{B}_\mathbf{D}$ 87
5.4.2 Properties of $\mathbf{B}_\mathbf{D}$ 88
5.5 Sørensen-Dice loss **91**
5.6 Experimental Results **93**
5.6.1 Dice Loss 93
5.6.2 Face Classification in Video Sequences 93
5.7 Discussion **98**

Overview In this chapter, a novel generic convex surrogate for general non-modular loss functions is introduced, which provides for the first time a tractable solution for loss functions that are neither supermodular nor submodular. This convex surrogate is based on a submodular-supermodular decomposition for which the existence and uniqueness is proven in this chapter. It takes the sum of two convex surrogates that separately bound the supermodular component and the submodular component using slack-rescaling and the Lovász hinge, respectively. It is further proven that this surrogate is convex, piecewise linear, an extension of the loss function, and the subgradient computation is polynomial time. Empirical results are reported on a non-submodular loss based on the Sørensen-Dice difference function, and a real-world face track dataset with tens of thousands of frames, demonstrating the improved performance, efficiency, and scalability of the novel convex surrogate.

The work presented in this chapter is based on the following paper:

- Jiaqian Yu and Matthew B. Blaschko. A convex surrogate operator for general non-modular loss functions. International Conference on Artificial Intelligence and Statistics, volume 51 of Journal of Machine Learning Research, pages 1032–1041, 2016

5.1 Introduction

Alternatives to the Hamming loss are frequently employed in the discriminative learning literature: [Cheng et al., 2010] uses a rank loss which is supermodular; [Petterson and Caetano, 2011] uses a non-submodular loss based on F-score; [Doppa et al., 2014] uses modular losses e.g. Hamming loss and F1 loss which is non-submodular; and losses that are nonmodular are common in a wide range of problems, including Jaccard index based losses [Blaschko and Lampert, 2008; Everingham et al., 2010; Nowozin, 2014], or more general submodular-supermodular objectives [Narasimhan and Bilmes, 2005].

We have demonstrated in previous chapters polynomial time convex surrogates for supermodular losses [Tsochantaridis et al., 2005] and submodular losses [Yu and Blaschko, 2015b], but not for more general non-modular losses. We may perform approximate inference in polynomial time via a greedy optimization procedure to compute a subgradient or cutting plane of a convex surrogate for a general increasing function, but this leads to poor performance of the training procedure in practice [Finley and Joachims, 2008; Joachims et al., 2009]. A decomposition-based method for a general set function has been proposed in the literature [Iyer and Bilmes, 2012], showing that under certain conditions a decomposition into a submodular plus a supermodular function can be efficiently found. Other relevant work includes the hardness results on submodular Hamming optimization and its approximation algorithms [Gillenwater et al., 2015].

In this chapter, we propose a novel convex surrogate for general non-modular loss functions, which is solvable for the first time for non-supermodular and non-submodular loss functions. In Section 5.2, we define a decomposition for a general non-modular loss function into supermodular and submodular components (Section 5.3), propose a novel convex surrogate operator based on this decomposition (Section 5.4), and demonstrate that it is convex, piecewise linear, an extension of the loss function, and for which subgradient computation is polynomial time (Section 5.4.2). In Section 5.5, we introduce the Sørensen-Dice loss, which is neither submodular nor supermodular. In Section 5.6 we demonstrate the feasibility, efficiency and scalability of our convex surrogate with the Sørensen-Dice loss on a synthetic problem, and a range of non-modular losses on a real-world face-track dataset comprising tens of thousands of video frames.

5.2 A Submodular-Supermodular Decomposition

In this section, we will study on a submodular-supermodular decomposition for general set functions. As in previous chapter, we always consider that the loss function as a set function

$\Delta(y, \tilde{y}) = \ell(\{i|y^i \neq \tilde{y}^i\})$.

Definition 5.1. *A set function ℓ is non-negative if $\ell(A) - \ell(\emptyset) \geq 0,\ \forall A \subseteq V$.*

We denote the set of all set functions as $\mathcal{F}$. We consider ℓ is non-negative, which we will denote $\ell \in \mathcal{F}_+$. We denote the set of all submodular functions as $\mathcal{S}$, and the set of all supermodular functions as $\mathcal{G}$.

To recall, we say that a set function ℓ is symmetric if $\ell(A) = c(|A|)$ for some function $c : \mathbb{Z}^* \mapsto \mathbb{R}$; a symmetric set function ℓ is submodular if and only if c is concave [Bach, 2013, Proposition 6.1]; a set function $\ell : \mathcal{P}(V) \mapsto \mathbb{R}$ is increasing if and only if for all subsets $A \subset V$ and elements $x \in V \setminus A$, $\ell(A) \leq \ell(A \cup \{x\})$.

We note that the set of increasing supermodular functions is identical to $\mathcal{G}_+$. We will propose a convex surrogate operator for a general non-negative loss function, based on the fact that set functions can always be expressed as the sum of a submodular function and a supermodular function:

Proposition 5.1. *For all set functions ℓ, there always exists a decomposition into the sum of a submodular function $f \in \mathcal{S}$ and a supermodular function $g \in \mathcal{G}$:*

$$\ell = f + g \tag{5.2.1}$$

A proof of this proposition is given in [Narasimhan and Bilmes, 2005, Lemma 4].

Proposition 5.2. *For an arbitrary decomposition $\ell = f + g$ where g is not increasing, there exists a modular function m_g s.t.*

$$\ell = (f - m_g) + (g + m_g) \tag{5.2.2}$$

with $\tilde{f} := f - m_g \in \mathcal{S}$, and $\tilde{g} := g + m_g \in \mathcal{G}_+$ is increasing.

Proof. Any modular function can be written as

$$m_g(A) = \sum_{j \in A} w_j \tag{5.2.3}$$

for some coefficient vector $w \in \mathbb{R}^{|V|}$ (see Theorem 2.3 in Chapter 2). For each $j \in V$, we may set

$$w_j = -\min_{A \subseteq V} g(A \cup \{j\}) - g(A). \tag{5.2.4}$$

The resulting modular function will ensure that $g + m_g$ is increasing (cf. Definition 2.3 in Chapter 2 Section 2.1). □

This proof indicates that a decomposition $\ell = f + g$ is not-unique due to a modular factor. We subsequently demonstrate that decompositions can vary by more than a modular factor:

Proposition 5.3 (Non-uniqueness of decomposition up to modular transformations.)**.** *For any set function, there exist multiple decompositions into submodular and supermodular components such that these components differ by more than a modular factor:*

$$\begin{aligned}&\exists f_1, f_2 \in \mathcal{S},\ g_1, g_2 \in \mathcal{G}, \textit{ such that,}\\ &(\ell = f_1 + g_1 = f_2 + g_2) \wedge (g_1 + m_{g_1} \neq g_2 + m_{g_2})\end{aligned} \tag{5.2.5}$$

where $\wedge$ denotes "logical and," m_{g_1} and m_{g_2} are constructed as in Equations (5.2.3) *and* (5.2.4).

Proof. Let m be a submodular function that is not modular. For a given decomposition $\ell = f_1 + g_1$, we may construct $f_2 := f_1 + m$ and $g_2 := g_1 - m$. As m is not modular, there is no modular m_1 such that $g_1 - m_1 = g_1 - m = g_2$. □

5.3 A Canonical Decomposition

We will show in this section the unique decomposition for a general non-negative loss starting from any arbitrary submodular-supermodular decomposition, which allows us to define a convex surrogate operator based on such a canonical decomposition.

In this section, we define an operator $\mathbf{D}$ such that $g^* := \mathbf{D}\ell \in \mathcal{G}_+$ is unique and $f^* := \ell - \mathbf{D}\ell \in \mathcal{S}$ is then unique. We have demonstrated in the previous section that we may consider there to be two sources of non-uniqueness in the decomposition $\ell = f + g$: a modular component and a non-modular component related to the curvature of g (respectively f). We define $\mathbf{D}$ such that these two sources of non-uniqueness are resolved using a canonical decomposition $\ell = f^* + g^*$.

Definition 5.2. *We define an operator $\mathbf{D} : \mathcal{F} \mapsto \mathcal{G}_+$ as*

$$\mathbf{D}\ell = \arg\min_{g \in \mathcal{G}_+} \sum_{A \subseteq V} g(A), \quad \textit{s.t. } \ell - g \in \mathcal{S}. \tag{5.3.1}$$

We note that minimizing the values of g will simultaneously remove the non-uniqueness due both to the modular non-uniqueness described in Proposition 5.2, as well as the non-modular non-uniqueness described in Proposition 5.3. We formally prove this in Proposition 5.4.

Proposition 5.4. *$\mathbf{D}l$ is unique for all $\ell \in \mathcal{F}$ that have a finite base set V.*

Proof. We note that the $\arg\min$ in Equation (5.3.1) is equivalent to a linear program: g is uniquely determined by a vector in $\mathbb{R}^{2^{|V|}-1}$ the coefficients of which correspond to $g(A)$ for all $A \in \mathcal{P}(V) \setminus \emptyset$, and we wish to minimize the sum of the entries subject to a set of linear constraints enforcing supermodularity of g, non-negativity of g, and submodularity of $\ell - g$.

From [Mangasarian, 1979, Theorem 2], an LP of the form

$$\min_{x \in \mathbb{R}^d} \quad r^T x \tag{5.3.2}$$

$$\text{s.t.} \quad Cx \geq q \tag{5.3.3}$$

has a unique solution if there is no $y \in \mathbb{R}^d$ simultaneously satisfying

$$C_J y \geq 0, \quad r^T y \leq 0, \quad y \neq 0 \tag{5.3.4}$$

where $J = \{i | C_i x^* = q_i\}$ is the active set of constraints at an optimum x^*. We note that as r is a vector of all ones (cf. Equation (5.3.1)), $r^T y \leq 0$ constrains y to lie in the non-positive orthant. However, as the linear program is minimizing the sum of x subject to lower bounds on each entry of x (e.g. positivity constraints), we know that $C_J y \geq 0$ will bound y to lie in the non-negative orthant. This means, at most, these constraints overlap at $y = 0$, but this is expressly forbidden by the last condition in Equation (5.3.4). $\square$

Although Equation (5.3.1) is a linear programming problem, we do not consider this definition to be constructive in general as the size of the problem is exponential in $|V|$ (see [Iyer and Bilmes, 2012]). However, it may be possible to verify that a given decomposition satisfies this definition for some loss functions of interest. Furthermore, for some classes of set functions, the LP has lower complexity, e.g. for symmetric set functions the resulting LP is of linear size, and loss functions that depend only on the number of false positives and false negatives (such as the Sørensen-Dice loss discussed in Section 5.5) result in a LP of quadratic size.

We finally note that from Equation (3.2.5), for every $\Delta(y, \cdot)$ we may consider its equivalence to a set function $\ell = g^* + f^*$, and denote the resulting decomposition of

$$\Delta(y, \cdot) = \Delta_{\mathcal{G}}(y, \cdot) + \Delta_{\mathcal{S}}(y, \cdot) \tag{5.3.5}$$

into its supermodular and submodular components, respectively.[1]

5.4 A Convex Surrogate Operator $\mathbf{B_D}$

5.4.1 Definition of $\mathbf{B_D}$

In our analysis of convex surrogates for non-modular loss functions, we will employ several results for the Structured Output SVM [Tsochantaridis et al., 2005], which assumes that a structured prediction is made by taking an inner product of a feature representation of inputs and outputs: $\text{sign}(h(x)) = \arg\max_y \langle w, \phi(x, y)\rangle$. In the sequel, we consider a feature function such that $\langle w, \phi(x, y)\rangle = \sum_{j=1}^{p} \langle w^j, x^j \rangle y^j$. Each w^j is then a vector of length d, and $w \in \mathbb{R}^{d \cdot p}$. Therefore p individual prediction functions parametrized by w^j are simultaneously optimized, although we may also consider cases in which we constrain $w^j = w^i \; \forall i, j$. More generally, we may consider $h : \mathcal{X} \mapsto \mathbb{R}^p$, which may have non-linearities, e.g. deep neural networks.

From previous sections, we have defined a unique decomposition $\ell = g^* + f^*$, we will use this decomposition to construct a surrogate $\mathbf{B}\Delta$ that is convex, piecewise linear, an

[1]Note that Δ_S and Δ_G are due to Equation (3.2.5) for f^* and g^* which explicitly depend on $\mathbf{D}$. For simplicity of notation, we will use $\Delta_{\mathcal{G}}$ instead of $\Delta_{\mathbf{D}\mathcal{G}}$, $\Delta_{\mathcal{S}}$ instead of $\Delta_{\mathbf{D}\mathcal{S}}$

extension of Δ, and for which subgradient computation is polynomial time. We construct a surrogate $\mathbf{B}$ by taking the sum of two convex surrogates applied to $\Delta_{\mathcal{G}}$ and $\Delta_{\mathcal{S}}$ independently. These surrogates are slack-rescaling [Tsochantaridis et al., 2005] applied to $\Delta_{\mathcal{G}}$ and the Lovász hinge [Yu and Blaschko, 2015b] applied to $\Delta_{\mathcal{S}}$, following the notion from Chapter 4:

$$\mathbf{S}\Delta_{\mathcal{G}}(y, h(x)) := \max_{\tilde{y}\in\mathcal{Y}} \Delta(y,\tilde{y})\left(1 + \langle h(x), \tilde{y}\rangle - \langle h(x), y\rangle\right), \tag{5.4.1}$$

$$\mathbf{L}\Delta_{\mathcal{S}}(y, h(x)) := \left(\max_{\pi} \sum_{j=1}^{p} s^{\pi_j}\left(\ell\left(\{\pi_1,\cdots,\pi_j\}\right) - \ell\left(\{\pi_1,\cdots,\pi_{j-1}\}\right)\right)\right)_+, \tag{5.4.2}$$

where $(\cdot)_+ = \max(\cdot, 0)$, π is a permutation,

$$s^{\pi_j} = 1 - h^{\pi_j}(x) y^{\pi_j}, \tag{5.4.3}$$

and $h^{\pi_j}(x)$ is the π_jth dimension of $h(x)$.

Definition 5.3 (General non-modular convex surrogate)**.** *For an arbitrary non-negative loss function Δ, we define*

$$\mathbf{B_D}\Delta := \mathbf{L}\Delta_{\mathcal{S}} + \mathbf{S}\Delta_{\mathcal{G}} \tag{5.4.4}$$

where $\Delta_{\mathcal{S}}$ and $\Delta_{\mathcal{G}}$ are as in Equation (5.3.5), and $\mathbf{D}$ is the decomposition of ℓ defined by Definition 5.2.

We use a cutting plane algorithm to solve the max-margin problem as shown in Algorithm 5. This is a variant of the classical cutting plane algorithm used in the structured output SVM. At Line 5 and Line 5, instead of finding one single cutting plane by margin rescaling constraints or slack rescaling constraints (c.f. Chapter 2, Algorithm 1 and Algorithm 2), we propose the strategy that finding two cutting planes, i.e. finding the most violated constraints by slack rescaling and Lovász hinge with the decomposed supermodular function and the submodular function, respectively. Then we sum up these two cutting planes so as to form the expected cutting plane for the original regularized optimization problem.

5.4.2 Properties of $\mathbf{B_D}$

In the remainder of this section, we show that $\mathbf{B_D}$ has many desirable properties. Specifically, we show that $\mathbf{B_D}$ is closer to the convex closure of the loss function than slack rescaling and that it generalizes the Lovász hinge (Theorems 5.1 and 5.2). Furthermore, we formally show that $\mathbf{B_D}\Delta$ is convex (Theorem 5.3), an extenstion of Δ for a general class of loss functions (Theorem 5.4), and polynomial time computable (Theorem 5.5).

Lemma 5.1. *If $\ell \in \mathcal{G}$, then $f^* := \ell - \mathbf{D}\ell \in \mathcal{S}\cap\mathcal{G}$ i.e. modular.*

Algorithm 5 Cutting plane algorithm

```
1: Input: $(x_1, y_1), \cdots, (x_n, y_n), C, \epsilon$
2: $S^i = \emptyset, \forall i = 1, \cdots, n$
3: repeat
4:   for $i = 1, \cdots, n$ do
5:     $\hat{y}_L = \arg\max_{\tilde{y}} H_L(y_i) = \arg\max_{\tilde{y}} \mathbf{L}\Delta_{\mathcal{S}}$
6:     $\hat{y}_S = \arg\max_{\tilde{y}} H_S(y_i) = \arg\max_{\tilde{y}} \mathbf{S}\Delta_{\mathcal{G}}$
7:     $H(\hat{y}) = H_L(\hat{y}_L) + H_S(\hat{y}_S)$
8:     $\xi^i = \max\{0, H(y_i)\}$
9:     if $H(\hat{y}) > \xi^i + \epsilon$ then
10:      $S^i := S^i \cup \{y_i\}$
11:      $w \leftarrow$ optimize Equation (4.3.3) with constraints defined by $\cup_i S^i$
12:    end if
13:  end for
14: until no $S^i$ has changed during an iteration
15: return $(w, \xi)$
```

Proof. First we set $\ell = g_m + f_m$ where f_m is modular and $f_m(\{j\}) = \ell(\{j\})$. Then for any subset $S \subseteq V$ we have

$$f_m(S) = \sum_{j \in S} \ell(\{j\}) \tag{5.4.5}$$

$$g_m(S) = \ell(S) - \sum_{j \in S} \ell(\{j\}) \tag{5.4.6}$$

$$\sum_{S \subseteq V} g_m(S) = \sum_{S \subseteq V} \left(\ell(S) - \sum_{j \in S} \ell(\{j\}) \right). \tag{5.4.7}$$

The sum in Equation (5.4.7) is precisely the sum that should be minimized in Equation (5.3.1). We now show that this sum cannot be minimized further while allowing f_m to be non-modular. If there exists any g s.t. $f := \ell - g$ is submodular but not modular, by definition there exists at least one subset $S_s \subseteq V$ and one $j \in S_s$ such that

$$f(S_s \setminus \{j\}) + f(\{j\}) > f(S_s) + f(\emptyset). \tag{5.4.8}$$

Then by subtracting each time one element from the subset S_s, we have

$$\begin{aligned} f(S_s) &< f(S_s \setminus \{j\}) + f(\{j\}) \\ &\leq f(S_s \setminus (\{j\} \cup \{k\})) + f(\{k\}) + f(\{j\}) \\ &\leq \cdots \leq \sum_{j \in S_s} f(\{j\})), \qquad \forall k \in S_s \setminus (\{j\} \end{aligned} \tag{5.4.9}$$

which implies

$$g(S_s) > l(S_s) - \sum_{j \in S_s} \ell(\{j\}) \tag{5.4.10}$$

By taking the sum of the inequalities as in Equation (5.4.10) for all subsets S, we have that

$$\sum_{S\subseteq V} g(S) > \sum_{S\subseteq V}\left(\ell(S) - \sum_{j\in S}\ell(\{j\})\right) = \sum_{S\subseteq V} g_m(S)$$

which means $\sum_{S\subseteq V} g(S) > \sum_{S\subseteq V} g_m(S)$ for any g. By Definition 5.2, $g^* = g_m = \mathbf{D}\ell$, thus $f^* := \ell - \mathbf{D}\ell = f_m$ is modular. $\square$

Lemma 5.2. *For a loss function Δ such that $\Delta_{\mathcal{S}}$ is increasing, we have*

$$\mathbf{S}\Delta_{\mathcal{G}} = \mathbf{S}(\Delta - \Delta_{\mathcal{S}}) = \mathbf{S}\Delta - \mathbf{S}\Delta_{\mathcal{S}}. \tag{5.4.11}$$

Proof. By Equation (5.4.1), for every single cutting plane determined by some $\tilde{y}$, we have

$$\begin{aligned}
&\mathbf{S}\left(\Delta(y,\tilde{y}) - \Delta_{\mathcal{S}}(y,\tilde{y})\right)\\
&= \left(\Delta(y,\tilde{y}) - \Delta_{\mathcal{S}}(y,\tilde{y})\right)\left(1 + \langle h(x),\tilde{y}\rangle - \langle h(x),y\rangle\right)\\
&= \Delta(y,\tilde{y})\left(1 + \langle h(x),\tilde{y}\rangle - \langle h(x),y\rangle\right) - \Delta_{\mathcal{S}}(y,\tilde{y})\left(1 + \langle h(x),\tilde{y}\rangle - \langle h(x),y\rangle\right)\\
&= \mathbf{S}\Delta(y,\tilde{y}) - \mathbf{S}\Delta_{\mathcal{S}}(y,\tilde{y}).
\end{aligned} \tag{5.4.12}$$

As this property holds for all cutting planes, it also holds for the supporting hyperplanes that define the convex surrogate and $\mathbf{S}(\Delta - \Delta_{\mathcal{S}}) = \mathbf{S}\Delta - \mathbf{S}\Delta_{\mathcal{S}}$. $\square$

For reminder, we use the definition of an extension as defined in Chapter 4:

Definition 5.4. *A convex surrogate function $\mathbf{B}\Delta(y,\cdot)$ is an extension when*

$$\mathbf{B}\Delta(y,\cdot) = \Delta(y,\cdot) \tag{5.4.13}$$

on the vertices of the 0-1 unit cube under the mapping to $\mathbb{R}^p$: $i = \{1,\dots,p\}$, $[u]^i = 1 - h^{\pi_i}(x)y^{\pi_i}$

Theorem 5.1. *If $\ell \in \mathcal{G}_+$, then $\mathbf{B_D}\Delta \geq \mathbf{S}\Delta$ over the unit cube given in Definition 5.4, and therefore $\mathbf{B_D}$ is closer to the convex closure of Δ than $\mathbf{S}$.*

Proof. By the definition of the Lovász hinge $\mathbf{L}$ [Yu and Blaschko, 2015b], we know that for any modular function $\Delta_{\mathcal{S}}$ we have $\mathbf{L}\Delta_{\mathcal{S}} \geq \mathbf{S}\Delta_{\mathcal{S}}$ over the unit cube. As a result of Lemma 5.1 and Lemma 5.2,we have

$$\begin{aligned}
\mathbf{B_D}\Delta &= \mathbf{S}\Delta_{\mathcal{G}} + \mathbf{L}\Delta_{\mathcal{S}} = \mathbf{S}(\Delta - \Delta_{\mathcal{S}}) + \mathbf{L}\Delta_{\mathcal{S}}\\
&\geq \mathbf{S}\Delta - \mathbf{S}\Delta_{\mathcal{S}} + \mathbf{S}\Delta_{\mathcal{S}} = \mathbf{S}\Delta.
\end{aligned}$$

$\square$

Theorem 5.2. *If $\ell \in \mathcal{S}$, then $\mathbf{B_D}\Delta = \mathbf{L}\Delta$*

Proof. For $\ell \in \mathcal{S}$, we construct $g^* = \mathbf{0}$, and $f^* = \ell$ is submodular. By Definition 5.2, $g^*(V)$ is minimum, so $g^* = \mathbf{D}\ell$. Then $\mathbf{B_D}\Delta = \mathbf{L}\Delta + \mathbf{S0} = \mathbf{L}\Delta$ □

Theorem 5.3. $\mathbf{B_D}\Delta$ *is convex for arbitrary* Δ.

Proof. By Definition 5.2, $\mathbf{B_D}\Delta$ is the sum of the two convex surrogates, which is a convex surrogate. □

Theorem 5.4. $\mathbf{B_D}\Delta$ *is an extension of* Δ *if and only if* $\Delta_{\mathcal{S}}$ *is non-negative.*

Proof. From [Yu and Blaschko, 2015b, Proposition 1], $\mathbf{S}\Delta$ is an extension for any supermodular increasing Δ; $\mathbf{L}\Delta$ is an extension if and only if Δ is submodular and non-negative as in this case, $\mathbf{L}$ coincides with the Lovász extenstion [Lovász, 1983]. By construction from Definition 5.2 we have $\Delta_{\mathcal{G}}$ and $\Delta_{\mathcal{S}}$ for $g \in \mathcal{G}_+$ and $f \in \mathcal{S}$, respectively. Thus Equation (5.4.13) holds for both $\mathbf{S}\Delta_{\mathcal{G}}$ and $\mathbf{L}\Delta_{\mathcal{S}}$ if $\Delta_{\mathcal{S}}$ is non-negative. Then $\mathbf{B_D}\Delta$ taking the sum of the two extensions, Equation (5.4.13) also holds for every vertex of the unit cube as $\Delta = \Delta_{\mathcal{G}} + \Delta_{\mathcal{S}}$, which means $\mathbf{B_D}$ is also an extension of Δ . □

Theorem 5.5. *The subgradient computation of* $\mathbf{B_D}\Delta$ *is polynomial time given polynomial time oracle access to* f^* *and* g^*.

Proof. Given f^* and g^* we know that the subgradient computation of $\mathbf{L}\Delta_{\mathcal{S}}$ and $\mathbf{S}\Delta_{\mathcal{G}}$ are each polynomial time. Thus taking the sum of the two is also polynomial time. □

5.5 Sørensen-Dice loss

The Sørensen-Dice criterion [Dice, 1945; Sørensen, 1948] is a popular criterion for evaluating diverse prediction problems such as image segmentation [Sabuncu et al., 2010] and language processing [Rychlỳ, 2008]. In this section, we introduce the Sørensen-Dice loss based on the Sørensen-Dice coefficient. We prove that the Sørensen-Dice loss is neither supermodular nor submodular, and we will show in the experimental results section that our novel convex surrogate can yield improved performance on this measure.

Definition 5.5 (Sørensen-Dice Loss)**.** *Denote* $P_y \subseteq V$ *the set of positive labels, e.g. foreground pixels, the Sørensen-Dice loss on given a groundtruth* y *and a predicted output* $\tilde{y}$ *is defined as*

$$\Delta_D(y, \tilde{y}) = 1 - \frac{2|P_y \cap P_{\tilde{y}}|}{|P_y| + |P_{\tilde{y}}|}. \tag{5.5.1}$$

Proposition 5.5. $\Delta_D(y, \tilde{y})$ *is neither submodular nor supermoduler under the isomorphism* $(y, \tilde{y}) \to A := \{i | y^i \neq \tilde{y}^i\}$, $\Delta_J(y, \tilde{y}) \cong \ell(A)$.

We will use the diminishing returns definition of submodularity in Definition 2.1 to first prove the following lemma:

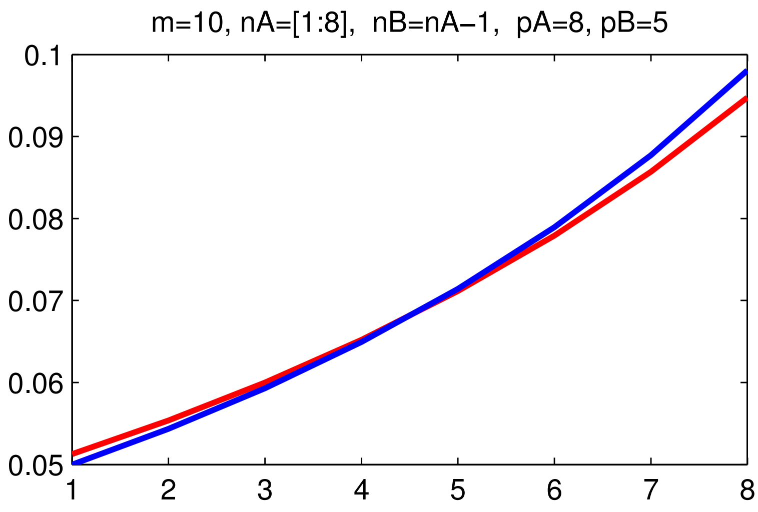


Figure 5.1: Plots of Equation (5.5.4) (red) and Equation (5.5.5) (blue) as a function of n_A. As these two plots cross, neither function bounds the other.

Lemma 5.3. *Δ_D restricted to false negatives is neither submodular nor supermoduler.*

Proof. With the notation $m := |P_y| > 0$, $p := |P_{\tilde{y}} \setminus P_y|$, and $n := |P_y \setminus P_{\tilde{y}}|$, we have that

$$\Delta_D(y, \tilde{y}) = 1 - \frac{2m - 2n}{2m - n + p} = \frac{n + p}{2m - n + p} \tag{5.5.2}$$

For a given groundtruth y i.e. m, we have if $B \subseteq A$, then $n_B \leq n_A$, and $p_B \leq p_A$.

Considering i is an extra false negative, we calculate the marginal gain on A and B respectively:

$$\Delta_D(A \cup \{i\}) - \Delta_D(A)$$

$$= \frac{n_A + 1 + p_A}{2m - n_A - 1 + p_A} - \frac{n_A + p_A}{2m - n_A + p_A} \tag{5.5.3}$$

$$= \frac{2m + 2p_A}{(2m - n_A + p_A - 1)(2m - n_A + p_A)} \tag{5.5.4}$$

$$\Delta_D(B \cup \{i\}) - \Delta_D(B)$$

$$= \frac{2m + 2p_B}{(2m - n_B + p_B - 1)(2m - n_B + p_B)}. \tag{5.5.5}$$

Numerically, we have following counter examples which prove that Δ_D restricted to false negatives is neither submodular nor supermodular. We set $m = 10$, $n_A = [1 : 8]$, $n_B = n_A - 1 \leq n_A$, $p_A = 8$, $p_B = 5 \leq p_A$, and we plot the values of Equation (5.5.4) and Equation (5.5.5) as a function of n_A. We can see from Figure 5.1 that there exists a cross point between these two plots, which indicates that submodularity (Definition 2.1) does not hold for Δ_D or its negative. □

Lemma 5.3 implies Proposition 5.5 as the restriction of a submodular function is itself submodular.

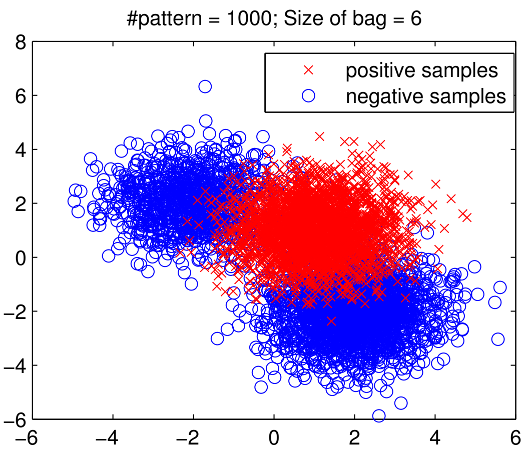


Figure 5.2: The data for the synthetic problem. The negative samples are drawn from a mixture of Gaussians.

5.6 Experimental Results

We demonstrate the correctness and feasibility of the proposed convex surrogate on experiments using Dice loss, as well as on a face classification problem from video sequences with a family of non-modular losses.

5.6.1 Dice Loss

We test the proposed surrogate on a binary set prediction problem. Two classes of 2-dimensional data are generated by different Gaussian mixtures as shown in Fig 5.2. We use the $\mathbf{B_D}$ during training time with the non-modular loss Δ_D to construct a convex surrogate. We compare it to slack rescaling $\mathbf{S}$ with an approximate optimization procedure based on greedy maximization. We additionally train an SVM (denoted 0-1 in the results table) for comparison. During test time, we evaluate with Δ_D and with Hamming loss to calculate the empirical error values as shown in Table 5.1.

We can see from the result that training Δ_D with $\mathbf{B_D}$ yields the best result while using Δ_D during test time. $\mathbf{B_D}$ performs better than $\mathbf{S}$ in both cases due to the failure of the approximate maximization procedure necessary to maintain computational feasibility [Krause and Golovin, 2014].

5.6.2 Face Classification in Video Sequences

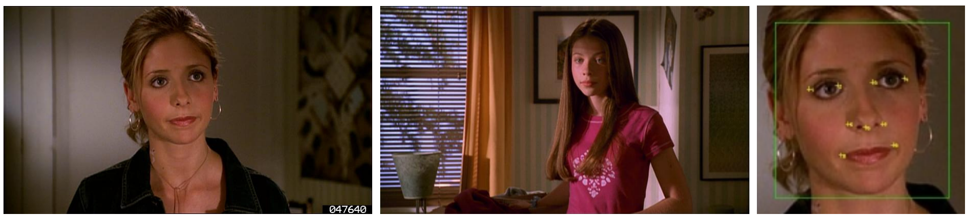
We also evaluate the proposed convex surrogate operator on a real-world face track dataset [Everingham et al., 2006, 2009; Sivic et al., 2009]. The frames of the dataset are from the TV series “Buffy the Vampire Slayer”. This dataset contains 1437 tracks and 27504 frames in total.

	Test	
$p = 6$	Δ_D	0-1
$\mathbf{B_D}$	$\mathbf{0.1121 \pm 0.0040}$	0.6027 ± 0.0125
0-1	0.1497 ± 0.0046	0.5370 ± 0.0114
S	0.3183 ± 0.0148	0.7313 ± 0.0209

Table 5.1: For the synthetic data experiment, the cross comparison of average loss values (with standard error) using different surrogate operations during training, and different evaluation functions during test time. Δ_D is the Dice loss as in Equation (5.5.1).

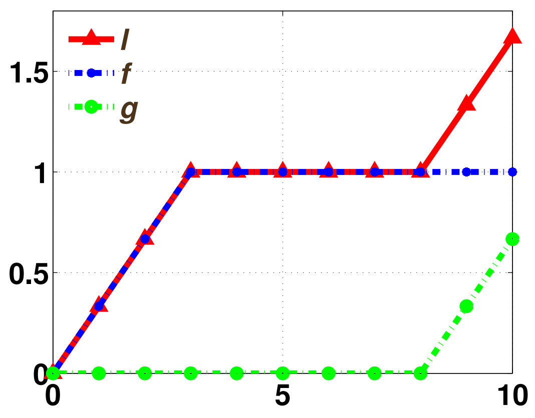
	loss functions			
	Δ_1	Δ_2	Δ_3 (Δ_S negative)	Δ_4 (Δ_S negative)
$\mathbf{B_D}$	$\mathbf{0.194 \pm 0.006}$	$\mathbf{0.238 \pm 0.008}$	0.148 ± 0.005	0.108 ± 0.004
0-1	0.228 ± 0.007	0.284 ± 0.004	0.144 ± 0.004	0.107 ± 0.003
S	0.398 ± 0.015	0.243 ± 0.005	0.143 ± 0.006	0.106 ± 0.003

Table 5.2: For the face classification task, the cross comparison of average loss values (with standard error) using different surrogate operator and losses as in Equation (5.6.1) to Equation (5.6.4) during training, respectively. For the cases that the submodular component is non-negative, i.e. using Δ_1 and Δ_2, the lowest empirical error is achieved when using $\mathbf{B_D}$.

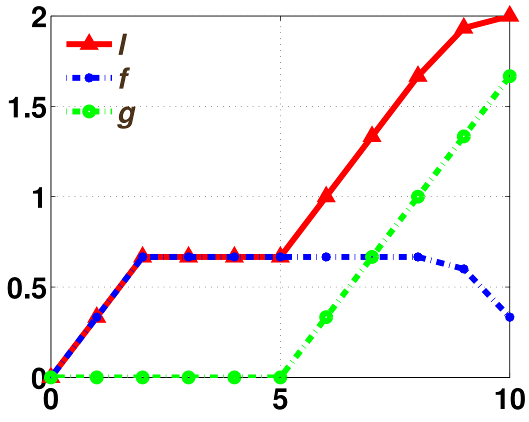


(a) (b) (c)

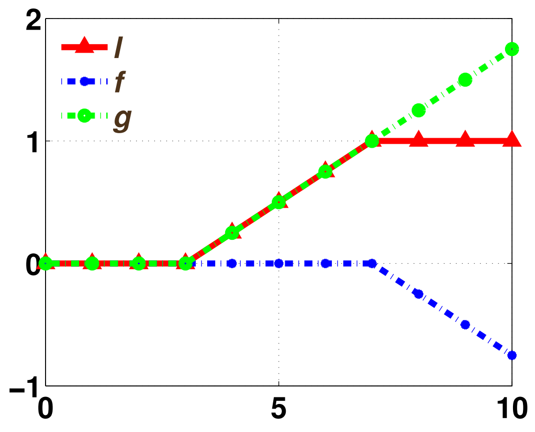
Figure 5.3: Examples of the face track images. Figure 5.3(a) shows the "Buffy" role thus a positive-labelled image and Figure 5.3(b) shows a negative-labelled image. An automated pipeline described in Everingham et al. [2006, 2009]; Sivic et al. [2009] was used for feature extraction (Figure 5.3(c)).



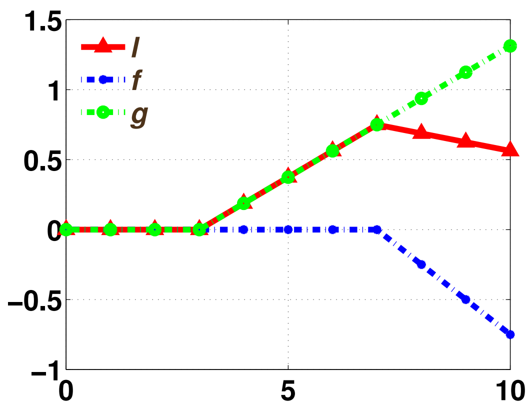
(a) Δ_1



(b) Δ_2



(c) Δ_3



(d) Δ_4

Figure 5.4: The plot of the four loss functions used in our experiments as in Equations (5.6.1) to (5.6.4). The x axis is the number of mispredictions for each track (we show here the loss functions corresponding to track length equal to 10 as an example), and the y axis is the value of loss function. The original losses are drawn in red; the supermodular components are drawn in green, and the submodular components in blue.

We focus on a binary classification task to recognize the leading role: "Buffy" is positive-labelled, "not Buffy" is negative-labelled. Example images are shown in Figure 5.3. Each track is represented as a bag of frames, for which the size of the tracks varies from 1 frame to more than 100 frames, and each image is represented as a Fisher Vector Face descriptor of dimension 1937.

Loss functions We have used different non-supermodular and non-submodular loss functions in our experiments as shown in Equations (5.6.1) to (5.6.4):

$$\Delta_1(y,\tilde{y}) = \min\left(|\mathbf{I}|, \frac{|y|}{3}, |\mathbf{I}| - \frac{|y|}{3}\right) \tag{5.6.1}$$

$$\Delta_2(y,\tilde{y}) = \min\left(|\mathbf{I}|, \frac{|y|}{4}, |\mathbf{I}| - \frac{|y|}{4}, \alpha\right) \tag{5.6.2}$$

$$\Delta_3(y,\tilde{y}) = \min\left(\max\left(0, |\mathbf{I}| - \frac{|y|}{3}\right), \frac{|y|}{3}\right) \tag{5.6.3}$$

$$\Delta_4(y,\tilde{y}) = \min\left(\max\left(0, |\mathbf{I}| - \frac{|y|}{3}\right), \alpha\right) \tag{5.6.4}$$

$|y|$ gives the length of each sequence, $\mathbf{I} = \{i | y^i \neq \tilde{y}^i\}$ gives the set of incorrect prediction elements; α is a parameter that allows us to define the value of $\ell(V)$. Due to the fact that the size of the tracks varies widely, we further normalize the loss function with respect to the track size. We use $\alpha = 2$ for Δ_2 and $\alpha = 0.5$ for Δ_4 in the experiments.

As we can see explicitly in Figure 5.4, no Δ is supermodular or submodular (plotted in red, with legends ℓ). Δ_1, Δ_2 and Δ_3 are increasing loss functions, while Δ_4 is non-increasing. For Δ_1 and Δ_2, we notice that the values of the set functions on a single element are non-zero i.e. $\ell_1(\{j\}) > 0,\ \ell_2(\{j\}) > 0,\ \forall j \in V$; while for the loss Δ_3 and Δ_4 these values are zero i.e. $\ell_3(\{j\}) = \ell_4(\{j\}) = 0,\ \forall j \in V$.

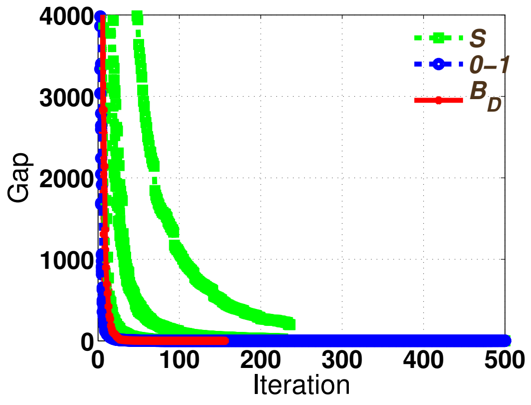
These non-modular functions are motivated by the fact that in a sequence of frames, the roles that appear in the scene should maintain certain continuity, thus the penalization on the misprediction doesn't need to be linear with respective to the size of misprediction, which is equivalent to have different submodularity of the loss function that is to be minimized.

Figure 5.4 shows the corresponding decomposition of each loss into the supermodular componendts (plotted in green, with legends g) and submodular components (plotted in blue, with legends f) as specified in Definition 5.2. We denote each loss function as $\ell_k = f_k + g_k$, for $k = \{1,2,3,4\}$.

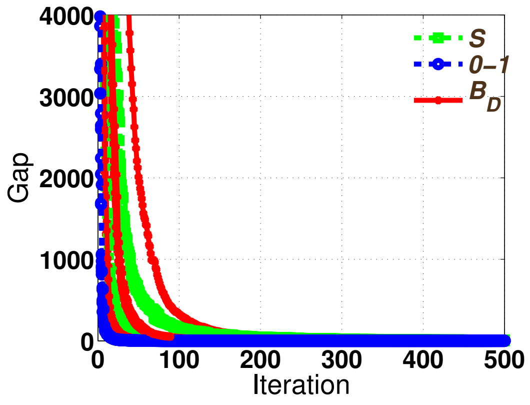
By construction, all supermodular g_k, for $k = \{1,2,3,4\}$, are non-negative increasing. For the submodular component, f_1 is non-negative increasing, f_2 is non-negative and non-increasing, while f_3 and f_4 are both non-positive decreasing.

We compare different convex surrogates during training for these non-modular functions. And we additionally train on the Hamming loss (labelled 0-1) as a comparison. As training non-supermodular loss with slack rescaling is NP-hard, we have employed the simple application of the greedy approach as in Krause and Golovin [2014].

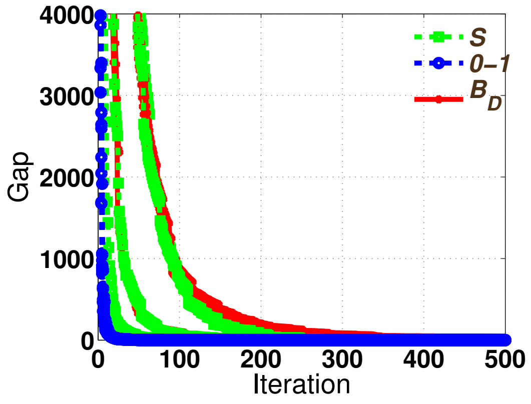
Empirical results For each experiment, we use 30% of the dataset for training, 30% for validation and the rest for testing. We retrain on the training/validation set and finally test on the testing dataset. 10-fold-cross-validation has been carried out and we obtain an average performance and standard error as shown in Table 5.2.



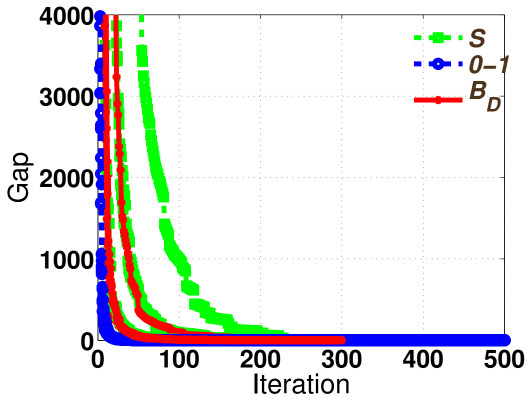
(a) The gap using Δ_1



(b) The gap using Δ_2



(c) The gap using Δ_3



(d) The gap using Δ_4

Figure 5.5: The primal-dual gap as a function of the number of cutting-plane iterations using different convex surrogates for the four non-modular functions in Equations (5.6.1) to (5.6.4). The primal-dual gap from $\mathbf{B_D}$ is drawn in red; the gap from $\mathbf{S}$ is drawn in green and gap from Hamming loss (labelled 0-1, and equivalent to a SVM) in blue. Our convex surrogate operator $\mathbf{B_D}$ can achieve a comparable convergence rate to an SVM, demonstrating that optimization is very fast in practice and the method scales well to large datasets.

	$p = 10$	$p = 50$	$p = 100$
$\mathbf{B_D}$	0.002 ± 0.000	0.018 ± 0.003	0.060 ± 0.008
$\mathbf{S}$	0.002 ± 0.000	0.016 ± 0.002	0.057 ± 0.002

Table 5.3: The comparison of the computation time (s) for one loss augmented inference.

From Table 5.2 we can see that when the submodular component of the decomposition is non-negative, i.e. in the case of using Δ_1 and Δ_2, the lowest empirical error is achieved by using our convex surrogate operator $\mathbf{B_D}$.

Figure 5.5 shows the primal-dual gap as a function of the cutting plane iterations for each experiment using different loss functions and different convex surrogate operators (more than one training procedure are plotted). We can see that in all cases, the convergence of $\mathbf{B_D}$ is at a rate comparable to an SVM, supporting the wide applicability and scalability of the convex surrogate. We have also compared the expected time of one loss augmented inference. Table 5.3 shows the comparison using Δ_1 with $\mathbf{B_D}$ and $\mathbf{S}$. As the cost per iteration is comparable to slack-rescaling, and the number of iterations to convergence is also comparable, there is consequently no computational disadvantage to using the proposed framework, while the statistical gains are significant.

5.7 Discussion

The experiments have demonstrated that the proposed convex surrogate is efficient, scalable, and reduces test time error for a range of loss functions, including the Sørensen-Dice loss, which is a popular evaluation metric in many problem domains. We see that slack rescaling with greedy inference can lead to poor performance for non-supermodular losses. This is especially apparent for the results of training with Δ_1, in which the test-time loss was approximately double that of the proposed method. Similarly, ignoring the loss function and simply training with 0-1 loss can lead to comparatively poor performance, e.g. Δ_1 and Δ_2. This clearly demonstrates the strengths of the proposed method for non-modular loss functions for which a decomposition with a non-negative submodular component is possible (Δ_1 and Δ_2, but not Δ_3 or Δ_4). The characterization and study of this family of loss functions is a promising avenue for future research, with implications likely to extend beyond empirical risk minimization with non-modular losses as considered in this paper. The primal-dual convergence results empirically demonstrate that the loss function is feasible to apply in practice, even on a dataset consisting of tens of thousands of video frames. The convex surrogate is directly amenable to other optimization techniques, such as stochastic gradient descent [Bottou and Bousquet, 2008], or Frank-Wolfe approaches [Lacoste-Julien et al., 2013], as well as alternate function classes including neural networks.

In this work, we have introduced a novel convex surrogate for general non-modular loss functions. We have defined a decomposition for an arbitrary loss function into a supermodular non-negative function and a submodular function. We have proved both the existence and the uniqueness of this decomposition. Based on this decomposition, we have proposed a novel convex surrogate operator taking the sum of two convex surrogates that separately bound the supermodular component and the submodular component using slack-rescaling and the Lovász hinge, respectively. We have demonstrated that our new operator is a tighter approximation to the convex closure of the loss function than slack rescaling, that it generalizes the Lovász hinge, and is convex, piecewise linear, an extension of the loss function, and for which subgradient computation is polynomial time. Open-source code of ℓ_2 regularized risk minimization with this operator is available for download from `https://github.com/yjq8812/aistats2016`.

Chapter 6

Conclusion

Contents

6.1 Contributions 101
6.2 Publication list 104

6.1 Contributions

This thesis studies the interface between discrete optimization, combinatorial optimization and continuous optimization, in particular using statistical learning theory for structured prediction problems. Non-modularity is one of the fundamental analysis tools in structured prediction. Viewing the outcome as a joint set prediction is then essential so as to better incorporate real-world circumstances.

In this thesis, we proposed tractable and efficient methods for dealing with supermodular loss functions, submodular loss functions as well as in general non-modular loss functions, with correctness and scalability validated by empirical results. A list of non-modular loss functions that have been derived in this thesis is shown in Table 6.1. The structure of the contributions of this thesis is shown in Figure 6.1.

Supermodular loss functions First, in Chapter 3, we proposed a novel supermodular loss functions that penalizes more on the neighboring pixels that are both mispredicted. We applied a 8-connected graph structure for which the isomorphism to a supermodular function is not the same as from the inference term. More generally, the structure of the loss function is not necessary the same as the inference term, thus it will be non-trivial to solve the loss augmented inference problem. In this work, we proposed an ADMM-based framework to solve the loss augmented inference problem that only depends on two individual solvers for the loss function term and for the inference term as two independent subproblems. In this way, we can achieve more flexibility in choosing our loss function of interest. For an image segmentation task, we showed that the novel supermodular loss function can empirically

Loss Functions	**Non-modularity**	**Chapter**
8-connected loss: $\Delta(y,\tilde{y}) = \sum_{j=1}^{p}[y^j \neq \tilde{y}^j] + \sum_{(k,l)\in\mathcal{E}_\ell} \gamma[y^k \neq \tilde{y}^k \wedge y^l \neq \tilde{y}^l];$ Square loss: $\Delta(y,\tilde{y}) = \left(\sum_{j=1}^{p}[y^j \neq \tilde{y}^j]\right)^2;$	supermodular	Chapter 3
$\Delta(y,\tilde{y}) = \sum_{i=1}^{p} e^{-i} \min\left(\sum_{j=1}^{i}[y^j \neq \tilde{y}^j],\ \frac{i}{2}\right);$ $\Delta(y,\tilde{y}) = \min\left(l_{\max}, \left\langle \beta, \frac{1-y\odot\tilde{y}}{2} \right\rangle\right),\ l_{\max} < \|\beta\|_1\ ;$ $\Delta(y,\tilde{y}) = 1 - \exp\left(-\alpha \sum_{j=1}^{i}[y^j \neq \tilde{y}^j]\right);$ $\Delta(y,\tilde{y}) = 1 - \exp\left(-\sum_{j=1}^{i}[y^j \neq \tilde{y}^j]\right) + \left\langle \beta, \frac{1-y\odot\tilde{y}}{2} \right\rangle;$ Jaccard loss: $\Delta_J(y,\tilde{y}) = 1 - \frac{\lvert P_y \cap P_{\tilde{y}} \rvert}{\lvert P_y \cup P_{\tilde{y}} \rvert};$ cluster-label loss: $\Delta(y,\tilde{y}) = \sqrt{m\left(\{j \vert y^j \neq \tilde{y}^j\}\right)}, m(\{j\}) = \sum_{A\in F}[j \in A];$	submodular	Chapter 4
$\Delta(y,\tilde{y}) = \min(\sum_{j=1}^{p}[y^j \neq \tilde{y}^j], \lvert y\rvert/3, \sum_{j=1}^{p}[y^j \neq \tilde{y}^j]\rvert - \lvert y\rvert/3);$ $\Delta(y,\tilde{y}) = \min(\sum_{j=1}^{p}[y^j \neq \tilde{y}^j], \lvert y\rvert/4, \sum_{j=1}^{p}[y^j \neq \tilde{y}^j] - \lvert y\rvert/4, \alpha)\ ;$ $\Delta(y,\tilde{y}) = \min(\max(0, \sum_{j=1}^{p}[y^j \neq \tilde{y}^j] - \lvert y\rvert/3), \lvert y\rvert/3);$ $\Delta(y,\tilde{y}) = \min(\max(0, \sum_{j=1}^{p}[y^j \neq \tilde{y}^j] - \lvert y\rvert/3), \alpha);$ Dice loss: $\Delta_D(y,\tilde{y}) = 1 - \frac{2\lvert P_y \cap P_{\tilde{y}} \rvert}{\lvert P_y\rvert + \lvert P_{\tilde{y}} \rvert};$	neither submodular nor supermodular	Chapter 5

Table 6.1: A summary of loss functions that appeared in this thesis, on given a groundtruth y and a predicted output $\tilde{y}$, α, β, γ and $l_{\max}$ are real-valued parameters, F is the set of frequent itemsets.

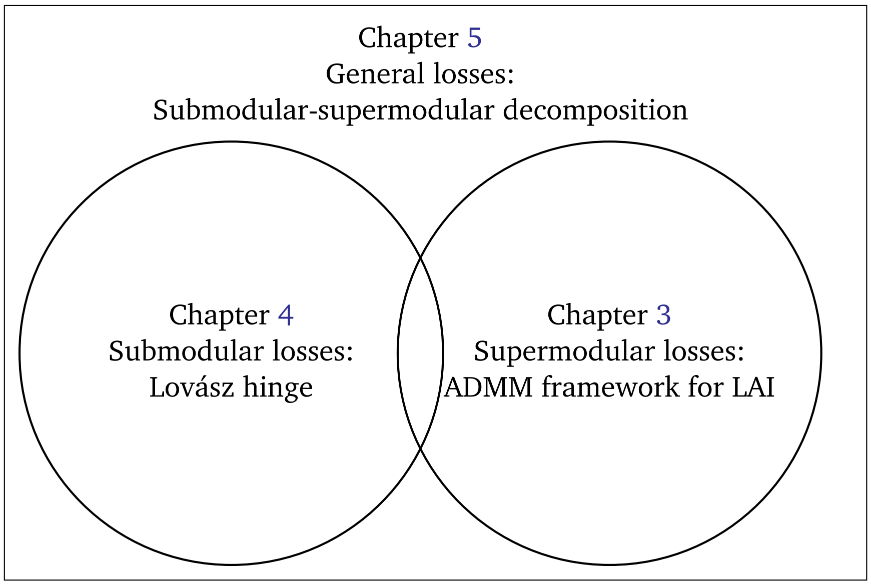


Figure 6.1: The structure of the contribution of this thesis in a Venn diagram related to the submodularity of the loss functions.

achieve better performance on the boundary of the objects, finding elongated structures such as *legs* and *heads*, which we have validated on a binary segmentation task.

Submodular loss functions Second, in Chapter 4, we proposed a novel convex surrogate loss function, the Lovász hinge, which provides for the first time in the literature a polynomial-time feasible solution for training with submodular loss functions. The computation of the gradient or cutting plane can be achieved in $\mathcal{O}(p \log p)$ time. We showed that the Jaccord loss which is based on the intersection over union score, a popular evaluation criterion in computer vision tasks, is a submodular function with respect to the set of mispredictions. We have proven the correctness and feasibility of the Lovász hinge in learning different submodular loss functions, including Jaccard loss, a cluster-label loss and others, on various tasks, including multilabel prediction tasks on the Pascal VOC and the MS COCO datasets. We showed that for submodular loss functions, training with the Lovász hinge achieves lower empirical error value than margin rescaling and slack rescaling, which is expected from a correctly defined convex surrogate.

General non-modular loss functions Third, in Chapter 5, based on the developments in the previous chapters, we further extended our work to deal with general non-modular loss functions. Given a loss function that is neither submodular nor supermodular, we proposed to decompose this function into an unique pair of a submodular function plus a supermodular function, under the constraint that the supermodular function has minimum values. We have proven formally the existence and uniqueness of this constrained decomposition.

Based on this, we are able to introduce a novel surrogate function, that uses slack rescaling on the supermodular function, and the Lovász hinge on the submodular function. We are unaware of previous methods in the literature that provide a convex surrogate to such a wide range of non-modular loss functions. In addition, we have proven that the Dice loss, which is defined based on the Dice score, a popular evaluation criterion in computer vision especially in medical imaging tasks, is neither supermodular nor submodular. We have demonstrated that our new operator is a tighter approximation to the convex closure of the loss function than slack rescaling, that it generalizes the Lovász hinge, and is convex, piecewise linear, an extension of the loss function, and its subgradient computation is polynomial time.

6.2 Publication list

A complete list of publications is as follows:

Top international conferences

- Jiaqian Yu and Matthew B. Blaschko. Efficient learning for discriminative segmentation with supermodular losses. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2016c
- Jiaqian Yu and Matthew B. Blaschko. A convex surrogate operator for general non-modular loss functions. In Arthur Gretton and Christian Robert, editors, *International Conference on Artificial Intelligence and Statistics*, volume 51 of *Journal of Machine Learning Research: W&CP*, pages 1032–1041, 2016a
- Jiaqian Yu and Matthew B. Blaschko. Learning submodular losses with the Lovász hinge. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1623–1631, 2015b

Under submission

- Jiaqian Yu and Matthew B. Blaschko. An efficient decomposition framework for discriminative segmentation with supermodular losses. 2017. arXiv:1702.03690
- Jiaqian Yu and Matthew B. Blaschko. The Lovász hinge: A convex surrogate for submodular losses. 2015a. arXiv:1512.07797

National conferences and Workshops

- Jiaqian Yu and Matthew B. Blaschko. Efficient learning for discriminative segmentation with supermodular losses. In *Women in Machine Learning Workshop (WiML)*, Barcelona, Spain, 2016d

- Jiaqian Yu and Matthew B. Blaschko. A convex surrogate operator for general non-modular loss functions. In *Benelearn*, Kortrijk, Belgium, 2016b
- Matthew B. Blaschko and Jiaqian Yu. Hardness results for structured learning and inference with multiple correct outputs. In *Constructive Machine Learning Workshop at ICML*, Lille, France, July 2015
- Jiaqian Yu and Matthew B. Blaschko. Lovász hinge for learning submodular losses. In *NIPS Workshop on Representation and Learning Methods for Complex Outputs*, Montreal, Canada, December 2014

Appendix A

Résumé

Cette thèse aborde le problème de l'apprentissage avec des fonctions de perte non-modulaires. Pour les problèmes de prédiction, où plusieurs sorties sont prédites simultanément, les interdépendances dans la perte entre les prédictions peuvent être exprimées comme étant des fonctions de perte non séparable. L'affichage du résultat comme un ensemble commun de prédiction est alors essentiel afin de mieux incorporer les circonstances du monde réel. Dans la minimisation du risque empirique, nous visons à réduire au minimum une somme empirique sur les pertes encourues sur l'échantillon fini avec une certaine perte fonction qui pénalise sur la prévision compte tenu de la réalité du terrain. Dans cette thèse, nous proposons des méthodes analytiques et algorithmiquement efficaces pour traiter les fonctions de perte supermodulaires, submodulaires ainsi que des fonctions de perte non modulaires des fonctions de perte en général. L'exactitude et l'évolutivité sont validées par des résultats empiriques.

D'abord, nous analysons la faisabilité de l'utilisation d'une sortie structurée des machines à vecteurs de support (SVM) avec redimensionnement de marge et des fonctions de perte *supermodulaires*. Nous présentons la dureté de l'intégration de fonctions de perte dans le terme supermodular inférence lorsqu'ils ont des structures graphiques différents. Nous avons ensuite introduit une méthode de décomposition pour la perte d'inférence augmentée. Le principe est basé sur la méthode d'orientation alternée des multiplicateurs, qui ne dépend que de deux solveurs individuels pour la fonction de perte et pour l'infèrence comme deux sous-problémes indèpendants. De cette façon, nous acquérons une méthode algorithmiquement plus efficace permettant d'obtenir plus de flexibilité dans le choix de nos fonctions de perte d'intérêts.

Deuxièmement, nous démontrons la nécessité d'utiliser des fonctions de perte *submodulaires* en problèmes de la prédiction structurés. L'apprentissage avec fonctions submodulaires n'a pas été suffisamment développé. Nous avons prouvé que le redimensionnement de la marge et le redimensionnement de mou dans mènent à des substituts, convexe si et seulement si la fonction de perte augmentation en fonction du nombre de prédictions incorrectes. Cependant, le calcul du gradient ou la méthode des plans sécants pour ces fonctions non-supermodulaire est NP-difficile. En conséquence, nous proposons une nouvelle fonction de substitution pour ces pertes submodulaires, la Lovász hinge, qui conduit à une compléxité en $\mathcal{O}(p \log p)$ avec $\mathcal{O}(p)$ oracle pour la fonction de perte pour calculer un gradient ou méthode de coupe. Nous valider l'exactitude de la Lovász hinge sur diverses tàches. Nous montrons que pour les fonctions de perte submodulaires, la formation avec le Lovász hinge réalise une erreur empirique inférieure à la valeur du redimensionnement de la marge et du redimensionnement de mou. Ce résultat est conforme dû au fait que le convexe est définit correctement comme substitut.

Enfin, sur la base des contributions précédentes, nous introduisons un nouvel opérateur de fonction de substitution convexe pour des fonctions de perte *non-modulaires*, qui fournit pour la première fois une solution facile pour les pertes qui ne sont ni supermodulaires ni submodulaires. Cet opérateur est basé sur une décomposition canonique submodulaire-supermodulaire. De plus, il est prouvé que cet opérateur est linéaire par morceaux, convexe, une extension de la fonction de perte, et pour lesquels le calcul du subgradient est en temps polynomial. Nous prouvons aussi que la perte de Dice, qui est définie en fonction de l'indice de Sørensen-Dice, n'est ni supermodular ni submodular. Les résultats empiriques utilisant la perte de la Sørensen-Dice et un ensemble de fonctions de perte non-modulaires démontrent l'amélioration de la performance, l'efficacité algorithmiquement et l'évolutivité du nouveau substitut convexe.

Bibliography

Stavros Alchatzidis, Aristeidis Sotiras, and Nikos Paragios. Discrete multi atlas segmentation using agreement constraints. In *BMVC*, 2014.

Dragomir Anguelov, Ben Taskar, Vassil Chatalbashev, Daphne Koller, Dinkar Gupta, Geremy Heitz, and Andrew Ng. Discriminative learning of Markov random fields for segmentation of 3D scan data. In *CVPR*, volume 2, pages 169–176, 2005.

Francis Bach. Structured sparsity-inducing norms through submodular functions. In *Advances in Neural Information Processing Systems*, pages 118–126, 2010.

Francis Bach. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2-3):145–373, 2013. ISSN 1935-8237. doi: 10.1561/2200000039.

Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Yoshua Bengio. Learning deep architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009. ISSN 1935-8237. doi: 10.1561/2200000006.

Dimitri P. Bertsekas. *Nonlinear Programming*. Athena, 1999.

Andrew Blake, Carsten Rother, Matthew Brown, Patrick Perez, and Philip Torr. Interactive image segmentation using an adaptive GMMRF model. In *ECCV*, pages 428–441, 2004.

Matthew B. Blaschko. Branch and bound strategies for non-maximal suppression in object detection. In Yuri Boykov, Fredrik Kahl, Victor Lempitsky, and Frank R. Schmidt, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 6819 of *Lecture Notes in Computer Science*, pages 385–398. Springer, 2011.

Matthew B. Blaschko and Christoph H. Lampert. Learning to localize objects with structured output regression. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *European Conference on Computer Vision*, volume 5302 of *Lecture Notes in Computer Science*, pages 2–15. 2008.

Matthew B. Blaschko and Jiaqian Yu. Hardness results for structured learning and inference with multiple correct outputs. In *Constructive Machine Learning Workshop at ICML*, Lille, France, July 2015.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 161–168. 2008.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *T-PAMI*, 26(9):1124–1137, 2004.

Yuri Boykov, Olga Veksler, and Ramin Zabih. Efficient approximate energy minimization via graph cuts. *T-PAMI*, 20(12):1222–1239, 2001.

Deeparnab Chakrabarty, Prateek Jain, and Pravesh Kothari. Provable submodular minimization using Wolfe's algorithm. In *NIPS*, 2014.

Guillaume Charpiat. Exhaustive family of energies minimizable exactly by a graph cut. In *CVPR*, 2011.

Yuxin Chen, Hiroaki Shioi, Cesar Fuentes Montesinos, Lian Pin Koh, Serge Wich, and Andreas Krause. Active detection via adaptive submodularity. In *ICML*, pages 55–63, 2014.

Weiwei Cheng, Eyke Hüllermeier, and Krzysztof J. Dembczynski. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the International Conference on Machine Learning*, pages 279–286, 2010.

Gustave Choquet. Theory of capacities. In *Annales de l'institut Fourier*, volume 5, pages 131–295. Institut Fourier, 1953.

Peter Clifford. Markov random fields in statistics. *Disorder in physical systems: A volume in honour of John M. Hammersley*, pages 19–32, 1990.

Michael Collins and Nigel Duffy. Convolution kernels for natural language. In *Advances in neural information processing systems*, pages 625–632, 2001.

Michael Collins and Nigel Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 263–270. Association for Computational Linguistics, 2002.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, 2(Dec):265–292, 2001.

Antonio Criminisi and Jamie Shotton. *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.

Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26 (3):297–302, July 1945.

Jorge Díez, Oscar Luaces, Juan José del Coz, and Antonio Bahamonde. Optimizing different loss functions in multilabel classifications. *Progress in Artificial Intelligence*, 3(2):107–118, 2015. ISSN 2192-6360. doi: 10.1007/s13748-014-0060-7.

Janardhan Rao Doppa, Jun Yu, Chao Ma, Alan Fern, and Prasad Tadepalli. HC-search for multi-label prediction: An empirical study. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2014.

Jack Edmonds. Matroids and the greedy algorithm. *Mathematical programming*, 1(1):127–136, 1971.

Mark Everingham, Josef Sivic, and Andrew Zisserman. "Hello! My name is... Buffy" – automatic naming of characters in TV video. In *Proceedings of the British Machine Vision Conference*, 2006.

Mark Everingham, Josef Sivic, and Andrew Zisserman. Taking the bite out of automatic naming of characters in TV video. *Image and Vision Computing*, 27(5), 2009.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. ISSN 0920-5691. doi: 10.1007/s11263-009-0275-4. URL http://dx.doi.org/10.1007/s11263-009-0275-4.

Vitaly Feldman, Venkatesan Guruswami, Prasad Raghavendra, and Yi Wu. Agnostic learning of monomials by halfspaces is hard. *SIAM Journal on Computing*, 41(6):1558–1590, 2012.

Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *Proceedings of the 25th International Conference on Machine Learning*, pages 304–311, 2008.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

Satoru Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5(2):186–196, 1980.

Satoru Fujishige. *Submodular functions and optimization*, volume 58. Elsevier, 2005.

B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *ICCV*, 2009.

Wei Gao and Zhi-Hua Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199:22–44, 2013. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/j.artint.2013.03.001.

Jennifer Gillenwater, Rishabh Iyer, Bethany Lusch, Rahul Kidambi, and Jeff Bilmes. Submodular Hamming metrics. In *Neural Information Processing Society (NIPS)*, Montreal, Canada, December 2015.

Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. Springer, 1988. URL http://eudml.org/doc/204222.

Varun Gulshan, Carsten Rother, Antonio Criminisi, Andrew Blake, and Andrew Zisserman. Geodesic star convexity for interactive image segmentation. In *CVPR*, pages 3129–3136, 2010.

Michael Gygli, Helmut Grabner, and Luc Van Gool. Video summarization by learning submodular mixtures of objectives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3090–3098, 2015.

Feng Han and Song-Chun Zhu. Bottom-up/top-down image parsing with attribute grammar. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):59–73, 2009.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: Data mining, inference and prediction*. Springer, 2 edition, 2009.

Satoru Iwata. A faster scaling algorithm for minimizing submodular functions. *SIAM Journal on Computing*, 32(4):833–840, 2003.

Rishabh Iyer and Jeff Bilmes. Algorithms for approximate minimization of the difference between submodular functions, with applications. In *Uncertainty in Artificial Intelligence (UAI)*, 2012.

Rishabh Iyer and Jeff Bilmes. The Lovász-Bregman divergence and connections to rank aggregation, clustering, and web ranking: Extended version. *Uncertainity in Artificial Intelligence*, 2013.

Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

Alexander Kirillov, Dmitrij Schlesinger, Dmitry Vetrov, Carsten Rother, and Bogdan Savchynskyy. M-best-diverse labelings for submodular energies and beyond. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 613–621, Cambridge, MA, USA, 2015. MIT Press.

Daphne Koller, Nir Friedman, Lise Getoor, and Ben Taskar. Graphical models in a nutshell. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.

A. N. Kolmogorov and S. V. Fomin. *Introductory Real Analysis*. Dover, 1975.

Vladimir Kolmogorov. Minimizing a sum of submodular functions. *Discrete Applied Mathematics*, 160(15):2246–2258, 2012.

Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence*, 26(2):147–159, 2004.

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, 2007.

Andreas Krause. SFO: A toolbox for submodular function optimization. *JMLR*, 11:1141–1144, 2010.

Andreas Krause and Daniel Golovin. Submodular function maximization. In Lucas Bordeaux, Youssef Hamadi, and Pushmeet Kohli, editors, *Tractability: Practical Approaches to Hard Problems*. Cambridge University Press, 2014.

Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 53–61, 2013.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 510–520. Association for Computational Linguistics, 2011.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

László Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer, 1983.

O. L. Mangasarian. Uniqueness of solution in linear programming. *Linear Algebra and its Applications*, 25(0):151–162, 1979. ISSN 0024-3795. doi: http://dx.doi.org/10.1016/0024-3795(79)90014-4. URL `http://www.sciencedirect.com/science/article/pii/0024379579900144`.

David McAllester. Generalization bounds and consistency for structured labeling. In *Predicting Structured Data*. MIT Press, 2007.

Ofer Meshi, Nathan Srebro, and Tamir Hazan. Efficient training of structured svms via soft constraints. In *AISTATS*, pages 699–707, 2015.

Mukund Narasimhan and Jeff Bilmes. A submodular-supermodular procedure with applications to discriminative structure learning. In *Uncertainty in Artificial Intelligence (UAI)*, 2005.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions–I. *Mathematical Programming*, 14(1): 265–294, 1978.

Robert Nishihara, Stefanie Jegelka, and Michael I. Jordan. On the convergence rate of decomposable submodular function minimization. In *NIPS*, pages 640–648, 2014.

Sebastian Nowozin. Optimal decisions from probabilistic models: The intersection-over-union case. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

Sebastian Nowozin and Christoph H. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3–4):185–365, 2011.

Oystein Ore and Ystein Ore. *Theory of graphs*, volume 38. American Mathematical Society Providence, 1962.

James B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming*, 118(2):237–251, 2009.

Anton Osokin and Pushmeet Kohli. Perceptually inspired layout-aware losses for image segmentation. In *ECCV*, 2014.

James Petterson and Tibério S. Caetano. Submodular multi-label learning. In *Advances in Neural Information Processing Systems*, pages 1512–1520, 2011.

Patrick Pletscher and Pushmeet Kohli. Learning low-order models for enforcing high-order statistics. In *AISTATS*, 2012.

Simon JD Prince. *Computer vision: models, learning, and inference*. Cambridge University Press, 2012.

Maurice Queyranne. Minimizing symmetric submodular functions. *Mathematical Programming*, 82(1-2):3–12, 1998.

Mani Ranjbar, Greg Mori, and Yang Wang. Optimizing complex loss functions in structured prediction. *Computer Vision–ECCV 2010*, pages 580–593, 2010.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–519, 2014.

Torsten Rohlfing. Image similarity and tissue overlaps as surrogates for image registration accuracy: widely used but unreliable. *IEEE Transactions on Medical Imaging*, 31(2):153–163, 2012.

Carsten Rother, Vladimir Kolmogorov, Victor Lempitsky, and Martin Szummer. Optimizing binary MRFs via extended roof duality. In *CVPR*, 2007.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. ImageNet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. URL `http://arxiv.org/abs/1409.0575`.

Pavel Rychlỳ. A lexicographer-friendly association score. In *Proceedings of Recent Advances in Slavonic Natural Language Processing*, 2008.

Mert R Sabuncu, BT Thomas Yeo, Koen Van Leemput, Bruce Fischl, and Polina Golland. A generative model for image segmentation based on label fusion. *Medical Imaging, IEEE Transactions on*, 29(10):1714–1729, 2010.

Mark Schmidt. UGM: Matlab code for undirected graphical models, 2012.

Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2002.

Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann Le-Cun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations*, 2014.

Josef Sivic, Mark Everingham, and Andrew Zisserman. "Who are you?" – learning person specific classifiers from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

T. Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Biol. Skr.*, 5:1–34, 1948.

Peter Stobbe and Andreas Krause. Efficient minimization of decomposable submodular functions. In *NIPS*, 2010.

Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using graph cuts. In *ECCV*, pages 582–595, 2008.

Daniel Tarlow and Richard S Zemel. Structured output learning with high order loss functions. In *AISTATS*, 2012.

Daniel Tarlow, Inmar E. Givoni, and Richard S. Zemel. HOP-MAP: Efficient message passing with high order potentials. In *AISTATS*, 2010.

Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin markov networks. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 25–32. MIT Press, 2004.

Ambuj Tewari and Peter L. Bartlett. On the consistency of multiclass classification methods. *The Journal of Machine Learning Research*, 8:1007–1025, 2007.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(9):1453–1484, 2005.

Zhuowen Tu, Xiangrong Chen, Alan L Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *International Journal of computer vision*, 63 (2):113–140, 2005.

WT Tutte. *Introduction to the Theory of Matroids*. 1966.

Ranjith Unnikrishnan, Caroline Pantofaru, and Martial Hebert. Toward objective evaluation of image segmentation algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):929–944, 2007.

Takeaki Uno, Tatsuya Asai, Yuzo Uchida, and Hiroki Arimura. An efficient algorithm for enumerating closed patterns in transaction databases. In *Discovery Science*, pages 16–31. Springer, 2004.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning (ICML)*, Lille, France, 2015.

Neil White. *Theory of matroids*. Number 26. Cambridge University Press, 1986.

Jiaqian Yu and Matthew B. Blaschko. Lovász hinge for learning submodular losses. In *NIPS Workshop on Representation and Learning Methods for Complex Outputs*, Montreal, Canada, December 2014.

Jiaqian Yu and Matthew B. Blaschko. The Lovász hinge: A convex surrogate for submodular losses. 2015a. arXiv:1512.07797.

Jiaqian Yu and Matthew B. Blaschko. Learning submodular losses with the Lovász hinge. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1623–1631, 2015b.

Jiaqian Yu and Matthew B. Blaschko. A convex surrogate operator for general non-modular loss functions. In Arthur Gretton and Christian Robert, editors, *International Conference on Artificial Intelligence and Statistics*, volume 51 of *Journal of Machine Learning Research: W&CP*, pages 1032–1041, 2016a.

Jiaqian Yu and Matthew B. Blaschko. A convex surrogate operator for general non-modular loss functions. In *Benelearn*, Kortrijk, Belgium, 2016b.

Jiaqian Yu and Matthew B. Blaschko. Efficient learning for discriminative segmentation with supermodular losses. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2016c.

Jiaqian Yu and Matthew B. Blaschko. Efficient learning for discriminative segmentation with supermodular losses. In *Women in Machine Learning Workshop (WiML)*, Barcelona, Spain, 2016d.

Jiaqian Yu and Matthew B. Blaschko. An efficient decomposition framework for discriminative segmentation with supermodular losses. 2017. arXiv:1702.03690.

Titre : Minimisation du risque empirique avec des fonctions de perte non-modulaires

Mots clefs : Prédiction structurée, Fonction de perte, Submodular, Supermodular, La fonction de perte de substitution.

Résumé : Cette thèse aborde le problème de l'apprentissage avec des fonctions de perte non-modulaires. Pour les problèmes de prédiction, où plusieurs sorties sont prédites simultanément, l'affichage du résultat comme un ensemble commun de prédiction est essentiel afin de mieux incorporer les circonstances du monde réel. Dans la minimisation du risque empirique, nous visons à réduire au minimum une somme empirique sur les pertes encourues sur l'échantillon fini avec une certaine perte fonction qui pénalise sur la prévision compte tenu de la réalité du terrain. Dans cette thèse, nous proposons des méthodes analytiques et algorithmiquement efficaces pour traiter les fonctions de perte non-modulaires. L'exactitude et l'évolutivité sont validées par des résultats empiriques. D'abord, nous avons introduit une méthode pour les fonctions de perte supermodulaires, qui est basé sur la méthode d'orientation alternée des multiplicateurs, qui ne dépend que de deux problémes individuels pour la fonction de perte et pour l'infèrence. Deuxièmement, nous proposons une nouvelle fonction de substitution pour les fonctions de perte submodulaires, la Lovász hinge, qui conduit à une compléxité en $\mathcal{O}(p \log p)$ avec $\mathcal{O}(p)$ oracle pour la fonction de perte pour calculer un gradient ou méthode de coupe. Enfin, nous introduisons un opérateur de fonction de substitution convexe pour des fonctions de perte non-modulaire, qui fournit pour la première fois une solution facile pour les pertes qui ne sont ni supermodular ni submodular. Cet opérateur est basé sur une décomposition canonique submodulaire-supermodulaire.

Title : Empirical risk minimization with non-modular loss functions

Keywords : Structured Prediction, Loss Function, Submodular, Supermodular, Surrogate Loss Function

Abstract : This thesis addresses the problem of learning with non-modular losses. In a prediction problem where multiple outputs are predicted simultaneously, viewing the outcome as a joint set prediction is essential so as to better incorporate real-world circumstances. In empirical risk minimization, we aim at minimizing an empirical sum over losses incurred on the finite sample with some loss function that penalizes on the prediction given the ground truth. In this thesis, we propose tractable and efficient methods for dealing with non-modular loss functions with correctness and scalability validated by empirical results. First, we present the hardness of incorporating supermodular loss functions into the inference term when they have different graphical structures. We then introduce an alternating direction method of multipliers (ADMM) based decomposition method for loss augmented inference, that only depends on two individual solvers for the loss function term and for the inference term as two independent subproblems. Second, we propose a novel surrogate loss function for submodular losses, the Lovász hinge, which leads to $\mathcal{O}(p \log p)$ complexity with $\mathcal{O}(p)$ oracle accesses to the loss function to compute a subgradient or cutting-plane. Finally, we introduce a novel convex surrogate operator for general non-modular loss functions, which provides for the first time a tractable solution for loss functions that are neither supermodular nor submodular. This surrogate is based on a canonical submodular-supermodular decomposition.