Structured prediction for feature selection and performance evaluation

Mao, Qi

2013

Mao, Q. (2013). Structured prediction for feature selection and performance evaluation.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/55288>

<https://doi.org/10.32657/10356/55288>

STRUCTURED PREDICTION FOR FEATURE SELECTION AND PERFORMANCE EVALUATION

A thesis
submitted to the Nanyang Technological University
in fulfilment of the requirement
for the degree of
Doctor of Philosophy of

MAO QI

June 2013

Abstract

Machine learning methods can be employed to discover the relationship between inputs and their desired outputs from a large collection of data points. The outputs of many real-world problems are naturally formed as structured objects in which elements are interdependent in terms of the given structure. Although the conventional methods may be directly applied to solve these problems by treating the elements in each output object independently, they tend to yield suboptimal performance because of the ignorance of interdependency information in the structured outputs. During the last few years, structured prediction provides a natural way to directly model the relationship between inputs and structured outputs. Structure prediction model consists of three components: feature engineering, learning the optimal hypothesis from data by minimizing specific loss, and prediction. In the thesis, the main focus is on the first two components which are interweaved in the following three pieces of my works, where the generalized linear model is used in the prediction.

The first work of this thesis deals with the feature engineering problem which is critical in the structured prediction modeling. The success of structure prediction models is attributed to the fact that their discriminative models are able to account for overlapping features on the whole input observations. These features are usually generated from data by applying a given set of templates on labeled data, but improper templates may lead to degraded performance. To alleviate the difficulty of template selection in feature engineering phrase, a novel multiple template learning paradigm has been proposed to learn a structured prediction model and the importance of each template simultaneously, so that hundreds of arbitrary templates could be added into the learning model without caution of degraded performance. This paradigm has been further extended for structured prediction using generalized p-block norm regularization.

The second work of this thesis focus on the application based loss functions in the special structured prediction problem called automatic image annotation which is one of major tools to enhance the semantic understanding of web images. However, the insufficient performance of image annotation methods prevents these applications from being practical. Although many image annotation methods have been proposed, most of them are inevitably trapped into suboptimal performance because the optimized measure is not the measure for the performance evaluation. To address this issue, a variety of objective-guided performance measures is first summarized under a unified representation. And then, a unified multi-label learning framework has been proposed by directly optimizing a variety of performance measures of multi-label learning tasks.

Instead of template selection for structured prediction, the third work of this thesis studies how to select features for binary classification by optimizing specific multivariate performance measures based on structured prediction model. A generalized sparse regularizer has been proposed. Based on the proposed regularizer, a unified feature selection framework has also been presented for general loss functions. In particular, I have studied the novel feature selection paradigm by optimizing multivariate performance measures based on Structural SVM. To solve the challenging problem of the resultant formulation for high-dimensional data, a two-layer cutting plane algorithm has been proposed, and the convergence has been proved. In addition, the proposed method has been adapted to optimize multivariate measures for multiple instance learning problems.

Acknowledgments

I would like to express my deepest gratitude to my academic supervisor, **Dr. Ivor Wai-Hung TSANG**, for patiently giving guidance and encouragement through my Ph.D research. He guides me into the popular research area in machine learning and the cutting edge research to develop new algorithms for kernel learning and kernel methods. His excellent instructions make me quickly familiar with the new topics. His flexible ways of thinking frequently inspires me to work on the new topics and provide solutions to the difficult problems.

I would also like to thank all my friends and staffs in Center for Computational Intelligence Lab (C2I) for their technical support and enthusiastic help. Many thanks are given to the members of the research group I am working in, particularly Li Wang, Mingkui Tan, Jianbo Yang, Qiaoliang Xiang, Tianyi Zhou, Chunwei Seah and Yiteng Zhai, the colleagues I worked with, particularly Dr. Yew-Soon Ong, Dr. Kian Ming Adam, Dr. Hai Leong Chieu, Dr. Sinno Jialin Pan and Dr. Shenghua Gao, and my roommates, particularly Liang Feng, Xinxing Xu and Lin Chen. Working and living with them inspires my research and makes the tedious Ph.D. life lively and interesting.

Thanks to Prof. Zhi-Hua Zhou from Nanjing University and Prof. Rong Jin from Michigan State University for being my thesis external examiners and Prof. Steven Hoi from Nanyang Technological University for being my thesis internal examiner. They gave me the valuable suggestions and corrections to this work, which greatly helped to improve this thesis in many aspects.

Special thanks to my parents and two elder sisters for their constant support, understanding and love.

Contents

Abstract	i
Acknowledgments	iii
List of Figures	v
List of Tables	vi
List of Notation	vii
1 Introduction	1
1.1 Supervised Learning	1
1.2 Structured Output Prediction Problems	3
1.3 Structured Output Prediction Models	6
1.4 Contributions	8
1.5 Thesis Outline	10
2 Literature Review	12
2.1 Supervised Models	12
2.1.1 Classification	13
2.1.2 Kernelization	16
2.1.3 Multiple Kernel Learning	20
2.1.4 Feature Selection	22
2.2 Structured Output Prediction	24
2.2.1 Probabilistic Models	25
2.2.2 Margin-based Methods	30
2.2.3 Feature Engineering Issue	32
2.2.4 Performance Evaluation Issue	35

3	Structured-Prediction-Based Feature Selection	39
3.1	Introduction	39
3.2	Multiple Template Learning	41
3.2.1	Feature v.s. Template	41
3.2.2	Model Formulation	43
3.2.3	Multiple Kernel Learning by Cutting Plane Method	44
3.2.4	Convergence & Computational Complexity	49
3.3	Extension to p -Block-Norm Regularization	53
3.4	Experiments	55
3.4.1	Experimental Setting	55
3.4.2	Chinese Word Segmentation	57
3.4.3	Named Entity Recognition	65
3.4.4	Dependency Parsing	67
3.4.5	MTL ^{struct} with p -Block-Norm Regularization	69
3.5	Conclusion	70
4	Structured-Prediction-Based Performance Measure	72
4.1	Introduction	72
4.2	Objective-guided Performance Measures	74
4.2.1	Example-based Measures	75
4.2.2	Macro-averaging Measures	76
4.2.3	Micro-averaging Measures	77
4.2.4	Hamming Measure	77
4.3	Analysis under Image Annotation Setting	78
4.3.1	An Illustrating Example	79
4.4	Optimizing Objective-guided Performance Measures	80
4.4.1	Objective-Guided Risks	80
4.4.2	Proposed Formulation	81
4.4.3	Joint Feature Map for Bipartition Measures	82
4.4.4	Generalized Learning Algorithm	83
4.4.5	The Most Violated Constraint for Bipartition Measures	85

4.4.6	Subproblem over the Set of Violated Constraints	86
4.4.7	Complexity Analysis	86
4.5	Experiments	87
4.5.1	Methods for Comparisons	87
4.5.2	Experiments on Multi-label Datasets	88
4.5.3	Experiments on Image Annotation Datasets	91
4.6	Conclusion	97
5	Structured-Prediction-Based Feature Selection for Performance Measures	99
5.1	Introduction	99
5.2	SVM for Multivariate Performance Measure	101
5.3	Generalized Sparse Regularizer	102
5.4	Feature Selection for Multivariate Performance Measures	104
5.5	Two-Layer Cutting Plane Algorithm	105
5.5.1	Group Feature Generation	105
5.5.2	Group Feature Selection	106
5.5.3	The Proposed Algorithm	112
5.5.4	Finding the Most Violated $\bar{y}'$ and $\mathbf{d}$	115
5.6	Relations To Existing Methods	117
5.6.1	Connections to SKM and l_1 SVM	117
5.6.2	Connection to NMMKL	117
5.6.3	Connection to FGM	118
5.6.4	Connection to l_0 SVM	119
5.7	Multiple Instance Learning for Multivariate Performance Measures	120
5.8	Experiments	122
5.8.1	Parameter Sensitivity Analysis	123
5.8.2	Time Complexity Analysis	125
5.8.3	Feature Selection for Accuracy	126
5.8.4	Feature Selection for Multiple Instance Learning	128
5.8.5	Multivariate Performance Measures for Document Retrieval	131
5.9	Conclusion	134

6	Conclusions and Future Work	135
6.1	Summary of Contributions	135
6.2	Future Work	138
6.2.1	Structured Output Regression	138
6.2.2	Semi-supervised Multiple Template Learning	139
6.2.3	Extensions for Other Multivariate Performance Measures	139
6.2.4	Exploring the Generalized Sparse Regularizer	141
	Publication	142
	References	144

List of Figures

1.1	Supervised learning setting	2
1.2	Illustration of an input-output pair for part of speech tagging task. . . .	3
1.3	Illustration of an input-output pair for dependency parsing task.	4
1.4	Illustration of an input-output pair for multi-label classification task. . .	5
1.5	Structured learning setting	6
1.6	The diagram illustrates the relationships of the proposed methods with three main components of the general structured prediction models. The corresponding applications are also listed for empirical evaluations of the proposed methods in this thesis.	9
2.1	Three models with different linear chain structures.	27
2.2	Example images from UIUC-Sport dataset. For each image, the ground truth keywords are shown at the bottom. Image annotation aims at predicting these keywords for the given images.	36
3.1	The average CPU time on PKU dataset by varying C on five randomly sample of training and testing datasets.	61
3.2	The performance and CPU time of OnlineMKL on PKU dataset with benchmark split and $C = 1$ by varying the number of epochs.	61
3.3	The learned template weights for Chinese word segmentation in TP1 and TP2 on PKU.	62
3.4	F1 measure by varying C in TP2 on the Chinese word segmentation datasets.	63
3.5	The CPU time on Spanish and Dutch dataset by varying C	65

3.6	The relative learned template weights for Named entity recognition on Spanish and Dutch. The horizontal line with 1 stands for methods with the average weights.	66
3.7	The relative learned template weights for Dependency Parsing on Danish and Swedish. The horizontal line with 10^0 stands for methods with the average weights.	68
3.8	The normalized weights of $p\text{MTL}^{hmm}$ on the large set of templates by varying p values on PKU and ESP datasets, respectively.	70
4.1	(a) The frequency of each keyword on UIUC-Sport dataset. (b) The effect on the bipartition measures with varying numbers of $\{a\}$ -Rare keywords (the number of columns).	78
4.2	The hierarchical structure of multi-layer hypotheses where the rectangle stands for the keyword matrix $\mathbf{Y}$. The blank area in the rectangle means no independence assumption.	81
4.3	The training time of the proposed methods for optimizing the objective-guided performance measures on Medical data.	91
4.4	Histograms (a)-(d) show the frequency of images with the specific number of keywords on four datasets.	92
4.5	The example-based F_1 score from IAP_k method by varying k on UIUC-Sport.	96
4.6	Example annotations on UIUC-sport dataset by different methods: ground truth, IAP_1 , κNN , and Tagprop from top to bottom.	97
4.7	Example annotations on UIUC-sport dataset by different methods on the top 5 keywords: ground truth, IAP_5 , κNN , and Tagprop from top to bottom.	98
5.1	First row (a-b): Testing Accuracy and F_1 scores as well as the number of selected features of the proposed method FS_{multi}^Δ on <i>News20.binary</i> dataset by varying C and the number of selected features. Second row (c-d): The corresponding relationship among parameters B , C , and the number of selected features.	124
5.2	Training time on different datasets	125

5.3	Testing accuracy on different datasets	126
5.4	The sparsity of features of $FS_{multi}^{hamming}$ with varying B on different datasets. Each row bar with different color represents the different subset of features selected under current B , where the white region means the features are not selected.	127
5.5	Testing F_1 scores on <i>Image</i> dataset.	129
5.6	Testing Accuracy and F_1 scores on <i>Image</i> dataset by comparing the pro- posed methods with other multiple instance learning methods.	131
5.7	The average performance improvement of FS_{multi}^{Δ} with varying B on dif- ferent datasets.	133
5.8	Testing F_1 in terms of the average number of selected features on Sector and News20.	133
6.1	The diagram illustrates the relationships of the proposed methods with three main components of the general structured prediction models, and the summary of contributions for each proposed method is illustrated at the bottom.	136
6.2	The diagram illustrates the relationships of the proposed methods with three main components of the general structured prediction models, and the future work based on the proposed method is illustrated at the bottom.	138

List of Tables

3.1	Benchmark datasets for Chinese word segmentation.	57
3.2	Two sets of templates used for Chinese word segmentation task: TP1 and TP2. The symbol '-' means that this template is not used in the setting.	58
3.3	The performance (TP1/TP2) of Chinese word segmentation over five randomly sampled training and testing datasets. For each method, the first row is the mean and the second row is the corresponding standard deviation. The best results across the compared methods are shown in bold on TP1 and TP2, individually.	59
3.4	The Chinese word segmentation results (TP1/TP2) of comparing methods on all the testing datasets. The best results across the compared methods are shown in bold on TP1 and TP2, individually.	60
3.5	The overall performance of the comparing methods on Spanish and Dutch for named entity recognition. The best results across different methods are shown in bold.	64
3.6	The overall performance of the comparing methods on Spanish and Dutch for named entity recognition with the selected subset of templates. The best results across different methods are shown in bold.	67
3.7	The results (MSTParser / MTL^{parse}) of the dependency parsing on Danish and Swedish. The best results across the compared methods are shown in bold.	68
3.8	The overall results of $pMTL^{hmm}$ on PKU and ESP by varying p . The best results across the different p are shown in bold.	69

4.1	Contingency table based measures with the entries of contingency table as the count of $t = 1 \& p = 1$, $t = -1 \& p = 1$, $t = 1 \& p = -1$, $t = -1 \& p = -1$ to be a, b, c, d , respectively.	74
4.2	The summary of hypotheses and their corresponding score functions, loss functions and the set of i.i.d. samples.	80
4.3	A variety of performance measures (mean $\pm$ standard deviation, %) by comparing different methods on Emotions by five-fold cross validation except that hamming loss is one minus hamming measure without percentage. The best results are in bold.	90
4.4	A variety of performance measures (mean $\pm$ standard deviation, %) by comparing different methods on Medical by five-fold cross validation except that hamming loss is one minus hamming measure without percentage. The best results are in bold.	90
4.5	The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on UIUC-Sport by five-fold cross validation. The best results are in bold.	93
4.6	The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on LabelMe by five-fold cross validation. The best results are in bold.	94
4.7	The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on NUS-WIDE-Object by five-fold cross validation. The best results are in bold.	95
4.8	The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on Corel5k by five-fold cross validation. The best results are in bold.	96
5.1	Datasets used in the experiments	122
5.2	The macro-average testing performance comparisons among different methods. The quantities in the parentheses represent won/lost of the current method comparing with FS_{multi}^{Δ} . The last column indicates the average number of features is actually used in the current method for a specific measure. The symbol '*' indicates the level of significance at 0.95 according to t-test applied to pairs of results over classes	132

6.1	Various performance measures are widely used in the literature for evaluating learning methods in different learning settings. The contingency-table based measures belong to a large family which has been mentioned in Chapter 4 and Chapter 5. The rest of performance measures are listed but not the least. The column “Concerned” stands for the measures used in this thesis.	140
-----	--	-----

Preliminary

In the sequel, the comma “,” inside the pair of “[” and “]” is used to concatenate elements to form a row vector, while the semicolon “;” is used to form a column vector. The transpose of a matrix or a vector is denoted by T and $\langle \mathbf{u}, \mathbf{v} \rangle$ is the inner product of two vectors $\mathbf{u}$ and $\mathbf{v}$. $\mathbf{A} \succeq \mathbf{0}$ means that the matrix $\mathbf{A}$ is symmetric and positive semidefinite (psd). Denote l_p norm of a vector $\mathbf{v}$ by $\|\mathbf{v}\|_p$. Binary operator $\odot$ represents the elementwise product between two vectors/matrices.

Chapter 1

Introduction

Many tasks in the real world cannot be defined well by human beings. In other words, one might not be able to specify a concise relationship between inputs and their desired outputs. However, a collection of input-output pairs is usually achievable. One would like machines to approximate the relationship implicitly from the collected examples so as to produce correct outputs for a large number of sample inputs. Moreover, the important relationships and correlations are mostly hidden in a large collection of data points. These hidden structures are useful to interpret the data or predict the future unseen data.

In this thesis, I have studied the structured prediction model from two aspects, feature functions and user-specified losses, in several different disciplines, including natural language processing, computer vision, and computational biology. I have also derived the formulations and shown the results over a wide range of applications including Chinese word segmentation, named entity recognition, dependency parsing, automatic image annotation, and image/document retrieval.

1.1 Supervised Learning

A hypothesis h is selected from a class of functional space $\mathcal{H}$ according to a priori. Traditionally, h takes an observation $\mathbf{x}$ as the input and a response y as the output, i.e., $y = h(\mathbf{x})$. The value of y can be a simple scalar, in the sense that they are characterized by an arbitrary identifier. For example, y is a category from a given set of predefined categories. If the size of the category set is two, the learning task is called binary

classification, otherwise multi-class classification; y can also be a real value, the models of which are called regression. Problems considered by traditional methods assume that individual outputs are independent and disconnected.

Figure 1.1: Supervised learning setting

In the supervised setting, we are given a set of labeled data $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ with n number of input-output pairs $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ are independent and identically distributed (i.i.d.) from an unknown joint probability distribution. Given a hypothesis space $\mathcal{H}$, the goal of supervised learning is to obtain the best hypothesis $h \in \mathcal{H}$ which can generalize well on the unseen data. In the prediction procedure, given a test input, the predicted output is given by the learned model. In general, the given set of data is not preprocessed. In other words, the data is collected according to the specific task of interest. Taking document classification for example, a collection of documents which consist of sequences of characters or words is provided. We have to transform each document into its feature representation, such as tf-idf, bag of words, and bi-gram. These documents are the raw input denoted by O , while the representation $\mathbf{x}$ of the raw input O is called features. This preprocessing step is called feature extraction. Figure 1.1 shows the diagram of supervised learning setting.

Supervised classification problem has plenty of theories and practices based on the given hypothesis class $\mathcal{H}$ and the criterion for selecting a hypothesis $h \in \mathcal{H}$ given training data such that the selected hypothesis can work well on unseen data [1–3]. In this thesis, the linear model is taken as the hypothesis set $\mathcal{H}$ since the linear models can approximate arbitrary non-linear decision boundaries in the reproducing kernel Hilbert space (RKHS) [4].

1.2 Structured Output Prediction Problems

For many real world applications, outputs could often be complex objects with certain internal structures. However, traditional methods generally deal with this complex output by ignoring their internal structures and treating elements of each output independently. There are at least two merits to treat the output as a whole. First, the internal structure underlying the output explored by learning methods is useful for good prediction performance. Second, the non-decomposable performance measures can be evaluated and optimized properly since they are defined on the whole output and are not decomposable in terms of elements of the output. The following tasks are used as examples to illustrate the above two merits which are captured in structured output prediction models. These tasks are also used for illustrating the performance of the proposed methods in this thesis.

Figure 1.2: Illustration of an input-output pair for part of speech tagging task.

Sequence Labeling. Sequence labeling is to assign sequences of labels to sequences of associated objects. It is very common in NLP and bioinformatics. Figure 1.2 illustrates one input-output pair for part of speech (POS) tagging task. The input $\mathbf{x}$ is a token sequence “ *U.K. base rates are at their highest level in eight years .* ” (each token is an English word), and the output $\mathbf{y}$ is also a token sequence (each token is a tag from the POS tag set). For example, in Penn Treebank 3, there are

36 POS tags, e.g., “JJ” is adjective, “NN” is noun, singular or mass, and “VBP” is verb. The underlying structure of sequence labeling is that the current tag depends only on its neighbor tags, and is independent of the rest of tags in the sequence. In the case of POS tagging, the POS tag of the current position is related to the POS tags of its neighbors. For example, if the current word “U.K.” is labeled as “JJ”, the following word “base” is labeled as “NN” with higher probability than that labeled as “VBP”. The POS tag of “U.K.” might not affect the POS tag of “years” because the distance between two words are too long. As the English sentence is naturally formed as a sequence of words, the sequence structure to model POS task should be much effective. Learning methods that exploit this information will be greatly helpful to improve the prediction performance, while conventional methods ignore this useful information by labeling each word separately. This assumption has been captured by structured output prediction methods in many applications, such as Chinese word segmentation [5], part of speech tagging [6], chunking [7], named entity recognition [8], and information extraction [9].

Figure 1.3: Illustration of an input-output pair for dependency parsing task.

Dependency Parsing. Dependency parsing is to find a generic dependency tree y for a given generic input sentence $\mathbf{x}$. Figure 1.3 shows one example of the input-output pair where the input is a sentence “*John hit the ball with the bat*”, and the output is a directed tree structure with an augmenting node called “root”. It is clear to see that the elements in the output of a given input are formed in a directed tree structure. Traditional binary classification methods can be adapted to predict each directed edge independently, but this strategy is based on the strong assumption that all the directed edges are independent and identically distributed. This is not the case of dependency parsing task where directed edges are usually strongly

dependent on each other. Learning methods need to determine the hypothesis h which takes the whole sentence as the input and returns corresponding directed tree structure to exploit the dependency structure with the underlying trees. Structured output prediction models have been successfully applied in this problem [10].

Figure 1.4: Illustration of an input-output pair for multi-label classification task.

Multi-label Classification. Multi-label classification is to assign a subset of a predefined set or hierarchy of labels to each instance. This is an extension of the common single-label problem where only a single label from the label set is assigned to each instance, such as multi-class problems. It is quite natural to many domains, such as text classification, scene and video classification, and medical diagnosis [11]. Figure 1.4 shows one example for image scene classification. For the given image, three labels are assigned, i.e., “*desert*”, “*sunset*”, and “*trees*”. Conventional methods train the binary classifier for each label and then predict the associated label independently. Finally, all the predicted labels are grouped together to form the predicted output. In the given example, five binary classifiers are learned from the training data to distinguish whether the corresponding label exists or not for an unseen image. These methods imply the loss is decomposable in terms of labels. However, the frequently used performance measures in multi-label classification are non-decomposable, such as precision, recall, and F measure [11, 12]. These measures require the statistics of all the training data as a whole for computation. By grouping all inputs and outputs in the training data to form an input object and corresponding output object respectively, structured output prediction problem is formed over the pair of the input object and output object. More importantly, the non-decomposable performance measures can be readily formulated and further optimized on the training data in the structured output prediction framework [13].

1.3 Structured Output Prediction Models

Although the conventional classification methods can be adapted to solve above problems by breaking the structured output into a collection of yes/no prediction, they fail to exploit the interdependencies among elements in a predicted output. The natural way is to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ mapping inputs $\mathbf{x} \in \mathcal{X}$ to complex and structured outputs $\mathbf{y} \in \mathcal{Y}$. The name of such tasks are called structured output prediction [14]. Figure 1.5 shows the diagram of the structured output learning setting.

Figure 1.5: Structured learning setting

By comparing with conventional classification framework shown in Figure 1.1, there are three special challenges arising from structured output prediction problem:

Feature function. In conventional supervised classification, the output space of binary classification is $\mathcal{Y} = \{+1, -1\}$, while that of multi-class classification is $\mathcal{Y} = \{1, 2, \dots, c\}$. It is reasonable to assume that labels are independent. However, in structured prediction problems, label dependency is an important factor that needs to be captured. Joint feature map $\phi : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^d$ in d dimensional real space can readily capture the label dependency as well as the dependency between

input and output. In general, the raw input O is provided for specific applications, taking the English sentence “*U.K. base rates are at their highest level in eight years* .” in part of speech tagging task for example. The feature representation $\mathbf{x}_i$ of the raw input O_i is required for learning. Hence, feature engineering is a critically important step to transform the pair of input and output into joint feature representation. This transformation is more related to domain prior knowledge, that is often provided by domain experts.

Decision function. The linear model is used in this thesis. Hence, the decision function for structured output prediction problems can be defined as

$$y = \arg \max_{y \in \mathcal{Y}} \mathbf{w}^T \phi(\mathbf{x}, y). \quad (1.1)$$

The conventional supervised classification methods based on the linear model are the special cases of structured output prediction. If $\phi(\mathbf{x}, y) = y\mathbf{x}$ where $y \in \{+1, -1\}$, then $y = \arg \max_{y \in \{+1, -1\}} y\mathbf{w}^T \mathbf{x}$ is equivalent to $y = \text{sgn}(\mathbf{w}^T \mathbf{x})$. Since labels are dependent, the feasible space $\mathcal{Y}$ can be very large. This makes the problem (1.1) hard to solve. This problem is usually named as inference.

Loss function. Traditional supervised learning has a plenty of theoretical supports for good generalization by minimizing the empirical loss of training data. Hence, loss functions are considered as an important criteria for learning a good hypothesis. 0/1 loss is frequently considered as the goal to be minimized in traditional supervised learning. However, it will be beneficial to take into account different penalties for different errors [6]. For example, in the task of dependency parsing, there are a huge amount of directed trees which are not the same as the ground truth one for a given sentence. It is reasonable to think that one tree with two edges labeled incorrectly may be worse than the one with only one edge labeled incorrectly. Even though they are both labeled incorrectly, their risks should be penalized differently. The loss of a dependency tree is defined as the number of words that have the incorrect parent in [10]. In the information retrieval applications, one wants to find a set of relevant instances given a label. Averaged over the set of possible labels are the most widely accepted performance measures, such as macro-F1 [15, 16]. Many applications with

different loss functions for structured output prediction are discussed in [6]. More performance measures [11] are employed for evaluating the performance of multi-label classification problems. Loss functions used in structured output prediction will be discussed in details in Chapter 2.

Given a model $\mathbf{w}$, the loss over the input-output pair $(\mathbf{x}_i, y_i)$ is denoted as

$$\Delta \left(y_i, \arg \max_{y \in \mathcal{Y}} \mathbf{w}^T \phi(\mathbf{x}_i, y) \right) \quad (1.2)$$

where $\Delta(y, y')$ is a user specified loss function over the ground truth output y and the predicted output y' . For two different predicted outputs, the losses may be quite different. It is worth noting that the loss as the criterion to be minimized is equivalent to the performance measure to be maximized [13].

The standard methods of structured prediction problems focus mainly on the preprocessed features generated by a set of templates [6, 9, 17], but take less attention on the discrepancy of these templates [18]. And, many applications have their intrinsic evaluation criterion (performance measure). Most methods leads to sub-optimal solutions because they are not proposed to optimize their intrinsic performance measures.

1.4 Contributions

This thesis builds on structured prediction models and proposes new methods to learn feature functions for feature engineering and optimize user-specific performance measures. I have proposed an efficient template learning methods that can be applied to many applications in natural language processing (Chapter 3). For image annotation, I have given analysis for a variety of user-guided measures and have also proposed a general framework to optimize them (Chapter 4). To easily optimize the non-decomposable user-specified measure, I have also proposed to learn classifier for collective classification and choose effective features simultaneously (Chapter 5). Figure 1.6 shows the relations of the proposed methods in this thesis with three main components of the general structured prediction models as well as the corresponding applications for empirical evaluations. The detailed contributions of the proposed methods in this thesis are listed as follows:

Figure 1.6: The diagram illustrates the relationships of the proposed methods with three main components of the general structured prediction models. The corresponding applications are also listed for empirical evaluations of the proposed methods in this thesis.

- **Efficient Multiple Template Learning for Structured Prediction [19]**

A Multiple Template Learning (MTL^{struct}) paradigm has been proposed to learn the weight of each template and the structured prediction model in the batch mode, simultaneously. Following the group sparsity, to capture the dependency among a group of features instantiated from one template, learning the weighting of these groups is formulated as a Multiple Kernel Learning (MKL) problem. This specific MKL problem usually involves exponential number of constraints due to the interdependencies among output variables. An efficient cutting plane algorithm has been proposed to solve this MKL problem. Moreover, its convergence has been proved. Two well-known structured prediction tasks, i.e. sequence labeling and dependency parsing, are showcased. I have further extended the proposed learning paradigm to solve structured output problems with p -block norm regularization.

- **A Unified Objective-guided Image Annotation [20]**

A unified representation of various performance measures based on contingency table is proposed for image annotation. The analysis reveals that macro-averaging

measures are very sensitive to infrequent keywords, which is rarely studied in literatures. Moreover, hamming measure is easily affected by skewed distributions which lead to be a bit over-estimated. To optimize the objective-guided performance measure, a hierarchical structure of multi-layer hypothesis has been proposed to unify all the contingency table based measures. The unified loss functions can be defined correspondingly. With the relaxed surrogate function, the learning model can be optimized efficiently.

- **A Feature Selection Method for Multivariate Performance Measures [21, 22]**

A generalized sparse regularizer for feature selection has been proposed. After that, a unified feature selection framework has been presented for general loss functions based on the proposed regularizer. Particularly, optimizing multivariate performance measures has been studied in this framework. To our knowledge, this is the first work to optimize multivariate performance measures for feature selection. Due to exponential number of constraints brought by non-smooth multivariate loss functions and exponential number of feature subset combinations, the resultant optimization problem is very challenging for high-dimensional data. To tackle this challenge, a two-layer cutting plane algorithm has been proposed to solve this problem effectively and efficiently. Specifically, Multiple Kernel Learning (MKL) by cutting plane algorithm is proposed to deal with exponential size of constraints induced by multivariate losses. Connections to a variety of the state-of-the-art feature selection methods are discussed in details. I have also extended it for Multiple Instance Learning (MIL). This is seldom studied in MIL scenarios, but it is important for the real world applications of MIL tasks.

1.5 Thesis Outline

Below is a summary of the rest of chapters in this thesis:

Chapter 2. Literature Review: Some related concepts about the definition and statistical learning framework for classification, including hypothesis space, loss functions, and empirical risk are first reviewed. Linear models including logistic regression and support vector machines are discussed. And then their corresponding

structured prediction models in both probabilistic and margin-based points of view are discussed. The kernel is introduced for obtaining non-linear decision boundary. To obtain the good kernel, kernel learning especially multiple kernel learning is reviewed. Its application in feature selection and its related methods are also briefly introduced.

Chapter 3. Structured-Prediction-Based Feature Selection: The relations between features and template are first discussed in the field of natural language processing. And then the model formulation is proposed to automatically weigh templates from a given set of templates and learn the structured prediction model over these weighed templates. To solve the formulated problem, an efficient cutting plane method is proposed. Further, the proposed model is extended for p-Block norm regularization. Experiments on three NLP tasks are conducted.

Chapter 4. Structured-Prediction-Based Performance Measure: First, various performance measures used for evaluating the performance of automatic image annotation are summarized. Then, a unified framework for optimizing objective-guided performance measures is proposed. Experiments on four image annotation datasets are conducted.

Chapter 5. Structured-Prediction-Based Feature Selection for Performance Measures:

This chapter first reviews the SVM method for multivariate performance measure. To enforce sparsity over the model parameter, a generalized sparse regularizer is proposed. Based on this regularizer, the proposed feature selection method for multivariate performance measures is given. To solve the proposed formulation, a two-layer cutting plane algorithm is proposed. Moreover, the connections to various existing methods are discussed. The proposed methods are further extended for multiple instance learning. Experiments on large scale datasets are conducted to compare with baseline methods.

Chapter 6. Conclusion and Future Directions A short summary about the main contributions of the thesis is given. Extensions and future research directions which are not addressed in this thesis are also discussed.

Chapter 2

Literature Review

2.1 Supervised Models

The goal of a learning method is to find a hypothesis $h : \mathcal{X} \mapsto \mathcal{Y}$ from training data such that it can generalize well on the unseen data. The hypothesis h is from the hypothesis space $\mathcal{H}$ that maps an input $\mathbf{x} \in \mathcal{X}$ to an output $y \in \mathcal{Y}$. The input space $\mathcal{X}$ is an arbitrary set, usually in d -dimensional real space $\mathcal{X} = \mathbb{R}^d$, while the output space $\mathcal{Y}$ is either discrete or continuous. A learning problem with a discrete output space $\mathcal{Y} = \{\Upsilon_1, \dots, \Upsilon_k\}$ is called *classification*. k is the number of classes. If the output space is in the real space $\mathcal{Y} = \mathbb{R}$, the learning problem is called *regression*. If the training data only consists of input-output pairs $(\mathbf{x}, \mathbf{y})$, it is called *supervised learning*. In this thesis, supervised classification problems are the main concerns where the training data $\mathcal{D}_{tr} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ is a set of independent and identically distributed samples drawn from a fixed but unknown joint distribution $p(\mathbf{x}, \mathbf{y})$. N is the number of input-output pairs in the training data.

For a specific learning algorithm, the following two problems need to be answered: 1) what is the hypothesis space $\mathcal{H}$? 2) what is the criterion for selection of h from $\mathcal{H}$? In this survey, the linear hypotheses $\mathcal{H}$ is considered. The criterion for selecting h is to minimize the *risk function* $\mathcal{R}_p^\ell(h)$ which measures the expected error of the approximation $h(\mathbf{x})$ to y :

$$\mathcal{R}_p^\ell(h) = \mathbf{E}_{(\mathbf{x}, y) \sim p}[\ell(\mathbf{x}, y, h(\mathbf{x}))] \quad (2.1)$$

where the *loss function* $\ell : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}^+$ measures the penalty for predicting $h(\mathbf{x})$ on the sample $(\mathbf{x}, y)$. Generally, since the distribution p is unknown, we have to estimate

the risk of h using its *empirical risk* $\mathcal{R}_{\mathcal{D}_{tr}}^\ell(h)$ on the training data $\mathcal{D}_{tr}$:

$$\mathcal{R}_{\mathcal{D}_{tr}}^\ell(h) = \frac{1}{N} \sum_{i=1}^N \ell(\mathbf{x}_i, y_i, h(\mathbf{x}_i)). \quad (2.2)$$

It may not be a good idea to select a hypothesis by minimizing the empirical risk

$$h^* = \arg \min_{h \in \mathcal{H}} \mathcal{R}_{\mathcal{D}_{tr}}^\ell(h) \quad (2.3)$$

since the solution h^* of (2.3) may *overfit* the training data with zero or very small empirical risk if $\mathcal{H}$ is large enough [1]. The tradeoff between complexity of $\mathcal{H}$ and the error on the training data measured by $\mathcal{R}_{\mathcal{D}_{tr}}^\ell(h)$ is achieved by minimizing the weighted combination of the two terms

$$h^* = \arg \min_{h \in \mathcal{H}} \Omega(h) + C \mathcal{R}_{\mathcal{D}_{tr}}^\ell(h) \quad (2.4)$$

where $\Omega(h)$ measures the complexity of h called regularization, and $C > 0$ is a tradeoff parameter. According to learning criterion (2.4), the learning algorithms are completely characterized by the hypotheses $\mathcal{H}$, the loss function ℓ , and the regularization $\Omega(h)$. In general, the loss function satisfies $\ell(\mathbf{x}, y, h(\mathbf{x})) = 0$ if $y = h(\mathbf{x})$, and $\ell(\mathbf{x}, y, h(\mathbf{x})) > 0$ otherwise¹. The regularization of h is commonly set as $\Omega(h) = \|h\|_2^2$. A proper choice of C is usually tuned by measuring performance on a hold-out data.

In what follows, I will focus on the specific linear hypothesis space and discuss a variety of learning algorithms in terms of their own special loss functions. And then their dual forms are derived so that the kernelized counterparts can be readily obtained. Kernel learning and feature selection which are most related to my work in this thesis are also discussed.

2.1.1 Classification

The conventional models for multi-class classification are discussed in the following. Since binary classification problem is the special case of multi-class classification, it will not

¹For example, the hinge loss is defined as $\ell(\mathbf{x}, y, f(\mathbf{x})) = \max(0, 1 - yf(\mathbf{x}))$ where $f(\mathbf{x})$ for binary classification is the confidence value of the input instance $\mathbf{x}$. And the hypothesis $h(\mathbf{x}) = \arg \max_{y \in \{-1, +1\}} yf(\mathbf{x}) = \text{sgn}(f(\mathbf{x}))$. In this case, $\ell(\mathbf{x}, y, h(\mathbf{x})) = 0$ if $y = h(\mathbf{x})$, otherwise 2. The hinge loss on hypothesis h still holds, but the relaxed version using confidence value does not hold any more.

be discussed specifically. The output space is a discrete set $\mathcal{Y} = \{\Upsilon_1, \dots, \Upsilon_k\}$ with k classes. The linear model of interest is denoted by

$$h(\mathbf{x}; W) = \arg \max_{y \in \mathcal{Y}} \langle \mathbf{w}_y, \mathbf{x} \rangle, \quad (2.5)$$

where $W = [\mathbf{w}_{\Upsilon_1}, \dots, \mathbf{w}_{\Upsilon_k}]$ is a $d \times k$ matrix with the r th column $\mathbf{w}_{y_r}$, and $\langle \mathbf{a}, \mathbf{b} \rangle$ is the inner product of vector $\mathbf{a}$ and vector $\mathbf{b}$. Given the training data $\mathcal{D}_{tr}$, the common loss function for classification is 0/1 loss

$$\ell^{0/1}(\mathbf{x}, y, h_W(\mathbf{x})) = \pi(y \neq h_W(\mathbf{x})) \quad (2.6)$$

where $\pi(\cdot)$ is the indicator function with $\pi(true) = 1$ and $\pi(false) = 0$. And then, the optimal hypothesis h_W is selected from $\mathcal{H}$ by solving (2.4) with respect to variable W . The discontinuity of 0/1 loss makes optimization problem (2.4) intractable. Several upper bound of 0/1 loss are designed, for example, hinge loss [23]. In this section, two closely related models are discussed: logistic regression and support vector machine. The former allows for an interpretation of the outputs of the classifier in probabilistic viewpoint; the latter leads to the notion of margins in a geometrical perspective.

Probabilistic Model: Logistic Regression [24]. The conditional probability distribution $p(y|\mathbf{x}; W)$ is defined based on the linear confidence measure $\langle \mathbf{w}_y, \mathbf{x} \rangle$ as

$$p(y|\mathbf{x}; W) = \frac{\exp(\langle \mathbf{w}_y, \mathbf{x} \rangle)}{Z(\mathbf{x}; W)} \quad (2.7)$$

where the partition function $Z(\mathbf{x}; W) = \sum_{y \in \mathcal{Y}} \exp(\langle \mathbf{w}_y, \mathbf{x} \rangle)$ is the normalization term for the necessary condition of probability $\sum_{y \in \mathcal{Y}} p(y|\mathbf{x}; W) = 1, p(y|\mathbf{x}; W) \geq 0, \forall y \in \mathcal{Y}$. This model is commonly referred to *logistic regression*. The decision function for multi-class classification is to find the y with the highest probability as the predicted class, that is

$$h_W(\mathbf{x}) = \arg \max_{y \in \mathcal{Y}} p(y|\mathbf{x}; W) = \arg \max_{y \in \mathcal{Y}} \log p(y|\mathbf{x}; W) \quad (2.8)$$

which is the same as (2.5). A standard estimation method for a probabilistic model is to maximize the conditional log-likelihood averaged over the train data

$$\mathcal{L}_{\mathcal{D}_{tr}}(W) = \sum_{i=1}^N \log p(y_i|\mathbf{x}_i; W) = \sum_{i=1}^N [\langle \mathbf{w}_{y_i}, \mathbf{x}_i \rangle - \log Z(\mathbf{x}_i; W)]. \quad (2.9)$$

We accordingly define the *log-loss* function as the negative conditional log-likelihood

$$\ell^{\log}(\mathbf{x}, y, h_W(\mathbf{x})) = \log Z(\mathbf{x}; W) - \langle \mathbf{w}_y, \mathbf{x} \rangle \quad (2.10)$$

which is the building block for empirical risk $\mathcal{R}_{\mathcal{D}_{tr}}^{\ell^{\log}} = \frac{1}{N} \sum_{i=1}^N \ell^{\log}(\mathbf{x}_i, y_i, h_W(\mathbf{x}_i))$ over the training data $\mathcal{D}_{tr}$. By substituting all the ingredients back to learning criterion (2.4), we can obtain the following optimization problem

$$\min_W \frac{1}{2} \|W\|_F^2 + \frac{C}{N} \sum_{i=1}^N \ell^{\log}(\mathbf{x}_i, y_i, h_W(\mathbf{x}_i)) \quad (2.11)$$

where $\|W\|_F = \sqrt{\sum_{i=1}^d \sum_{j=1}^k w_{i,j}^2}$ is the Frobenius norm. Similarly, Problem (2.11) can be equivalently obtained by the maximum a posteriori (MAP) estimation under a Bayesian model with Gaussian prior on W , that is $\mathbf{w}_y \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$ with $\mathbf{0}$ mean and $\sigma^2 I$ covariance for all $y \in \mathcal{Y}$,

$$\max_W \log p(W | \mathcal{D}_{tr}) \Leftrightarrow \max_W [\log p(\mathcal{D}_{tr} | W) p(W)] = \max_W \mathcal{L}_{\mathcal{D}_{tr}}(W) + \log \prod_{r=1}^k p(\mathbf{w}_r).$$

After some derivations, two problems are equivalent if $\sigma^2 = \frac{N}{C}$.

Large Margin Method: Support Vector Machines [25, 26]. Support vector machines [27] select the weights based on the “margin” of confidence $\langle \mathbf{w}_y, \mathbf{x} \rangle$. The loss, defined by the difference between the maximal confidence that the classifier has over all classes and the confidence it has in the correct class

$$\ell^{\text{hinge}}(\mathbf{x}, y, h_W(\mathbf{x})) = \max_{\hat{y} \in \mathcal{Y}} (\langle \mathbf{w}_{\hat{y}}, \mathbf{x} \rangle + \ell^{0/1}(\mathbf{x}, y, \hat{y})) - \langle \mathbf{w}_y, \mathbf{x} \rangle, \quad (2.12)$$

is called *hinge loss* [25, 26]. The margin on input-output pair $(\mathbf{x}, y)$ quantities by how much the true label “wins” over the incorrect ones [28]:

$$\gamma = \frac{1}{\|W\|_F} \left[\langle \mathbf{w}_y, \mathbf{x} \rangle - \max_{\hat{y} \in \mathcal{Y}} \langle \mathbf{w}_{\hat{y}}, \mathbf{x} \rangle \right] = \frac{1}{\|W\|_F} \min_{\hat{y} \in \mathcal{Y}} [\langle \mathbf{w}_y, \mathbf{x} \rangle - \langle \mathbf{w}_{\hat{y}}, \mathbf{x} \rangle]. \quad (2.13)$$

Maximizing the smallest margin over all training instances and allowing for negative margins by introducing slack variables $\xi_i \geq 0, \forall i$ is equivalent to minimizing the hinge loss in the framework of (2.4)

$$\begin{aligned} \min_W \quad & \frac{1}{2} \|W\|_F^2 + \frac{C}{N} \sum_{i=1}^N \xi_i \\ \text{s.t.} \quad & \langle \mathbf{w}_{y_i}, \mathbf{x} \rangle - \langle \mathbf{w}_{\hat{y}}, \mathbf{x} \rangle \geq \ell^{0/1}(\mathbf{x}, y_i, \hat{y}) - \xi_i, \forall i, \hat{y} \in \mathcal{Y}. \end{aligned} \quad (2.14)$$

The non-negative constraint of ξ_i is automatically satisfied with the fact that $\xi_i = 0$ if $y_i = \hat{y}$. Hence, it can be reformulated as

$$\min_W \frac{1}{2} \|W\|_F^2 + \frac{C}{N} \sum_{i=1}^N \ell^{hinge}(\mathbf{x}_i, y_i, h_W(\mathbf{x}_i)) \quad (2.15)$$

Although two models are started from different intuitions, they reach to the same framework with the only difference of loss functions. This difference leads to the following properties: Log-loss allows to define the probability output which is beneficial when logistic regression is used as one part of a whole system. Probability outputs are good for decision of actions since probability outputs represent how good it is. The objective function (2.11) is continuous and differentiable, so the parameter estimation involves unconstrained gradient based algorithm. Both algorithms can be kernelized (see section 2.1.2), but the hinge loss function usually produces sparse solution in terms of the number of support vectors, while the log-loss is generally non-sparse. The primal problem is piece-wise linear function which is non-smooth, so it is difficult to solve it directly. However, its corresponding dual problem is a quadratic program which is usually solved by sequential minimal optimization (SMO), but when N is large, it is still difficult. Both loss functions are upper-bound of the 0/1 loss, so we can consider all the algorithms as the optimization problems with one particular relaxation of 0/1 loss.

2.1.2 Kernelization

Several learning algorithms for the linear model have been mentioned in Section 2.1.1, but real-world problems usually demonstrate non-linear decision boundary [23]. Kernel provides the way for solving this problem by introducing a dot product in a so-called feature space $\mathcal{F}$ (usually in a high-dimension). An input feature map $\phi : \mathcal{X} \mapsto \mathcal{F}_{\mathcal{X}}$ is a function from an input domain $\mathcal{X}$ to its corresponding input feature space $\mathcal{F}_{\mathcal{X}}$. The class of kernels $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ corresponding to dot products in $\mathcal{F}_{\mathcal{X}}$ via the map ϕ is

$$\kappa(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{F}_{\mathcal{X}}}. \quad (2.16)$$

where only real domain of κ is considered. This makes computations of the dot products in the high-dimensional feature space as simple as the evaluations of kernel functions in

the input space, so one never explicitly computes them in high-dimensional feature space. The natural question is what kind of function κ has the property of (2.16) such that both ϕ and $\mathcal{F}_{\mathcal{X}}$ exist. We first present some important definitions to further understand the meaning of the kernel (see [23]):

Gram Matrix. Given a function $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ and inputs $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}$, the $N \times N$ matrix K with elements $K_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ is called the *gram matrix* of κ with respect to $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}$.

Positive Semi-Definite Matrix. A $N \times N$ gram matrix K satisfying

$$\sum_{i=1}^N \sum_{j=1}^N c_i c_j K_{i,j} \geq 0, \quad \forall c_i \in \mathbb{R}$$

is called *positive semi-definite matrix*.

Kernel. Let $\mathcal{X}$ be a nonempty set. A function $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ which for all $N \in \mathbb{N}$ and all $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}$ gives rise to a positive semi-definite gram matrix is called a *kernel*.

Mercer's theorem states that any kernel of a function κ corresponds to a ϕ in some reproducing kernel Hilbert space $\mathcal{F}_{\mathcal{X}}$. More details of kernels are referred to the book [23]. The frequently used kernels are linear kernel $\kappa(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$ in the original input space, Gaussian kernel $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-\frac{1}{2\sigma^2} \|\mathbf{x} - \mathbf{x}'\|^2)$ with the bandwidth parameter $\sigma > 0$, and polynomial kernels $\kappa(\mathbf{x}, \mathbf{x}') = (\langle \mathbf{x}, \mathbf{x}' \rangle + c)^d$ with a degree parameter $d \in \mathbb{N}$ and transition parameter $c \geq 0$. An important property of linear classifiers is that “kernel trick” can be easily deployed so that the model is a non-linear classifier in the original input space, but it is actually a linear classifier in the kernel induced feature spaces. By replacing the kernel with different types of kernels, the induced feature space is changed without any other modification of the whole learning algorithm.

Next, I will give the kernelized formulations of the algorithms discussed in Section 2.1.1. All the model parameters are concatenated to form an integrated column vector $\bar{\mathbf{w}} = [\mathbf{w}_{\mathbf{r}_1}; \dots; \mathbf{w}_{\mathbf{r}_k}]$. The term in (2.5) with feature map ϕ in a RKHS $\mathcal{F}_{\mathcal{X}}$ can then be reformulated as

$$\langle \mathbf{w}_y, \phi(\mathbf{x}) \rangle_{\mathcal{F}_{\mathcal{X}}} = \sum_{\hat{y} \in \mathcal{Y}} \delta(\hat{y}, y) \langle \mathbf{w}_{\hat{y}}, \phi(\mathbf{x}) \rangle_{\mathcal{F}_{\mathcal{X}}} = \sum_{\hat{y} \in \mathcal{Y}} \langle \mathbf{w}_{\hat{y}}, \delta(\hat{y}, y) \phi(\mathbf{x}) \rangle_{\mathcal{F}_{\mathcal{X}}} = \langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}, y) \rangle$$

The joint feature function is denoted by $\bar{\phi}(\mathbf{x}, y) = [\delta(\Upsilon_1, y)\phi(\mathbf{x}); \dots; \delta(\Upsilon_k, y)\phi(\mathbf{x})]$ and the indicator function $\delta(a, b) = 1$ if $a = b$, otherwise 0.

The concise representation of decision function for classification turns out to be

$$h(\mathbf{x}; \bar{\mathbf{w}}) = \arg \max_{y \in \mathcal{Y}} \langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}, y) \rangle. \quad (2.17)$$

Subsequently, support vector classification is reformulated as

$$\begin{aligned} \min_{\bar{\mathbf{w}}} \quad & \frac{1}{2} \|\bar{\mathbf{w}}\|^2 + \frac{C}{N} \sum_{i=1}^N \xi_i \\ \text{s.t.} \quad & \langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y}) \rangle \geq \ell^{0/1}(\mathbf{x}_i, y_i, \hat{y}) - \xi_i, \forall i, \hat{y} \in \mathcal{Y}, \end{aligned} \quad (2.18)$$

and the logistic regression is reformulated as

$$\begin{aligned} \min_{\bar{\mathbf{w}}, \xi} \quad & \frac{1}{2} \|\bar{\mathbf{w}}\|^2 + \frac{C}{N} \sum_{i=1}^N \left[\log \sum_{\hat{y} \in \mathcal{Y}} \exp(\xi_{i, \hat{y}}) \right] \\ \text{s.t.} \quad & \xi_{i, \hat{y}} = \langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}_i, \hat{y}) \rangle - \langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}_i, y_i) \rangle, \forall i, \hat{y} \in \mathcal{Y}. \end{aligned} \quad (2.19)$$

Slater's theorem states that strong duality holds because the objective functions are convex and the constraints are linear [29]. In other words, the optimal solution of the dual problem is also the optimal solution of its primal problem. The derivations of all dual problems are based on duality theorem [29].

Dual Problem of SVMs. The dual problem of (2.18) is

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i, \hat{y}} \alpha_{i, \hat{y}} \ell^{0/1}(\mathbf{x}_i, y_i, \hat{y}) - \frac{1}{2} \left\| \sum_{i, \hat{y}} \alpha_{i, \hat{y}} [\bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y})] \right\|^2 \\ \text{s.t.} \quad & \sum_{\hat{y}} \alpha_{i, \hat{y}} = \frac{C}{N}, \forall i \\ & \alpha_{i, \hat{y}} \geq 0, \forall i, \hat{y} \in \mathcal{Y}, \end{aligned} \quad (2.20)$$

where the primal variable $\bar{\mathbf{w}}$ is recovered by

$$\bar{\mathbf{w}} = \sum_{i, \hat{y}} \alpha_{i, \hat{y}} [\bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y})]. \quad (2.21)$$

Dual Problem of Logistic Regression. The dual problem of (2.19) is

$$\begin{aligned}
 \max_{\alpha} \quad & \sum_{i,\hat{y}} \alpha_{i,\hat{y}} \log\left(\frac{N}{C} \alpha_{i,\hat{y}}\right) - \frac{1}{2} \left\| \sum_{i,\hat{y}} \alpha_{i,\hat{y}} [\bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y})] \right\|^2 \\
 \text{s.t.} \quad & \sum_{\hat{y}} \alpha_{i,\hat{y}} = \frac{C}{N}, \forall i \\
 & \alpha_{i,\hat{y}} > 0, \forall i, \hat{y} \in \mathcal{Y},
 \end{aligned} \tag{2.22}$$

where the primal variable $\bar{\mathbf{w}}$ is recovered by

$$\bar{\mathbf{w}} = \sum_{i,\hat{y}} \alpha_{i,\hat{y}} [\bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y})]. \tag{2.23}$$

Note that all the dual problems involve the inner product of two input feature vectors in terms of L_2 norm:

$$\left\| \sum_{i,\hat{y}} \alpha_{i,\hat{y}} [\bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y})] \right\|^2 = \sum_{i,\hat{y}} \sum_{j,\hat{y}'} \alpha_{i,\hat{y}} \alpha_{j,\hat{y}'} Q_{(i,\hat{y}),(j,\hat{y}')}. \tag{2.24}$$

where $Q_{(i,\hat{y}),(j,\hat{y}')} = \langle \bar{\phi}(\mathbf{x}_i, y_i) - \bar{\phi}(\mathbf{x}_i, \hat{y}), \bar{\phi}(\mathbf{x}_j, y_j) - \bar{\phi}(\mathbf{x}_j, \hat{y}') \rangle$. In the prediction, the decision functions are also in the form of inner products.

An interesting point is that the joint feature function $\bar{\phi}(\mathbf{x}, y)$ takes the input-output pair as its arguments. The joint kernel [30] is a nonlinear similarity measure between input-output pairs $(\mathbf{x}, y)$ and $(\mathbf{x}', y')$ in terms of kernel function $\kappa : \mathcal{X} \times \mathcal{Y} \times \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$

$$\kappa((\mathbf{x}, y), (\mathbf{x}', y')) = \langle \bar{\phi}(\mathbf{x}, y), \bar{\phi}(\mathbf{x}', y') \rangle. \tag{2.25}$$

As a kernel, all the gram matrix of the function κ should be positive semi-definite. In addition to encoding the individual information of inputs and outputs, the advantage of the joint kernel is that it can also encode the given dependencies or correlations between inputs and outputs. The joint kernel used in SVMs or kernalized logistic regression for multi-class classification is tensor product kernel [30]. The tensor product kernel is defined on the maps $\bar{\phi}(\mathbf{x}, y) = \phi(\mathbf{x}) \otimes \Lambda(y)$ and $\Lambda(y) = [\delta(y, \Upsilon_1), \dots, \delta(y, \Upsilon_k)]$ where $\otimes$ is the tensor product operator [6]. Hence, it can be easily computed according to $\kappa((\mathbf{x}, y), (\mathbf{x}', y')) = \kappa(\mathbf{x}, \mathbf{x}') \langle \Lambda(y), \Lambda(y)' \rangle$ where $\langle \Lambda(y), \Lambda(y)' \rangle$ is called output kernel.

By comparing (2.20) and (2.22), the only difference is that SVM has the linear term $\sum_{i,\hat{y}} \alpha_{i,\hat{y}} \ell^{0/1}(\mathbf{x}_i, y_i, \hat{y})$ while logistic regression has the entropy term $\frac{C}{N} \sum_{i,\hat{y}} \frac{N}{C} \alpha_{i,\hat{y}} \log \frac{N}{C} \alpha_{i,\hat{y}}$. This leads to the following differences: 1) the variable $\alpha_{i,\hat{y}} = 0$ is ill-defined in kernalized logistic regression, so it does not enjoy the sparsity property. On the contrary, SVMs prefers sparse solution. Data point $\mathbf{x}_i$ associated with $\alpha_{i,\hat{y}} \neq 0$ is called support vector. 2) SVMs are solved efficiently by SMO in the dual since the primal objective (2.18) is non-smooth optimization problem. Logistic regression is easily solved in the primal form since it is unconstrained smooth optimization problem, so gradient descent such as L-BFGS can be employed. However, the difference of this aspect is not strict because SVMs trained in primal is also efficient [6] and logistic regression in dual also can be trained efficiently by SMO [31].

2.1.3 Multiple Kernel Learning

Kernel makes margin-based methods more powerful to handle non-linear decision boundary without additional efforts. In the training stage, selecting the optimal kernel function κ is still an important issue. In general, the best performing kernel function among a prefixed set of kernel functions is selected by cross-validation over a separate validation set which has to be different from the training set. Recently, multiple kernel learning (MKL) methods have been proposed to learn a combination of kernel functions and its corresponding parameters, instead of selecting only one specific kernel function.

The seminal work [32] proposed to learn an SVM with a linear combination of base kernels $K = \sum_{m=1}^M \theta_m K_m$ subject to the trace constraint $tr(K) = c$ and the positive semi-definite (PSD) constraint $K \succeq 0$. K_m is the m^{th} kernel matrix, θ_m is its corresponding weight, and c is a prefixed constant. This specific MKL form is rewritten as a semi-definite programming (SDP). Since SDP is too computationally expensive, most of the following research concentrates on designing more efficient solvers. To obviate the PSD constraint, it is a breakthrough by constraining the weights θ_m of kernels to be non-negative $\theta_m \geq 0$. This further reduces the computational complexity by transforming SDP to quadratically constrained quadratic programming (QCQP).

When all the base kernels K_m are trace normalized ($tr(K_m) = 1/c$), the constraint over weights is equivalent to $\sum_{m=1}^M \theta_m = 1$. [33] proposed a primal form based on the

mixed norm,

$$\begin{aligned} \min_{\mathbf{w}_m} & \frac{1}{2} \left(\sum_{m=1}^M \|\mathbf{w}_m\|_2 \right)^2 + C \sum_{i=1}^N \xi_i \\ \text{s.t.} & y_i \left(\sum_{m=1}^M \langle \mathbf{w}_m, \phi_m(\mathbf{x}_i) \rangle + b \right) \geq 1 - \xi_i, \forall i. \end{aligned} \quad (2.26)$$

The dual form of above problem is equivalent to the combination of kernels in the dual form of SVM [32]. The SVM-wrapped approaches [34–36] were proposed to solve the equivalent dual problems. [34] proposed to use semi-infinite linear programming (SILP) by wrapping SVM as a solver. [36] employed level-set method to further speed up SILP method. Alternative scheme proposed by [35] applied the reduced subgradient descent method on the equivalent primal formulation as

$$\begin{aligned} \min_{\mathbf{w}_m, \theta_m} & \frac{1}{2} \sum_{m=1}^M \frac{\|\mathbf{w}_m\|_2^2}{\theta_m} + C \sum_{i=1}^N \xi_i \\ \text{s.t.} & y_i \left(\sum_{m=1}^M \langle \mathbf{w}_m, \phi_m(\mathbf{x}_i) \rangle + b \right) \geq 1 - \xi_i, \forall i \\ & \sum_{m=1}^M \theta_m = 1, \theta_m \geq 0, \forall m. \end{aligned} \quad (2.27)$$

In addition to binary classification setting, they were also extended for multi-class and multi-label settings [37, 38]. All the above MKLs promote sparse solution due to the additional simplex constraint $\sum_{m=1}^M \theta_m = 1$. The sparse weights of kernels are preferred due to several reasons [39]: 1) sparse weights are easier to interpret; 2) at predicting time, the kernel function with weight zero will not be evaluated; 3) simplex constraint can be easily handled in the optimization. Following the motivation of [33], the aforementioned methods are preferred for selecting group of features for classification problem .

As frequently observed in the literature, sparseness of weights of base kernels is not always beneficial in practice [40–45]. [45] derived a projected gradient descent method

for ℓ_2 norm MKL. ℓ_p norm MKL was proposed independently by [42, 43] as

$$\begin{aligned} \min_{\mathbf{w}, b, \theta} & \frac{1}{2} \sum_{m=1}^M \frac{\|\mathbf{w}_m\|_2^2}{\theta_m} + C \sum_{i=1}^N \xi_i \\ \text{s.t.} & y_i \left(\sum_{m=1}^M \langle \mathbf{w}_m, \phi_m(\mathbf{x}_i) \rangle + b \right) \geq 1 - \xi_i, \forall i \\ & \|\theta\|_p \leq \mu, \theta \geq 0, \end{aligned} \quad (2.28)$$

where $1 \leq p \leq \infty$. In [44], the authors showed that the following ℓ_2/ℓ_q block norm

$$\begin{aligned} \min_{\mathbf{w}, b} & \frac{1}{2} \left(\sum_{m=1}^M \|\mathbf{w}_m\|_2^q \right)^{2/q} + C \sum_{i=1}^N \xi_i \\ \text{s.t.} & y_i \left(\sum_{m=1}^M \langle \mathbf{w}_m, \phi_m(\mathbf{x}_i) \rangle + b \right) \geq 1 - \xi_i, \forall i \end{aligned} \quad (2.29)$$

is more general than formulation (2.28). Specifically, the block-norm with parameter $q = \frac{2p}{p+1}$ is equivalent to having $\|\theta\|_q \leq 1$. This implies that the MKL formulation (2.29) is strictly more powerful than (2.28). [41] provided generalization bounds for both ℓ_1 and ℓ_p with $p \leq 2$ norm MKL.

The aforementioned MKLs fall into the framework of regularized risk minimization. Their regularizers are defined either over the model parameter $\mathbf{w}$ in terms of groups or over the weights θ of base kernels. There are also other forms of MKL, such as Bayesian approaches and Boosting approaches. In Bayesian approaches [46–48], the weights of base kernels are random variables with prior distributions over them. Ensemble and boosting methods [49, 50] consider the classifiers over each kernel and then combine them together for prediction. In this thesis, I mainly focus on the former methods and the latter two approaches are referred to survey papers of MKL [39, 51].

2.1.4 Feature Selection

Feature selection is crucial to many real applications such as text mining, image retrieval and bioinformatics. These applications usually contain a huge amount of features which incur very high computational costs for analysis. Pruning non-informative or noisy features can usually improve their prediction performances. Moreover, a small set of features

is beneficial to visualize or interpret the prediction results. To alleviate these issues, one can resort to embedded feature selection methods [52], which can be categorized into the following two major directions.

One way is to consider the sparsity of a decision weight vector $\mathbf{w}$ by replacing l_2 -norm $\|\mathbf{w}\|_2$ regularization in the structural risk functional (*e.g.* SVM, logistic regression) with l_1 -norm $\|\mathbf{w}\|_1$ [53–55]. A thorough study to compare several recently developed l_1 -regularized algorithms has been conducted in [56] as

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 + C \sum_{i=1}^N \ell(\mathbf{x}_i, y_i, h(\mathbf{x}_i)). \quad (2.30)$$

According to this study, coordinate descent method using one-dimensional Newton direction (CDN) achieves the state-of-the-art performance by solving l_1 -regularized models on large-scale and high-dimensional datasets. To achieve a sparser solution, the Approximation of the zeRO norm Minimization (AROM) was proposed [57] to optimize l_0 models. Its resultant problem is non-convex, so it easily suffers from local optima. However, the recent results [58] and theoretical studies [59, 60] have showed that l_p models (where $p < 1$) even with a local optimal solution can achieve better prediction performance than convex l_1 models, which are asymptotically biased [58].

Another way is to sort the weights of a SVM classifier and remove the smallest weights iteratively, which is known as SVM with Recursive Feature Elimination (SVM-RFE) [61]. However, as discussed in [36], such nested “monotonic” feature selection scheme leads to suboptimal performance. Non-monotonic feature selection (NMMKL) [36] has been proposed to solve this problem, but each feature corresponding to one kernel makes NMMKL infeasible for high-dimensional problems. Recently, Tan *et al.* [62] proposed Feature Generating Machine (FGM), which shows great scalability to non-monotonic feature selection on large-scale and very high-dimensional datasets. The key motivation is to introduce a vector of 0-1 control variables $\mathbf{d} = [d_1, \dots, d_m]^T$ in the domain of $\mathcal{D} = \{\mathbf{d} | \sum_{j=1}^m d_j \leq B, d_j \in \{0, 1\}, \forall j = 1, \dots, m\}$ where B is a budget parameter to control the sparsity of $\mathbf{d}$. FGM was proposed to solve the following mixed integer programming

$$\begin{aligned} \min_{\mathbf{d} \in \mathcal{D}} \min_{\mathbf{w}, \xi, \rho} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 + \frac{C}{2} \sum_{i=1}^N \xi_i^2 - \rho \\ \text{s.t.} \quad & y_i \mathbf{w}^T (\mathbf{x}_i \odot \mathbf{d}) \geq \rho - \xi_i, \forall i = 1, \dots, N, \end{aligned} \quad (2.31)$$

where $\odot$ is an element-wise product. It is impractical to solve this problem directly, so the problem is relaxed to a convex problem. Then, the convex problem is solved by cutting plane method [63].

The aforementioned feature selection methods [36, 56, 57, 61, 62] are usually designed for optimizing classification error only. To fulfill the needs of different applications, it is imperative to have a feature selection method designed for optimizing task-specific performance measures.

2.2 Structured Output Prediction

Several supervised learning methods in the setting of scalar outputs have discussed in Section 2.1. In this section, I concentrate on the tasks which have complex outputs, such as strings, sequences, trees and graphs. We call the complex output as an object. The intuitive method tries to transform such a complex output problem into learning problems with scalar outputs, so that the standard methods in supervised learning can be reused. However, the elements in one object are usually intertwined and strongly interdependent. Decomposition of elements in one object usually loses the important information of the interdependencies among elements. Therefore, for structured output prediction, the goal is to learn a model from training data and predict structured outputs of an input object by capturing interdependencies among their elements with some underlying structure.

Distinguished from the standard supervised learning, structured output prediction problems consider the output object $\mathbf{y} \in \mathcal{Y}$ where $\mathcal{Y}$ is reused to denote the domain of objects of interest. Given training data of input-output pairs $\mathcal{D}_{tr} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ which is a set of i.i.d. samples drawn from a fixed but unknown distribution, the goal of a structured output prediction algorithm is to find a hypothesis h such that $h(\mathbf{x})$ approximates $\mathbf{y}$ on a new data $\mathbf{x}$ from the same distribution. This leads to three important concerns in the design of structured output prediction algorithms: 1) the definition of hypothesis space and risk functions; 2) inference problem; 3) parameter estimation.

Structured output prediction problems have attracted considerable interest in the machine learning community over the last few years. Several methods [6, 9, 64] have been proposed to tackle this problem. In what follows, I will review these methods as the

extension of the previous mentioned learning methods and their corresponding variations in terms of the discussions of three main concerns.

Classification based approaches exploit a feature space $\bar{\phi}(\mathbf{x}, \mathbf{y})$ defined on input-output pairs $(\mathbf{x}, \mathbf{y})$. The learning problem is formulated as finding a $\bar{\mathbf{w}}$ -parameterized linear function in this feature space that provides a high score $\langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}, \mathbf{y}) \rangle$ to those input-output pairs appearing in the training sample. This is similar to (2.17) in supervised classification methods, but the joint feature map is based on both input and output objects. Hence, the decision function for structured output methods is defined by

$$h(\mathbf{x}; \bar{\mathbf{w}}) = \arg \max_{\mathbf{y} \in \mathcal{Y}} \langle \bar{\mathbf{w}}, \bar{\phi}(\mathbf{x}, \mathbf{y}) \rangle. \quad (2.32)$$

According to this similarity, SVMs and logistic regression models can be naturally extended for structured output prediction: margin-based methods such as Max-margin Markov Networks (M3N) and Structural SVMs [6, 64], and probabilistic model like Conditional Random Fields (CRFs) [9].

2.2.1 Probabilistic Models

To exploit the interdependencies among inputs and outputs in a structured output prediction task, a standard approach is to define a joint probability distribution over the random variables that represents the inputs and outputs. The joint probabilistic distribution allows each component to constrain each other, and thereby forces them to make decision globally. Probabilistic graphical model [65] provides a general framework to specify this distribution on a graph structure which defines the set of independencies among random variables. In this section, we discuss a variety of models by distinguishing between directed graphical model (Bayesian Network) and undirected graphical model (Markov Random Fields), as well as between generative model and conditional model. We first give some definitions in probabilistic graphical model as follows.

Definition 1. A *Bayesian Network structure* $\mathcal{G}$ is a directed acyclic graph whose nodes represent random variables $X_1, \dots, X_M$. Let $Pa_{X_i}^{\mathcal{G}}$ denote the parents of X_i in $\mathcal{G}$, and ND_{X_i} denote the variables in the graph that are not descendants of X_i . The $\mathcal{G}$ encodes the following set of conditional independence assumption, called the local independencies

$$X_i \perp ND_{X_i} | Pa_{X_i}^{\mathcal{G}}, \quad \forall i = 1, \dots, M. \quad (2.33)$$

Definition 2. A **Bayesian network** is a pair $\mathcal{B} = (\mathcal{G}, P)$ where P factorizes over $\mathcal{G}$, and where P is specified as a set of conditional probability distribution associated with $\mathcal{G}$'s nodes as

$$P(X_1, \dots, X_M) = \prod_{i=1}^M P(X_i | Pa_{X_i}^{\mathcal{G}}). \quad (2.34)$$

Definition 3. A **Markov Network structure** $\mathcal{H}$ is an undirected graph whose nodes represent random variables $\mathcal{X} = \{X_1, \dots, X_M\}$. Let $MB_{\mathcal{H}}(X)$ be the neighbors of X in $\mathcal{H}$. The $\mathcal{H}$ encodes the following equivalent Markov properties:

- *Pairwise Independency:* $X \perp Y | \mathcal{X} - \{X, Y\} : (X, Y) \notin H$.
- *Local Independency:* $X \perp \mathcal{X} - \{X\} - MB_{\mathcal{H}}(X) \mid MB_{\mathcal{H}}(X), \forall X$
- *Global Independency:* $\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z} : sep_{\mathcal{H}}(\mathbf{X}, \mathbf{Y}; \mathbf{Z})$

where $sep_{\mathcal{H}}(\mathbf{X}, \mathbf{Y}; \mathbf{Z})$ denotes that a set of nodes $\mathbf{Z}$ separates $\mathbf{X}$ and $\mathbf{Y}$.

Definition 4. A **Markov network (Markov Random Fields)** is a pair $\mathcal{M} = (\mathcal{H}, P)$ where P factorizes over a set of factors $\Phi_{\mathcal{H}} = \{\Phi_1(\mathbf{D}_1), \dots, \Phi_K(\mathbf{D}_K)\}$ as

$$P(X_1, \dots, X_M) = \frac{1}{Z} \prod_{k=1}^K \Phi_k(\mathbf{D}_k) \quad (2.35)$$

where each $\mathbf{D}_k$ ($k = 1, \dots, K$) is a complete subgraph of $\mathcal{H}$ often called a clique, and $Z = \sum_{X_1, \dots, X_M} \prod_{k=1}^K \Phi_k(\mathbf{D}_k)$ is a normalizing constant called the partition function.

For the positive factors Φ , a log-linear model over $\mathcal{H}$ is associated with the distribution P in the log-linear space $\log(\Phi_k(\mathbf{D}_k)) = w_k \phi_k(\mathbf{D}_k)$ where $\phi_k(\mathbf{D}_k)$ is a feature function from $\mathbf{D}_k$ to $\mathbb{R}$, so we can obtain a linear parameterized function

$$P(X_1, \dots, X_M) = \frac{1}{Z} \exp \left(\sum_{k=1}^K \bar{w}_k \phi_k(\mathbf{D}_k) \right), \quad (2.36)$$

where $\bar{\mathbf{w}}$ is the weight vector.

For classification, the random variables in the probabilistic model are divided into two categories: input variables $\mathbf{X} = \{X_1, \dots, X_r\}$ and output variables $\mathbf{Y} = \{X_{r+1}, \dots, X_M\}$. The goal is to estimate the probability distribution $P(\mathbf{Y}|\mathbf{X})$. To fulfill the prediction

Figure 2.1: Three models with different linear chain structures.

task, a *generative model* is to estimate the joint probabilistic distribution $p(\mathbf{y}, \mathbf{x})$ and then obtain $p(\mathbf{y}|\mathbf{x})$ by Bayesian rule, while a *conditional model* directly estimates $p(\mathbf{y}|\mathbf{x})$. In what follows, we present several popular probabilistic models in the structure of linear chains with $\mathbf{X} = \{X_1, \dots, X_M\}$ and $\mathbf{Y} = \{Y_1, \dots, Y_M\}$ so as to explain these models clearly in terms of their underlying conditional independencies among random variables.

Hidden Markov Model (HMM) [66]. The graph structure of an HMM is shown in Figure 2.1. According to the definition of Bayesian Networks, an HMM encodes the joint probabilistic distribution $P(\mathbf{Y}, \mathbf{X})$ as

$$P(\mathbf{Y}, \mathbf{X}) = \prod_{t=1}^M P(X_t|Y_t)P(Y_t|Y_{t-1}) \quad (2.37)$$

where $P(Y_t|Y_{t-1})$ is the state transition probability with the prior probability in the case of $t = 1$, and $P(X_t|Y_t)$ is the state-observation probability.

Maximum Entropy Markov Model (MEMM) [67]. This model is a straightforward application of logistic regression models to sequence labeling problems. The mixture structure with directed and undirected edges are shown in Figure 2.1. According to Bayesian Networks and each output variable conditioning on whole observations, the conditional probabilistic distribution is formulated as

$$P(\mathbf{Y}|\mathbf{X}) = \prod_{t=1}^M P(Y_t|Y_{t-1}, \mathbf{X}) \quad (2.38)$$

where $P(Y_t|Y_{t-1} = \Upsilon_k, \mathbf{X})$ is a logistic regression model.

Conditional Random Fields (CRF) [9]. Different from previous models, a CRF assumes that output random variables are symmetrically dependent, which is encoded

in the undirected graph shown in Figure 2.1. According to Markov random fields, the conditional probability $P(\mathbf{Y}|\mathbf{X})$ is defined as

$$P(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp \left(\sum_{t=1}^M \sum_{r=1}^s \bar{w}_r \phi_r(Y_t, Y_{t-1}, \mathbf{X}) \right), \quad (2.39)$$

where $Z(\mathbf{X})$ is an instance-specific normalization function

$$Z(\mathbf{X}) = \sum_{\mathbf{Y}} \exp \left(\sum_{t=1}^M \sum_{r=1}^s \bar{w}_r \phi_r(Y_t, Y_{t-1}, \mathbf{X}) \right), \quad (2.40)$$

and the cliques in the chain structure are $\mathbf{D}_t = (Y_t, Y_{t-1}), \forall t = 1, \dots, M$.

As HMMs are generative models, the observation probability distribution has to be modeled. This prevents HMMs from using overlapping and non-independent features since these features makes HMMs complex and intractable. Both MEMMs and CRFs consider to directly model the conditional probability distribution for prediction, so they can incorporate overlapping nonindependent features naturally. As stated in [9], CRFs overcome the label bias problem which makes MEMMs suboptimal. In addition to the simplest linear chain structure, the general graph structures associated with HMMs and CRFs are Bayesian Networks and general CRFs. For specific tasks, different priors are employed to construct the underlying graphs. In this case, we consider the general CRFs model with feature map $\bar{\phi}(\mathbf{x}, \mathbf{y})$. Take the linear chain CRFs for example, $\bar{\phi}(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^M \phi_r(y_t, y_{t-1}, \mathbf{x})$ where ϕ_r is a feature vector. For different graph structures, the differences of $\bar{\phi}(\mathbf{x}, \mathbf{y})$ come from the definition of feature functions. Therefore, the conditional probability $p(\mathbf{y}|\mathbf{x})$ of general CRFs is defined as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp (\bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}, \mathbf{y})) \quad (2.41)$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp (\bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}, \mathbf{y}))$. Feature functions are usually defined on the set of cliques over a given graph structure. Besides, other forms of feature functions are also considered in the literature[68, 69]. Relational Markov networks [68] are special cases of general CRFs in which the graphical structure and parameter tying are determined by an SQL-like syntax. Markov logic networks [69] are probabilistic logic in which there are parameters for each first-order rule in a knowledge base.

For the fully observed cases, CRFs are trained by maximizing the training data log-likelihood with respect to model parameter $\bar{\mathbf{w}}$. The log-likelihood for the i^{th} input-output pair $(\mathbf{x}_i, \mathbf{y}_i)$ is defined as

$$\mathcal{L}_i(\bar{\mathbf{w}}) = \log p(\mathbf{y}_i | \mathbf{x}_i; \bar{\mathbf{w}}) = \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) - \log \sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \exp \left\{ \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \right\}. \quad (2.42)$$

To avoid overfitting, maximization of a posteriori estimation $p(\bar{\mathbf{w}} | \mathcal{D}_{tr}) \propto \prod_{i=1}^N p(\mathbf{y}_i | \mathbf{x}_i, \bar{\mathbf{w}}) p(\bar{\mathbf{w}})$ is employed with a spherical Gaussian weight prior, that is $\bar{\mathbf{w}} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ [70],

$$\mathcal{L}'(\bar{\mathbf{w}}) = \sum_{i=1}^N \mathcal{L}_i(\bar{\mathbf{w}}) - \frac{1}{2\sigma^2} \|\bar{\mathbf{w}}\|^2. \quad (2.43)$$

For a training set with N input-output pairs, the model parameters are learnt by solving the following optimization problems,

$$\min_{\bar{\mathbf{w}}} \sum_{i=1}^N \left(\log \sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \exp \left\{ \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \right\} - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right) \quad (2.44)$$

$$\min_{\bar{\mathbf{w}}} \frac{1}{2\sigma^2} \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\log \sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \exp \left\{ \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \right\} - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right) \quad (2.45)$$

which are for non-regularized [9] and regularized [7] cases, respectively.

In addition, several variations are proposed in the literature [71–73]. Softmax-margin method [71] has been proposed to incorporate the user specific loss term

$$\min_{\bar{\mathbf{w}}} \frac{1}{2\sigma^2} \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\log \sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \exp \left\{ \bar{\mathbf{w}}^T \Phi(\mathbf{x}_i, \mathbf{y}') + \Delta(\mathbf{y}_i, \mathbf{y}') \right\} - \bar{\mathbf{w}}^T \Phi(\mathbf{x}_i, \mathbf{y}_i) \right) \quad (2.46)$$

where $\Delta(\mathbf{y}_i, \mathbf{y}')$ is a loss term which stands for the number of label difference between $\mathbf{y}_i$ and $\mathbf{y}'$. The unified methods [72, 73] were proposed to integrate the concept of a margin and a temperature ϵ into CRFs. Given the temperature $\epsilon \geq 0$, the unified formulation is

$$\min_{\bar{\mathbf{w}}} \frac{1}{2\sigma^2} \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\epsilon \log \sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \exp \left\{ \frac{\bar{\mathbf{w}}^T \Phi(\mathbf{x}_i, \mathbf{y}') + \Delta(\mathbf{y}_i, \mathbf{y}')}{\epsilon} \right\} - \bar{\mathbf{w}}^T \Phi(\mathbf{x}_i, \mathbf{y}_i) \right) \quad (2.47)$$

where the conditional probability distribution is

$$p(\mathbf{y} | \mathbf{x}; \epsilon) = \frac{1}{Z_\epsilon(\mathbf{x})} \exp \left\{ \frac{\bar{\mathbf{w}}^T \Phi(\mathbf{x}, \mathbf{y}') + \Delta(\mathbf{y}, \mathbf{y}')}{\epsilon} \right\}, \quad (2.48)$$

where $Z_\epsilon(\mathbf{x})$ is the normalization term. When ϵ approaches to 0, this probability distribution gets concentrated around its maximal value, so it approximates structural SVMs [6] or max-margin Markov networks [64] (see Section 2.2.2). When $\epsilon = 1$, it is the softmax-margin formulation. The work [72] gives the explanation of the augmented loss term from posterior perspective, where the corresponding variations of margin re-scaling and slack re-scaling can be obtained by different priors.

As discussed above, a variety of probabilistic modeling methods for structured outputs have been mentioned. In other words, the loss functions and the optimization problems have been defined. The question now is to find one hypothesis which can better fit the given data. This process is called training. According to the formulations of aforementioned methods, it is simplified to find an optimal solution of $\bar{\mathbf{w}}$. In addition to the likelihood or MAP estimation, the average single-node log-likelihood [74] and piecewise pseudo-likelihood [75] are also employed for efficient training of CRFs. For general CRFs, the objectives (2.44), (2.45), (2.46) and (2.47) are all smooth and convex functions. The optimal solution can be obtained by gradient descent methods, such as limited-memory quasi-Newton (L-BFGS) [76] and preconditioned conjugated gradient descent [7]. However, gradient-based training requires computing marginal distribution for computing descent direction, which could be computational expensive. If the graph has small treewidth such as sequences and trees, the junction tree algorithm [77] can be used to exactly compute the marginals, otherwise approximate inference algorithms are required, such as sampling-based approaches like Markov chain Monte Carlo (MCMC) [78], loopy belief propagation method [79] and variational methods [80]. These methods can also compute the most likely configuration given the input and the learnt model.

2.2.2 Margin-based Methods

Margin-based methods can be considered as the extensions of multi-class SVM for structured outputs. The objective requires that the desired label should have the highest satisfaction than the rest of labels in the feasible domain with a margin at least one

$$\bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \geq 1, \forall i, \mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i) \setminus \{\mathbf{y}_i\}. \quad (2.49)$$

where $\mathbf{y}_i$ is the true label of the given input $\mathbf{x}_i$. The empirical risk of this formula can be reduced to zero only when the data is separable. For the non-separable case, slack variables are introduced in several ways, which are originated from multi-class classification [25, 26]. The method [25] introduced one slack variable for each constraint, while [26] proposed to introduce one slack for a set of constraints over one input. Comparing these two methods, the method of [26] has the advantages of efficient structural extensions. By following [26], the objective of the learning methods can be formulated by minimizing the regularized structural risk functional as

$$\min_{\bar{\mathbf{w}}} \lambda \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\max_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i) \setminus \{\mathbf{y}_i\}} \left\{ 1 + \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \right\} - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right)^+, \quad (2.50)$$

where $(a)^+ = \max(0, a)$, and λ is the tradeoff parameter. This is known as a hidden Markov Support Vector Machine (HMSVM) [81]. However, the penalty strategy in HMSVM never considers the differences between $\mathbf{y}$ and an alternative configuration $\mathbf{y}'$ because it penalizes all alternative hypothesis based on the most promising alternative hypothesis. Max-Margin Markov Network (M3N) [64] considered replacing the constant one with a loss term $\Delta(\mathbf{y}, \mathbf{y}')$, i.e., the number of label difference between $\mathbf{y}$ and $\mathbf{y}'$,

$$\min_{\bar{\mathbf{w}}} \lambda \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\max_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i) \setminus \{\mathbf{y}_i\}} \left\{ \Delta(\mathbf{y}_i, \mathbf{y}') + \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \right\} - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right)^+, \quad (2.51)$$

which is equivalent to the Structural Support Vector Machines (Structural SVMs) with margin re-scaling method [6]. This comes from the intuition that the differences of confidences between the true label and an alternative configuration are large if the alternative configuration disagrees with the true label at many nodes. An alternative way is to change the scaling of the slack variables as

$$\min_{\bar{\mathbf{w}}} \lambda \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\max_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i) \setminus \{\mathbf{y}_i\}} \Delta(\mathbf{y}_i, \mathbf{y}') \left\{ 1 + \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right\} \right)^+, \quad (2.52)$$

which overcomes the drawback of the margin re-scaling method since it tries to make very large difference between competing labels that differ in many positions. This strategy is known as Structural SVMs with slack re-scaling [6]. Note that if the loss term $\Delta(\mathbf{y}_i, \mathbf{y}) = 0$, we can simplify them to be

$$\min_{\bar{\mathbf{w}}} \lambda \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\max_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \left\{ \Delta(\mathbf{y}_i, \mathbf{y}') + \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') \right\} - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right)^+, \quad (2.53)$$

$$\min_{\bar{\mathbf{w}}} \lambda \|\bar{\mathbf{w}}\|^2 + \sum_{i=1}^N \left(\max_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x}_i)} \Delta(\mathbf{y}_i, \mathbf{y}') \left\{ 1 + \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}') - \bar{\mathbf{w}}^T \bar{\phi}(\mathbf{x}_i, \mathbf{y}_i) \right\} \right)^+, \quad (2.54)$$

which are easier to solve than (2.51) and (2.52).

Different from CRFs, the objectives of large margin methods such as (2.50), (2.53) and (2.54) are non-differentiable convex functions, so gradient descent methods cannot be applied for large margin methods. However, subgradient methods [82] can be applied to solve the non-differentiable convex functions. A stochastic subgradient descent method (averaged perceptron) [83] is applied to solve the special cases of (2.53) with $\lambda = 0$ and $\Delta(\mathbf{y}, \mathbf{y}') = 0$. An alternative method is the Pegasos [84] which projects the descent solution onto a ℓ_2 ball so as to guarantee the optimal solution. In order to calculate the subgradient, the loss augmented maximization problem for structured output has to be solved. Although the margin based loss can be rewritten as an exponential number of constraints, optimal solution may not enforce all the constraints [6]. According to this assumption, cutting plane method [63] was proposed to solve (2.53) and (2.54) in N -slack formula with a polynomial-sized subset of the constraints with accuracy at least ϵ . The one slack formula in [14] showed the faster convergence than N -slack formula. Other methods to apply cutting plane algorithm to solve non-smooth problems were proposed by [85, 86] which are closely related to the bundle method, but differs in the regularizer. Instead of using cutting plane method to generate constraints, the reformulation of exponential constraints quadratic program into a polynomial-sized problem were also used [64, 87, 88]. Moreover, the min-max formulation by rearranging the constraint was considered for each training instance [89, 90].

2.2.3 Feature Engineering Issue

Structured output prediction [6, 9, 17] has been successfully applied to the problems with strong interdependencies among the output variables. In the realm of Natural Language Processing (NLP), various tasks are formulated as structured output prediction problems. A typical example is POS tagging which assigns a specific POS tag to each word in the input sentence. The POS tag of one word is strongly correlated with the POS tags of its neighbors under the dependencies encoded in the linear chain structure [6, 9]. More complicated structured output dependencies could be formed as trees or graphs,

such as Context-Free Grammar (CFG) [6], dependency parsing tree [10, 91], noun phrase coreference [92], and factor graph for relation extraction [93]. Note that there exist exact inference methods for sequences and trees. For the tasks with general output structures (*eg.*, the pairwise fully connected undirected graph), the exact inference problem is intractable. In such case, approximate inference methods are usually pursued to obtain approximate solutions [94].

The major advantage of structured output prediction models such as Conditional Random Fields (CRFs) [9] and Structural Support Vector Machines (Structural SVMs) [6, 17] is that they can easily integrate prior knowledge in a specific domain by feature engineering. For example, CRFs can account for overlapping features (*eg.*, first-order or even higher-order linear chain) on the whole observation sequence [9]. Structural SVMs [6, 14] relies on joint feature maps over the input-output pairs to represent the same set of features. The arbitrary non-independent features are usually extracted from a given input-output pair using a set of predefined templates (rules or feature functions), such as sequence learning [5, 8] and dependency parsing [10]. The templates are generally provided by domain experts.

During last decade, structured prediction algorithms take more effort on how to model the interdependent structures among the output variables [6, 9, 17], but less consideration is taken on the feature engineering [18]. Templates can be arbitrarily defined according to specific applications by exploring any internal or external knowledge as much as possible. And then, features extracted by different templates are concatenated into the learning models so as to boost prediction performance. However, template design is a non-trivial and tedious task for the general users because features generated from arbitrary templates may be redundant or non-informative. For a given learning method, new templates are created, while old templates might be removed. This gradually improve the prediction performance by manual selection for a good set of templates [95]. Structured prediction models, such as CRFs and Structural SVMs, only treat the features generated from each template equally without exploiting the importance of each template or its generated features. Therefore, some improper templates, which may generate conflicting or noisy features, degrade these structured prediction models.

In order to prune noisy or non-informative features for improved generalization and fast prediction, feature selection strategy can be incorporated into the structural prediction models in two directions. One way is to treat each instantiated feature from each template independently. For example, CRFs with ℓ_1 regularization has been explored for activity recognition by [96]. However, progressively reducing the features by ℓ_1 regularization methods may degrade the performance vastly due to the interdependencies among features in the same group. Another way is to consider that features are naturally grouped together by templates [18], and each group of features forms a base kernel in the framework of Multiple Kernel Learning (MKL) [32]. As discussed in [33, 97], group lasso and MKL can be used to obtain the group sparsity via the $\ell_{2,1}$ mixed norm and its square, respectively. More general groups sparsity regularizer such as ℓ_p - ℓ_q has also been explored [98] where $0 \leq p \leq 1$ and $1 \leq q \leq 2$. A plethora of research progress on group lasso and MKL have been made. Many works [34–36, 97, 99–101] focus on convergence and scalability issues of their optimization problems. Zien and Ong [37] formulated MKL in terms of structural prediction, but only focused on multi-class classification. Although SVM-based wrapped methods [34–36] with SMO solver [102] can efficiently solve some specific formulations of MKLs, it is intractable for these MKL and group lasso methods to deal with structured prediction because of the exponential number of constraints. Instead of the natural groups formed over features, we notice that the training data can also be naturally separated into groups, known as “Learning from structured data” [103]. However, groups in the feature level is the focus of this thesis.

Recently, Martins *et al.* [18] proposed an online multi-kernel learning strategy, namely OnlineMKL, to cater for the constraints of structure prediction. Recall that, due to the arbitrary design of templates in NLP, the extracted features are usually very high-dimensional and very sparse. Though the gradient descent step in each round of OnlineMKL training is efficient, the subsequent proximal projection step on the dense and very high dimensional weight vector of the learning model is very time-consuming (see Section 3.2.4). Apart from these computational issues, Dekel and Singer [104] stated that there is no notion of statistical generalization in the online learning setting. Additional online-to-batch strategy is required to convert online classifiers for the batch-mode learning setting because it fits most real-world problems more naturally [104, 105]. Even worse,

it is nontrivial to determinate the termination condition for online methods in order to achieve the optimal batch-mode prediction performance. Due to the recent development in online learning [106, 107], the last classifier returned by online learning does equip with a generalization error bound when loss function is strongly convex, even though it is as good as one provided by the suffix average. However, this generalization result for the last classifier returned by online method does not apply to that returned by OnlineMKL because both its loss function and mixed norm regularizer are not strongly convex.

2.2.4 Performance Evaluation Issue

Machine learning methods have been widely applied to a variety of learning tasks (*e.g.* classification, ranking, structure prediction, etc) arising in computer vision, text mining, natural language processing and bioinformatics applications. Depending on applications, specific performance measures are required to evaluate the success of a learning algorithm [13]. For instance, the error rate is a sound judgment for evaluating the classification performance of a learning method on datasets with balanced positive and negative examples. Generally, positive examples are usually very few in text classification. One can simply assign all testing examples with the negative class (the major class), this trivial solution can easily achieve very low error rate due to the extreme imbalance of the data. However, the goal of text classification is to correctly detect positive examples. Hence, the error rate is considered as a poor criterion for the problems with highly skewed class distributions [13]. To address this issue, F_1 -score and Precision/Recall Breakeven Point (PRBEP) are employed as the evaluation criteria for text classification. Beside this, in information retrieval, search engine systems are required to return the top k documents (images) with the highest precision because most users only scan the first few of them presented by the system, so precision/recall at k are preferred choices.

Instead of optimizing the error rate, Support Vector Machine for multivariate performance measures (SVM^{perf}) [13] was proposed to directly optimize the losses based on a variety of multivariate performance measures. A smoothing version of SVM^{perf} [108] was proposed to accelerate the convergence of the optimization problem specially for PRBEP and area under the Receiver Operating Characteristic curve (AUC). Structural SVMs are considered as the general framework for optimizing a variety of loss functions [6, 14, 85].

Figure 2.2: Example images from UIUC-Sport dataset. For each image, the ground truth keywords are shown at the bottom. Image annotation aims at predicting these keywords for the given images.

In addition, there are other works which optimize specific multivariate performance measures, such as F-score [109], normalize discount cumulative gain (NDCG) [110], ordinal regression [111], ranking loss [112] and so on.

As digital cameras become widespread, personal photos are easily taken by individuals daily. Sharing photos through the Internet (*e.g.* *Flickr.com*, *Facebook*, etc), which becomes a common practice, leads to archives in the order of millions of images. This convenience, however, makes the management of images highly challenging for users. The annotated and indexed images enable effective search and retrieval [113, 114], but manual annotation is prevented from the large scale collections of images due to the time-consuming and extremely labor intensive work [115–117]. Although computer-assisted systems are desired to lessen these difficulties by automation, it is still challenging for computers to understand the semantics of these images properly [116].

In the field of computer vision, automatic image annotation [12, 118–121] has been considered as one crucial strategy to improve the general image understanding. Its objective is to predict relevant keywords from given keyword vocabularies for an image (Figure 2.2). After annotating the images with keywords in the database, these images can be easily accessed with the annotated keywords [122]. Thus many computer vision tasks can vastly benefit from image annotation. For example, keyword based image retrieval [123] allows users to present a keyword as a textual query, and find the relevant images by matching the textual query with the keywords of images in the database.

To evaluate the performance of automatic image annotation, a variety of performance criteria from different considerations have been proposed in different objective/applications of image annotation. For instance, if the objective of image annotation is to facilitate keyword based image retrieval, one usually assigns each image k tags, then performs image retrieval by using each tag as a query. Based on the returned image list, the mean precision, mean recall, F1 measure, and precision/recall break-even point (PRBEP) over all the tags are calculated as the image annotation performance measure [121, 124, 125]. According to the categorization of performance measures in [11], most of the above measures belong to a large family of measures called bipartition. The measures for keyword based image retrieval are known as *macro-averaging measures* [15]. If the objective of image annotation is to facilitate personal photo management, one wants to annotate each image correctly and completely for his/her future management. In this case, image annotation is also known as image tagging [126]. *Example-based measures* are usually preferred and are calculated for each image, such as the precision (also known as success@ k or tag precision), recall (also known as tag recall), F_1 measure of the assigned tags [12, 120, 126, 127]. If the image annotation is cast as a multi-label classification in pattern recognition, Hamming measure is frequently used as the evaluation criterion [11, 15, 128]. In addition, *micro-averaging measures* have also been used in evaluating image annotation methods [129].

In the ideal case, all performance measures should be able to evaluate the performance of the image annotation methods. In other words, it is expected that the designed image annotation methods can demonstrate good performance for all applications. Although the goal of image annotation is to automatically annotate each image with correct and complete tags, current image annotation techniques are still far away from reaching this goal. Therefore, in practice, it is more realistic to prefer good performance for a specific application, *i.e.*, good performance for a certain objective specific measure. For example, better macro-averaging measures are preferred if one wants to perform image retrieval with image annotation. And better example-based measures are preferred if one wants to perform image tagging in his personal photo management. However, most of image annotation methods in the literature do not tackle image annotation towards these objective specific measures, so they are inevitably trapped into the suboptimal performance of these measures.

The learning models for image annotation and learning methods for optimizing specific performance measures are most relevant to the thesis are discussed as follows. Since generative models [130–132] were usually designed to estimate the joint probability distribution over both features and keywords, their solutions may not be optimal for predictive performances [121]. Hereafter, we concentrate our discussions on discriminative models for image annotation. Two popular approaches of discriminative methods are margin-based methods and nearest neighbor based methods. Margin-based methods [133, 134] learn a model for each keyword individually, and use these models for each test image to predict whether the keyword belongs to the image. The discriminative power of these methods may be impaired when keywords are semantically overlapped or imbalanced in the dataset [135]. To overcome the above issues, several works [128, 136] were proposed to harness the keyword correlation as the additional information. On the other hand, given an image, nearest neighbor based methods are to find its κ nearest neighbors from the training set and assign all the keywords of its neighbors to this image [118]. The work [121] attempted to learn a weighted nearest neighbor model by maximizing the log-likelihood of annotation in the training data. Both methods benefit from the combination of metrics on the different aspects of image contents.

In machine learning, image annotation is cast as multi-label classification problems [11]. Distinguishing from the traditional single-label classification, one of major challenges brought by multi-label classification is how to evaluate the performance of a learning algorithm with respect to the multi-label output. In order to account for the specific requirements of different applications, lots of performance measures are specially designed to evaluate multi-label classification [11, 137, 138]. Because learning algorithms in the literature seldom consider these measures as the criteria to optimize, they unavoidably get stuck into the sub-optimality. A few methods are specially designed to optimize one specific objective measure for multi-label classification. For example, ranking based methods [134, 139, 140] were proposed to optimize ranking loss function directly for multi-label classification, but it is not trivial to determine the number of labels to predict for an unknown input. Besides this, *macro-averaging* F1 score is also optimized in the literature such as Bayesian online multi-label classification framework [141] and reverse multi-label learning [15]. However, their algorithms cannot be adapted for other measures.

Chapter 3

Structured-Prediction-Based Feature Selection

3.1 Introduction

In this Chapter, I propose a Multiple Template Learning (MTL^{struct}) paradigm to learn the weight of each template and the structured prediction model in the batch mode, simultaneously. Notice that MTL is also the abbreviation of Multi-Task Learning [142]. In what follows, MTL only stands for Multiple Template learning. Following the group sparsity, to capture the dependency among a group of features instantiated from one template, learning the weighting of these groups is formulated as a Multiple Kernel Learning (MKL) problem. This specific MKL problem usually involves exponential number of constraints due to the interdependencies among output variables. To solve this MKL problem, an efficient cutting plane algorithm is proposed. Moreover, its convergence is proved. Recently, Martins et. al. (2010) proposed an online multi-kernel learning strategy, namely OnlineMKL, to cater for the constraints of structure prediction. The discussion of OnlineMKL is aforementioned in Section 2.2.3. The main contributions of this Chapter are listed as follows,

- Following the idea of group feature selection (template selection), I propose to learn the large margin based structured prediction model in the batch mode. Hence, the generalization performance can be guaranteed. As opposed to OnlineMKL, the empirical risk is naturally considered as the criterion of termination condition for the proposed MTL^{struct} .

- An adapted cutting plane algorithm is proposed to tackle the MKL problem with exponential number of constraints, and the convergence is guaranteed. This also leads to very sparse solution of the learned weights of templates, which is effective to prune noisy or non-informative templates for performance improvement, faster prediction and enhancing semantic interpretation.
- Detailed time complexities of state-of-the-art structured prediction methods are presented. The analysis reveals that the proposed MTL^{struct} , which avoids the proximal projection on the dense weight vector, is much more efficient than OnlineMKL in the case of very sparse and high-dimensional data. This characteristic of data is common in NLP applications when hundreds of templates are used. And the training efficiency of the proposed MTL^{struct} is also very comparable with Structural SVM. Moreover, since MTL^{struct} can automatically remove redundant templates, the testing efficiency of MTL^{struct} is much improved over structured SVM and various CRFs without performance reduction.
- Two well-known structured prediction tasks, i.e. sequence labeling and dependency parsing, are showcased in this thesis. Extensive experimental results demonstrate that the proposed paradigm is more effective than OnlineMKL to learn the importance of each template for the structured prediction tasks, and outperforms other state-of-the-art baseline methods. The learned weights have the ability to avoid the degraded performance which is caused by adding poorly designed or even conflicting templates into the learning model. This phenomenon is demonstrated in the experiments, so the proposed MTL^{struct} can tackle the challenge of choosing and weighting the informative templates for the optimal prediction performances.
- The proposed learning paradigm is further extended to solve structured output problems with p -block norm regularization. Experimental results on sequence labeling tasks show competitive results when $p \in [1, 2)$; otherwise, the overall performance may degrade vastly.

The rest of this Chapter is organized as follows. The proposed MTL^{struct} framework for structured prediction is presented in Section 3.2 and its extension to p -block norm

regularization is given in Section 3.3. Section 3.4 presents the experimental results of two showcases, i.e, sequence labeling and dependency parsing. Conclusive remarks are given in Section 3.5.

3.2 Multiple Template Learning

CRFs and Structural SVMs have already been used to model many structured prediction problems, such as sequence labeling with a first-order or even higher-order linear chain, and syntactic parsing with a directed tree. More general structures are possible if a feasible inference method exists. CRFs are graphical models by defining conditional probability density function over the given graph structure; while Structural SVMs incorporate the structural information by defining joint feature mapping under large margin theory. Even though two models are derived under different principles, they all confront the same challenge: how to extract features from structured input-output pairs so as to obtain better prediction performance. The proposed MTL^{struct} framework mainly focuses on how to efficiently and effectively manage the given set of templates and learn a better structured prediction model, simultaneously.

3.2.1 Feature v.s. Template

For the problems with structured inputs, features are usually not explicitly defined. For instance, part-of-speech tagging in NLP is to label each token with a specific tag in the given token sequence (sentence). In addition to the current token, the neighbors and their associated tags can be considered as the important features to determine the tag of the current token. All these intuitive information can be represented by feature functions called templates [5, 8, 10]. Hereby, one considers the templates as some predefined rules for feature extraction.

Given a training dataset $\mathcal{D} = \{(\mathcal{X}_i, \mathcal{Y}_i)\}_{i=1}^n$ with n structured input-output pairs $(\mathcal{X}_i, \mathcal{Y}_i)$ where $\mathcal{X}_i \in \mathbb{X}$ and $\mathcal{Y}_i \in \mathbb{Y}$ can be any structural objects, for instance, sequences, trees, or general graphs. Assume that there are m templates denoted by the operator $\kappa_j(\cdot), \forall j = 1, \dots, m$ over the domain of structural input-output pairs to generate a list of features, e.g., the bigram rule in the part of speech tagging. By applying $\kappa_j(\cdot)$ to all

pairs of $\mathcal{D}$, one can instantiate a set $\kappa_j(\mathcal{D}) = \{\kappa_j^1, \dots, \kappa_j^{d_j}\}$ with d_j different features. Therefore, the set of features over $\mathcal{D}$ are $\kappa(\mathcal{D}) = \{\kappa_1(\mathcal{D}), \dots, \kappa_m(\mathcal{D})\}$. The size of total features $|\kappa(\mathcal{D})|$ may be smaller than $\sum_{j=1}^m |\kappa_j(\mathcal{D})|$ since one feature may be generated from more than one template. The case of one feature in more than one group is not the concern of this thesis, so one can augment the index of the applied templates to the instantiated feature such that one feature belongs to only one group. Each input-output pair $(\mathcal{X}_i, \mathcal{Y}_i)$ now can be represented by a real value feature vector $\Phi(\mathcal{X}_i, \mathcal{Y}_i)$ using the same set of features in $\kappa(\mathcal{D})$. Since $\kappa(\mathcal{D})$ is extracted from $\mathcal{D}$ and a small subset of features in $\Phi(\mathcal{X}_i, \mathcal{Y}_i)$ could be activated, so each instance may have a very sparse feature representation.

Previous methods for structured inputs and outputs did not consider the properties of features, and directly used $\kappa(\mathcal{D})$ as the feature representation. Taking part-of-speech tagging for example, CRFs use the concatenation of all the different features instantiated from each template and the real-value of each feature is the occurred frequency. Actually, the subset of features $\kappa_j(\mathcal{D})$ applied by the j th template can be naturally formulated as a group. Each group of features stands for some specific meaning of the applied template, such as word, bi-gram, distance, orientation, and position related to the current token. Moreover, templates can be either defined arbitrarily by someone without any prior knowledge, or designed specially by domain experts. It is easy to see that hundreds of templates are required for typical NLP tasks. Due to the diversity of features, the importance of each template should be assigned differently. However, due to numerous predefined templates, it is infeasible and tedious for users to select or weight the informative templates for the optimal prediction performance.

It is imperative to have a unified way to properly interpret and manage the diverse templates. In particular, a proper template weighting scheme can prune away poorly designed or conflicting templates, and amplify the effective templates. Thereafter templates can be designed without cautions of performance reduction. In other words, removing noisy templates can improve the performance or removing redundancy templates can speedup the prediction process. As mentioned before, a template can instantiate a set of features on the given dataset. Hence, selecting template is equivalent to selecting the set of instantiated features by this template. If the set of features instantiated by one

template is considered as a group, group feature selection can naturally achieve the goal of template selection by group sparsity [33, 97].

Based on the property of group sparsity in NLP, Martins *et al.* [18] recently proposed an online multi-kernel learning strategy, namely OnlineMKL, to cater for the constraints of structure prediction. Recall that, due to the arbitrary design of templates in NLP, the extracted features are usually very high-dimensional and very sparse. Though the gradient descent step in each round of OnlineMKL training is efficient, the subsequent proximal projection step on the dense and very high dimensional weight vector of the learning model is very time-consuming (see Section 3.2.4). Even worse, it is nontrivial to determinate the termination condition for online methods in order to achieve the optimal batch-mode prediction performance. In the next subsections, a novel structured prediction model with group sparsity will be proposed to interpret the importance of templates, where features generated from the same template naturally form a group. The proposed model can be essentially deemed as a special MKL problem, where the base kernels are defined in accord with templates. Then an efficient cutting plane algorithm is proposed to solve this MKL.

3.2.2 Model Formulation

Given a training dataset $\mathcal{D}$, the i^{th} input-output pair $(\mathcal{X}_i, \mathcal{Y}_i)$ can be represented by $\Phi(\mathcal{X}_i, \mathcal{Y}_i) = [\Phi_1(\mathcal{X}_i, \mathcal{Y}_i); \dots; \Phi_m(\mathcal{X}_i, \mathcal{Y}_i)]$ using the group representation of $\kappa(\mathcal{D})$ where semicolon is used to concatenate column vectors, for concise representation. The goal of structured prediction learning is to learn the hypotheses $f : \mathbb{X} \rightarrow \mathbb{Y}$. According to Structural SVMs, the compatibility function $F : \mathbb{X} \times \mathbb{Y} \rightarrow \mathbb{R}$ over the input-output pairs are pursued and the prediction function can be derived by maximizing F over the output space $\mathbb{Y}$ for a given input $\mathcal{X} \in \mathbb{X}$. The general hypotheses f are the parameterized functions with parameter vector $\mathbf{w}$ as

$$f(\mathcal{X}; \mathbf{w}) = \arg \max_{\mathcal{Y} \in \mathbb{Y}} F(\mathcal{X}, \mathcal{Y}; \mathbf{w}) = \arg \max_{\mathcal{Y} \in \mathbb{Y}} \mathbf{w}^T \Phi(\mathcal{X}, \mathcal{Y}). \quad (3.1)$$

For structural outputs, the standard zero-one cost function frequently used in classification is not appropriate. Most applications need task-specific cost function, so one can define a general cost function $\Delta(\mathcal{Y}, \mathcal{Y}')$ if the instance with the true output $\mathcal{Y}$ is assigned

to be $\mathcal{Y}'$ with the property $\Delta(\mathcal{Y}, \mathcal{Y}) = 0$. The margin re-scaling with general loss functions and linear penalty term can be formulated as a minimization of the regularized empirical risk:

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{n} \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i, \forall \mathcal{Y}'_i \in \mathbb{Y} : \mathbf{w}^T \delta \Phi^i(\mathcal{Y}'_i) \geq \Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \xi_i \end{aligned} \quad (3.2)$$

where C is a trade-off parameter between training error minimization and margin maximization, and the difference of feature vectors is denoted by $\delta \Phi^i(\mathcal{Y}'_i) = \Phi(\mathcal{X}_i, \mathcal{Y}_i) - \Phi(\mathcal{X}_i, \mathcal{Y}'_i)$.

As mentioned in Section 3.2.1, group sparsity could be applied to this problem in terms of the naturally formed groups of features. According to Support Kernel Machines (SKMs) [33], Problem (3.2) can be readily formulated with group feature representation $\Phi(\mathcal{X}_i, \mathcal{Y}_i) = [\Phi_1(\mathcal{X}_i, \mathcal{Y}_i); \dots; \Phi_m(\mathcal{X}_i, \mathcal{Y}_i)]$ as,

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \Omega(\mathbf{w}) + \frac{C}{n} \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i, \forall \mathcal{Y}'_i \in \mathbb{Y} : \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i) \geq \Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \xi_i, \end{aligned} \quad (3.3)$$

where $\mathbf{w} = [\mathbf{w}_1; \dots; \mathbf{w}_m]$, and the regularizer $\Omega(\mathbf{w})$ can be defined as $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\| \right)^2$ for SKM [33], $\Omega(\mathbf{w}) = \frac{1}{2} \sum_{j=1}^m \|\mathbf{w}_j\|$ for group lasso [97], $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\|^p \right)^{2/p}$ or $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\|^p \right)^{2/p} + \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\| \right)^2$ where $p \geq 1$ for generalized MKL [44]. The number of constraints in Problem (3.3) depends on the specific structure in the space of $\mathbb{Y}$. In this Chapter, the main focus is on sequence or tree structure which usually induces exponential number of constraints. Therefore, the general SKMs, group lasso algorithms, as well as the state-of-the-art MKLs, cannot be applied here. In what follows, an efficient cutting plane algorithm is proposed to solve Problem (3.3).

3.2.3 Multiple Kernel Learning by Cutting Plane Method

Problem (3.3) has exponential number of constraints as well as non-differentiable mixed norm regularizer $\Omega(\mathbf{w})$ and thus, the optimization is very difficult. The following theorem states that Problem (3.3) can be equivalently formulated as 1-slack formulation [14].

Theorem 1. *Problem (3.3) is equivalent to the problem*

$$\begin{aligned}
 \min_{\mathbf{w}, \xi \geq 0} \quad & \Omega(\mathbf{w}) + C\xi \\
 \text{s.t.} \quad & \forall [\mathcal{Y}'_1, \dots, \mathcal{Y}'_n] \in \mathbb{Y}^n : \\
 & \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i) \geq \frac{1}{n} \sum_{i=1}^n \Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \xi.
 \end{aligned} \tag{3.4}$$

Proof: The empirical risk can be derived as follows,

$$\begin{aligned}
 \frac{1}{n} \sum_{i=1}^n \xi_i &= \frac{1}{n} \sum_{i=1}^n \max_{\mathcal{Y}'_i \in \mathbb{Y}} \left[\Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i) \right] \\
 &= \max_{[\mathcal{Y}'_1, \dots, \mathcal{Y}'_n] \in \mathbb{Y}^n} \frac{1}{n} \sum_{i=1}^n \left[\Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i) \right] = \xi
 \end{aligned}$$

where the second equality holds since for any given $\mathbf{w}$ each $\mathcal{Y}'_i$ can be optimized independently due to the linearly decomposed property in $\mathcal{Y}'_i$ [14]. This completes the proof. $\square$

Hereby, Algorithm 1 is proposed to solve Problem (3.4). This algorithm constructs a working set $\mathcal{W}$ iteratively. In each iteration, the most violated constraint is found, and then added into the working set $\mathcal{W}$. After that, a subproblem is solved over $\mathcal{W}$ in order to obtain a new solution $\mathbf{w}$.

Assume that the most violated outputs $(\hat{\mathcal{Y}}_1, \dots, \hat{\mathcal{Y}}_n)$ can be accessed given $\mathbf{w}$. After s iterations, one can construct a set of most violated labels $\mathcal{W}^s$, the subproblem in Algorithm 1 is formulated as

$$\begin{aligned}
 \min_{\mathbf{w}, \xi \geq 0} \quad & \Omega(\mathbf{w}) + C\xi \\
 \text{s.t.} \quad & \xi \geq q^r + \sum_{j=1}^m \mathbf{w}_j^T \mathbf{p}_j^r, \forall r = 1, \dots, s
 \end{aligned} \tag{3.5}$$

where $\mathbf{p}_j^r = -\frac{1}{n} \sum_{i=1}^n \delta \Phi_j^i(\hat{\mathcal{Y}}_i^r)$ and $q^r = \frac{1}{n} \sum_{i=1}^n \Delta(\mathcal{Y}_i, \hat{\mathcal{Y}}_i^r)$. Note that subproblem (3.5) has s constraints. For the sake of simplicity, one uses $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\| \right)^2$ as the showcase in this thesis¹. The following Proposition gives the detailed derivation to obtain its conic dual problem.

¹The dual of other regularizers can be found in [44].

Proposition 1. *By assuming $\Omega(\mathbf{w}) = \frac{1}{2}(\sum_{j=1}^m \|\mathbf{w}_j\|)^2$, the conic dual [33] of Problem (3.5) is*

$$\begin{aligned} \max_{\alpha \in \mathcal{A}_s} \max_{\theta} \quad & -\theta + \sum_{r=1}^s \alpha_r q^r \\ \text{s.t.} \quad & \frac{1}{2} \alpha^T Q^j \alpha \leq \theta, \forall j = 1, \dots, m \end{aligned} \quad (3.6)$$

where $\mathcal{A}_s = \{\sum_{r=1}^s \alpha_r \leq C, \alpha_r \geq 0, \forall r = 1, \dots, s\}$, and $Q_{r,r'}^j = \langle \mathbf{p}_j^r, \mathbf{p}_j^{r'} \rangle$.

Proof: Without loss of generality, denote $\mathbf{w} = [\mathbf{w}_1; \dots; \mathbf{w}_m]$ and $\mathbf{p}^r = [\mathbf{p}_1^r; \dots; \mathbf{p}_m^r]$. Problem (3.5) becomes

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\| \right)^2 + C\xi \\ \text{s.t.} \quad & \xi \geq q^r + \mathbf{w}^T \mathbf{p}^r, \forall r = 1, \dots, s \end{aligned}$$

By introducing a new variable $u \in \mathbb{R}$ and moving out summation operator from objective to be a constraint, one can obtain the equivalent optimization problem as

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} u^2 + C\xi \\ \text{s.t.} \quad & \xi \geq q^r + \mathbf{w}^T \mathbf{p}^r, \forall r = 1, \dots, s \\ & \sum_{j=1}^m \|\mathbf{w}_j\| \leq u. \end{aligned}$$

One can further simplify the above problem by introducing another variables $\rho \in \mathbb{R}^m$ such that $\|\mathbf{w}_j\| \leq \rho_j, \forall j = 1, \dots, m+1$ to be

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} u^2 + C\xi \\ \text{s.t.} \quad & \xi \geq q^r + \mathbf{w}^T \mathbf{p}^r, \forall r = 1, \dots, s \\ & \sum_{j=1}^m \rho_j \leq u, \|\mathbf{w}_j\| \leq \rho_j, \forall j = 1, \dots, m. \end{aligned}$$

It is well known that $\|\mathbf{w}_j\| \leq \rho_j$ is a second-order cone constraint. Following the recipe of [29], the self-dual cone $\|\mathbf{v}_j\|_2 \leq \eta_j, \forall j = 1, \dots, m$ can be introduced to form the Lagrangian function: $\mathcal{L}(\mathbf{w}, \xi, u, \rho; \alpha, \tau, \gamma, \mathbf{v}, \eta) = \frac{1}{2} u^2 + C\xi - \sum_{r=1}^s \alpha_r (\xi - q^r - \mathbf{w}^T \mathbf{p}^r) -$

$\tau\xi + \gamma(\sum_{j=1}^m \rho_j - u) - \sum_{j=1}^m (\langle \mathbf{v}_j, \mathbf{w}_j \rangle + \eta_j \rho_j)$, with dual variables $\alpha_r \in \mathbb{R}_+$, $\tau \in \mathbb{R}_+$, $\gamma \in \mathbb{R}_+$. The derivatives of the Lagrangian with respect to the primal variables have to vanish which leads to the following KKT conditions: $\mathbf{v}_j = \sum_{r=1}^s \alpha_r \mathbf{p}_j^r, \forall j = 1, \dots, m$, $C - \sum_{r=1}^s \alpha_r - \tau = 0$, $u = \gamma$, and $\gamma = \eta_j, \forall j = 1, \dots, m$. By substituting all the primal variables with dual variables by above KKT conditions, one can obtain the following dual problem,

$$\begin{aligned} \max_{\alpha, \gamma} \quad & -\frac{1}{2}\gamma^2 + \sum_{r=1}^s \alpha_r q^r \\ \text{s.t.} \quad & \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\| \leq \gamma, \forall j = 1, \dots, m \\ & \sum_{r=1}^s \alpha_r \leq C, \alpha_r \geq 0, \forall r = 1, \dots, m \end{aligned}$$

By setting $\theta = \frac{1}{2}\gamma^2$ and $\mathcal{A}_s = \{\sum_{r=1}^s \alpha_r \leq C, \alpha_r \geq 0, \forall r = 1, \dots, s\}$, one can reformulate the above problem as

$$\max_{\theta, \alpha \in \mathcal{A}_s} -\theta + \sum_{r=1}^s \alpha_r q^r : \frac{1}{2}\alpha^T Q^j \alpha \leq \theta, \forall j = 1, \dots, m$$

where $Q_{r,r'}^j = \langle \mathbf{p}_j^r, \mathbf{p}_j^{r'} \rangle$. According to the property of self-dual cone, one can obtain the primal solution from its dual as $\mathbf{w}_j = -\mu_j \mathbf{v}_j = -\mu_j \sum_{r=1}^s \alpha_r \mathbf{p}_j^r$ where μ_j is the dual variable of the j^{th} quadratic constraint such that $\sum_{j=1}^m \mu_j = 1, \mu_j \in \mathbb{R}_+, \forall j = 1, \dots, m$ [33]. The proof is completed. $\square$

Problem (3.6) is in form of a quadratically constrained quadratic programming (QCQP) problem, which is similar to the multiple kernel learning problem with a small size of constraints. Furthermore, the primal solutions can be recovered by

$$\mathbf{w}_j = -\mu_j \sum_{r=1}^s \alpha_r \mathbf{p}_j^r, \forall j = 1, \dots, m, \quad (3.7)$$

where μ is the Lagrangian multipliers, each of which corresponds to one constraint in (3.6). Together with the sparsity of α and the sparsity of μ , the primal variable $\mathbf{w}$ is sparse and so the recovery of $\mathbf{w}$ is also fast. The algorithm stops if no constraint is found with the desired precision or the maximum number of iterations is reached.

Algorithm 1 Multiple Template Learning

- 1: **Input:** $\mathcal{D} = \{(\mathcal{X}_1, \mathcal{Y}_1), \dots, (\mathcal{X}_n, \mathcal{Y}_n)\}, C, \epsilon$
 - 2: $\mathbf{w} = \mathbf{0}, \mathcal{W} = \emptyset,$
 - 3: **repeat**
 - 4: Find the most violated $(\hat{\mathcal{Y}}_1, \dots, \hat{\mathcal{Y}}_n)$
 - 5: $\mathcal{W} := \mathcal{W} \cup \{(\hat{\mathcal{Y}}_1, \dots, \hat{\mathcal{Y}}_n)\}$
 - 6: Obtain $\mathbf{w}$ by solving subproblem (3.6) over $\mathcal{W}$ according to (3.7)
 - 7: **until** ϵ -optimal
 - 8: **return** $\mathbf{w}$
-

As will be shown in Section 3.2.4, when $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\| \right)^2$, Algorithm 1 can converge to an ϵ -optimality in a finite number of iterations. Therefore, the QCQP problem in (3.6) with $s + 1$ variables and $m + 1$ constraints can be solved efficiently by a QCQP toolbox, such as Mosek [143]. Since Mosek simultaneously solves the primal and its dual form, the weights μ for each group of features can be obtained at the same time. Alternatively, one can apply other efficient MKL algorithms [34–36] to solve (3.6) when a sophisticated QCQP solver is not available, but the solution may not be as accurate as that of Mosek.

It is worth mentioning that the proposed method obtains $\mathbf{w}_j = -\mu_j \sum_{r=1}^s \alpha_r \mathbf{p}_j^r, \forall j = 1, \dots, m$; while Structural SVMs do not consider the weights μ , *ie.* a uniform weighting for templates. Another good property is that, when $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\| \right)^2$, the sparsity of $\text{MTL}^{\text{struct}}$ is not only on α but also on μ due to the feasible domain $\mathcal{A}_s$ and quadratic constraints in Problem (3.6), but the cutting plane algorithm for the uniform weighting does not enforce the sparsity on the feature groups.

Finding the most violated constraint with outputs $(\hat{\mathcal{Y}}_1, \dots, \hat{\mathcal{Y}}_n)$ is to solve the following problem

$$\arg \max_{(\mathcal{Y}'_1, \dots, \mathcal{Y}'_n) \in \mathbb{Y}^n} \sum_{i=1}^n \Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \sum_{i=1}^n \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i), \quad (3.8)$$

where the definition of cost $\Delta(\mathcal{Y}_i, \mathcal{Y}'_i)$ and the feature mapping Φ depend on the specific tasks. The input-output pairs in $\mathcal{D}$ are generally considered i.i.d., so Problem (3.8) can be generally decomposed into n independent optimization problems as

$$\hat{\mathcal{Y}}_i = \arg \max_{\mathcal{Y}'_i \in \mathbb{Y}} \Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i), \forall i = 1, \dots, n. \quad (3.9)$$

In the following subsections, I present the convergence analysis and complexity analysis of Algorithm 1. After that, I deal with specific tasks, namely sequence labeling and dependency parsing. Once the parameter $\mathbf{w}$ is learned, one can do prediction by solving an inference problem in (3.1) given an input $\mathcal{X} \in \mathbb{X}$.

3.2.4 Convergence & Computational Complexity

The termination condition of Algorithm 1 is defined as $R_{emp}(\mathbf{w}_s) - R_s(\mathbf{w}_s) < \epsilon$ where the risk of upper bound and lower bound of Problem (3.4) are

$$\begin{aligned} R_{emp}(\mathbf{w}) &= \frac{1}{n} \sum_{i=1}^n \max_{\mathcal{Y}'_i \in \mathbb{Y}} \left(\Delta(\mathcal{Y}_i, \mathcal{Y}'_i) - \sum_{j=1}^m \mathbf{w}_j^T \delta \Phi_j^i(\mathcal{Y}'_i) \right), \\ R_s(\mathbf{w}) &= \max_{r=1, \dots, s} \left(\sum_{j=1}^m \mathbf{w}_j^T \mathbf{p}_j^r + q^r \right). \end{aligned}$$

The maximum number of iterations could also be used to terminate the algorithm considering the time and space limitations.

Notice that the convergence proof for bundle method or cutting plane algorithm in [85, 144] does not apply in this case as the Fenchel dual of mixed norm fails to satisfy the strong convexity assumption if $m > 1$. As $m = 1$, Algorithm 1 is exactly the bundle method [85]. When $m > 1$, Theorem 2 shows the convergence of Algorithm 1 which can converge to ϵ -optimality in a finite number of iterations.

Theorem 2. *For any $0 < C, 0 < \epsilon \leq 4R^2C$ and any training example $(\mathcal{X}_1, \mathcal{Y}_1), \dots, (\mathcal{X}_n, \mathcal{Y}_n)$, Algorithm 1 converges to the desired precision ϵ after at most,*

$$\left\lceil \log_2 \left(\frac{\Delta}{4R^2C} \right) \right\rceil + \left\lceil \frac{16R^2C}{\epsilon} \right\rceil$$

iterations, where $R^2 = \max_{i, \mathcal{Y}'} \|\Phi(\mathcal{X}_i, \mathcal{Y}_i) - \Phi(\mathcal{X}_i, \mathcal{Y}'_i)\|^2$, $\Delta = \max_{i, \mathcal{Y}'_i} \Delta(\mathcal{Y}'_i, \mathcal{Y}_i)$ and $\lceil \cdot \rceil$ is the integer ceiling function.

Proof: Since there are m quadratic constraints, the dual objective of MTL^{struct} can be reformulated as

$$\max_{\alpha \in \mathcal{A}_s} \min_{j=1, \dots, m} \Theta_d(\alpha) = -\frac{1}{2} \sum_{r=1}^s \sum_{r'=1}^s \alpha_r \alpha_{r'} Q_{r, r'}^j + \sum_{r=1}^s \alpha_r q^r.$$

One considers each group of features at one time:

$$\max_{\alpha \in \mathcal{A}_s} \Theta_j(\alpha),$$

where Q^j is positive semi-definite matrix, and derivative $\partial\Theta_j(\alpha) = q^j - Q^j\alpha$. The Lemma 2 in [14] states that a line search starting at α and along an ascent direction η with maximum step-size $C > 0$ improves the objective by at least $\max_{0 \leq \beta \leq C} \{\Theta_j(\alpha + \beta\eta) - \Theta_j(\alpha)\} \geq \frac{1}{2} \min \left\{ C, \frac{\partial\Theta_j(\alpha)^T \eta}{\eta^T Q^j \eta} \right\} \partial\Theta_j(\alpha)^T \eta$. If one considers subgradient descent method, the line search along the subgradient of objective is $\partial\Theta_{j^*}(\alpha)$ where $j^* = \arg \min_{j \in \{1, \dots, m\}} \Theta_j(\alpha)$. Therefore, the maximum improvement is

$$\begin{aligned} & \max_{0 \leq \beta \leq C} \{\Theta_{j^*}(\alpha + \beta\eta) - \Theta_{j^*}(\alpha)\} \\ & \geq \frac{1}{2} \min \left\{ C, \frac{\partial\Theta_{j^*}(\alpha)^T \eta}{\eta^T Q^{j^*} \eta} \right\} \partial\Theta_{j^*}(\alpha)^T \eta \\ & \geq \frac{1}{2} \min_{j \in \{1, \dots, m\}} \left\{ C, \frac{\partial\Theta_j(\alpha)^T \eta}{\eta^T Q^j \eta} \right\} \partial\Theta_j(\alpha)^T \eta. \end{aligned} \quad (3.10)$$

One can observe that it is a special case of [14] if there is only one template. According to Theorem 5 in [14], for a newly added constraint $\hat{\mathcal{Y}} \in \mathcal{W}$ and some $\gamma_j > 0$, one can obtain $\partial\Theta_j(\alpha)^T \eta = \gamma_j \geq \epsilon$ by setting the ascent direction $\eta_{\hat{\mathcal{Y}}} = 1$ for the newly added $\hat{\mathcal{Y}}$ and $\eta_r = -\frac{1}{C}\alpha_r$ for the others. Here, let $\gamma = \min_{j \in \{1, \dots, m\}} \gamma_j$ so as to be the lower bound of $\partial\Theta_j(\alpha)^T \eta, \forall j = 1, \dots, m$. In addition, the upper bound for $\eta^T Q^j \eta \leq 4R^2, \forall j = 1, \dots, m+1$ can also be obtained by the fact that $\eta^T Q^j \eta = Q_{\hat{\mathcal{Y}}, \hat{\mathcal{Y}}}^j - \frac{2}{C} \sum_{r=1}^s \alpha_r Q_{r, \hat{\mathcal{Y}}}^j + \frac{1}{C^2} \sum_{r=1}^s \sum_{r'=1}^s \alpha_r \alpha_{r'} Q_{r, r'}^j \leq R^2 + \frac{2}{C} C R^2 + \frac{1}{C^2} C^2 R^2 = 4R^2, \forall j = 1, \dots, m$. By substituting them back to (5.18), the similar result shows the increase of the objective is at least

$$\min \left\{ \frac{C\gamma}{2}, \frac{\gamma^2}{8R^2} \right\}.$$

Let $\Theta(\alpha) = \min_{j=1, \dots, m} Q_d(\alpha)$. The initial optimality gap is at most $C\Delta$ when $\mathbf{w} = \mathbf{0}$. This means that the initial optimality gap $\delta(0) = \Theta(\alpha^*) - \Theta(\alpha^{(0)}) \leq C\Delta$ where α^* is the optimal dual solution. At the i^{th} iteration, the optimality gap $\delta(i) = \Theta(\alpha^*) - \Theta(\alpha^{(i)})$. One can have the following inequalities:

$$\begin{aligned} \delta(i) - \delta(i+1) &= \Theta(\alpha^*) - \Theta(\alpha^{(i)}) - (\Theta(\alpha^*) - \Theta(\alpha^{(i+1)})) \\ &= \Theta(\alpha^{(i+1)}) - \Theta(\alpha^{(i)}) \\ &\geq \min \left\{ \frac{C\gamma}{2}, \frac{\gamma^2}{8R^2} \right\}. \end{aligned}$$

According to the termination condition at the i^{th} iteration, the most violated constraint can only be found when $R_{emp}(\mathbf{w}^{(i)}) - R_i(\mathbf{w}^{(i)}) = \frac{1}{C}\delta(i) \leq \epsilon \leq \gamma[14]$. The optimality

gap $\delta(i)$ makes sure that there exists a constraint violated by at least $\gamma \geq \frac{\delta(i)}{C}$. If $\delta(i) \geq 4C^2R^2$, then $\gamma \geq 4CR^2$, so $\frac{C\gamma}{2} \leq \frac{\gamma^2}{8R^2}$. Then, one can further obtain

$$\delta(i) - \delta(i+1) \geq \frac{C\gamma}{2} \geq \frac{\delta(i)}{2}, \forall i.$$

At the i^{th} iteration, one can obtain the bounds for the the optimality gap

$$4C^2R^2 \leq \delta(i) \leq \frac{1}{2^i}\delta(0) \leq \frac{1}{2^i}C\Delta$$

In this case, the algorithm needs at most

$$i_1 \leq \left\lceil \log_2 \left(\frac{\Delta}{4R^2C} \right) \right\rceil$$

iterations until it has reached $\delta(i_1) \leq 4R^2C^2$ which means that $\frac{C\gamma}{2} \geq \frac{\gamma^2}{8R^2}$. If $\delta(i_1) \leq 4R^2C^2$, the optimality gap is bounded by

$$\delta(i) - \delta(i+1) \geq \frac{\gamma^2}{8R^2} \geq \frac{\delta(i)^2}{8R^2C^2}, \forall i.$$

By solving the differential equation $\frac{\partial \delta(i)}{\partial i} = -\frac{\gamma^2}{8R^2} \geq \frac{\delta(i)^2}{8R^2C^2}$ with the new starting boundary condition $\delta(0) = 4R^2C^2$. The solution is $\delta(i) \leq \frac{8R^2C^2}{i+2}$ as presented in [144]. The algorithm does not need more than

$$i_2 \leq \left\lceil \frac{8R^2C}{\epsilon} \right\rceil$$

iterations to reach $\delta(i) \leq C\epsilon$. As the optimality gap reaches ϵC , there is no guarantee of finding the most violated constraints, but they may still exist. Since each constraint leads to an increase at least $\frac{\delta(i)^2}{8R^2C^2} = \frac{(\epsilon C)^2}{8R^2C^2} = \frac{\epsilon^2}{8R^2}$ in the dual objective, there are

$$i_3 \leq \left\lceil \frac{8R^2C}{\epsilon} \right\rceil$$

iterations for adding this constraints before the gap becomes negative. By summing all iterations, one can obtain the overall bound. The proof is completed. $\square$

For the following applications in this chapter, the linear kernel is only used in the proposed algorithm. According to [32], given a QCQP problem with s variables and m quadratic constraints, the optimization tools such as Mosek can yield the worst-case complexity of $O(ms^3)$ by using interior-point method. At the s^{th} iteration, Algorithm

1 takes at most $O(ms^3)$ time for solving the reduced QCQP problem (3.6), $O(s^2)$ for computing ξ , $O(s|\kappa(\mathcal{D})|)$ for $\mathbf{w}$, $O(n)$ for q^s , $O(n|\kappa(\mathcal{D})|)$ for $\mathbf{p}^s$ and $O(s|\kappa_j(\mathcal{D})|)$ for adding a row/column to Q^j since Q^j can be computed incrementally, $\forall j = 1, \dots, m$. Another major training cost is to find the most violated constraint. For sequence labeling task, suppose the average length of sequences are l and the number of classes are c , then it takes $O(lc|\kappa(\mathcal{D})|)$ time to find the most violated constraint. Similarly, for dependency parsing task, it takes $O(l^3|\kappa(\mathcal{D})|)$ or $O(l^2|\kappa(\mathcal{D})|)$ time for projected and non-projected dependency tree structure, respectively. Hence, the overall time complexity is $O(ms^3 + s|\kappa(\mathcal{D})| + nu|\kappa(\mathcal{D})|)$, where $u = \{lc, l^2, l^3\}$ for different inference problems.

For sparse and high dimensional data, Algorithm 1 can efficiently exploit the sparsity of the feature vectors, so that computation for the elements in Q only depends on the number of non-zeros z . The overall time complexity is reduced to $O(ms^3 + sz + nuz)$. According to Theorem 2, the algorithm converges to the ϵ -optimal solution in $O(\frac{C}{\epsilon})$. And empirical observations show that several hundreds of iterations are enough for good performance. Moreover, the cut pruning strategy is also used in the implementation, which has been used in Structural SVM [14], to speed up the learning process. So it is faster than the analysis in practice. Compared with Structural SVM with l_2 norm, except that Structural SVM needs to solve a QP problem (takes $O(s^3)$ time) instead of solving a QCQP problem. The complexity for other parts is the same. So the overall time complexity of the proposed MTL^{struct} is very comparable with Structural SVM.

OnlineMKL [18] attempts to solve the similar problem by online learning where two operations are needed at each iteration: the gradient descent step and proximal step. By exploiting the sparsity of features, the gradient step takes $O(z)$ time to update $\mathbf{w}$, $O(uz)$ to find the most violated configuration for the randomly sampled instance, and $O(mz)$ for incremental update of $\|\mathbf{w}_j\|, \forall j = 1, \dots, m$. However, the proximal step needs to scale $\mathbf{w}$ according to these norms, so it has to take $O(|\kappa(\mathcal{D})|)$ time where it is hard to exploit the sparsity of $\mathbf{w}$. This is different from Algorithm 1 with both sparsity on α and μ . The overall time complexity for each epoch is $O(n(uz + mz + |\kappa(\mathcal{D})|))$ in OnlineMKL. Generally, it takes tens of epochs to gain a good performance. Hence, for high dimension but very sparse data, i.e. more than 10 million, $O(n|\kappa(\mathcal{D})|)$ will dominate the time complexity for OnlineMKL, while Algorithm 1 is only dominated by the inference

cost ($O(nuz)$). Taking the sequence labeling task for example, suppose 20 epochs for OnlineMKL, and 500 iterations for Algorithm 1 used in the following experiments, the time complexity of OnlineMKL is $O(20n|\kappa(\mathcal{D})|)$ while that of Algorithm 1 is $O(500nlcz)$ only. OnlineMKL is slower than Algorithm 1 since $|\kappa(\mathcal{D})| \gg 25lcz$ where l , c and z are usually very small (around tens). In the following applications, I will empirically demonstrate OnlineMKL is quite slow.

3.3 Extension to p -Block-Norm Regularization

The generalized form of mixed norm regularizer $\Omega(\mathbf{w}) = \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\|^p \right)^{2/p}$ with $p > 1$ [44] is considered. According to Theorem 1 and (3.5), one obtains the following subproblem

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} \left(\sum_{j=1}^m \|\mathbf{w}_j\|^p \right)^{2/p} + C\xi \\ \text{s.t.} \quad & \xi \geq q^r + \sum_{j=1}^m \mathbf{w}_j^T \mathbf{p}_j^r, \forall r = 1, \dots, s. \end{aligned} \quad (3.11)$$

after s iterations in Algorithm 1. The following proposition gives the dual problem of (3.11).

Proposition 2. *The dual problem of (3.11) with $p > 1$ is*

$$\max_{\alpha \in \mathcal{A}_s} -\frac{1}{2} \left(\sum_{j=1}^m \left(\sqrt{\alpha^T Q^j \alpha} \right)^{p^*} \right)^{2/p^*} + \sum_{r=1}^s \alpha_r q^r, \quad (3.12)$$

where $p^* = \frac{p}{p-1}$. The weights of kernels are recovered as

$$\mu_j = \left(\sum_{j=1}^m \left(\sqrt{\alpha^T Q^j \alpha} \right)^{\frac{p}{p-1}} \right)^{\frac{p-2}{p}} \left(\sqrt{\alpha^T Q^j \alpha} \right)^{\frac{2-p}{p-1}}, \quad (3.13)$$

$\forall j = 1, \dots, m$, and primal variables can be recovered by

$$\mathbf{w}_j = -\mu_j \sum_{r=1}^s \alpha_r \mathbf{p}_j^r, \forall j = 1, \dots, m. \quad (3.14)$$

Proof: The Lagrangian dual problem of (3.11) is formulated with dual variables $\alpha_r \geq 0, \forall r = 1, \dots, s$, and $\tau \geq 0$ as

$$\max_{\alpha \geq 0} \min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \|\mathbf{w}\|_{2,p}^2 + C\xi - \sum_{r=1}^s \alpha_r (\xi - q^r - \sum_{j=1}^m \mathbf{w}_j^T \mathbf{p}_j^r) - \tau \xi$$

where $\|\mathbf{w}\|_{2,p}^2 = (\sum_{j=1}^m \|\mathbf{w}_j\|^p)^{2/p}$. One can obtain the following KKT conditions

$$\begin{aligned} \mathbf{w}_j &= -\|\mathbf{w}\|_{2,p}^{p-2} \|\mathbf{w}_j\|^{2-p} \sum_{r=1}^s \alpha_r \mathbf{p}_j^r, \forall j = 1, \dots, m \\ \sum_{r=1}^s \alpha_r &\leq C, \alpha_r \geq 0, \forall r = 1, \dots, s. \end{aligned} \quad (3.15)$$

According to the Fenchel-Legendre conjugate function [44], one has the following equality

$$\max_{\mathbf{w}} -\frac{1}{2} \|\mathbf{w}\|_{2,p}^2 + \sum_{j=1}^m \mathbf{w}_j^T \left(-\sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right) = \frac{1}{2} \left\| -\sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|_{2,p^*}^2$$

such that $\frac{1}{p^*} + \frac{1}{p} = 1$. Substituting them back to the Lagrangian function, one can reformulate the dual problem as

$$\max_{\alpha \in \mathcal{A}_s} -\frac{1}{2} \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|_{2,p^*}^2 + \sum_{r=1}^s \alpha_r q^r$$

which is equivalent to the Problem (3.12).

According to (3.15), one can obtain the following equation

$$\|\mathbf{w}_j\| = \|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|^{\frac{1}{p-1}}, \forall j = 1, \dots, m.$$

By further substituting it back to (3.15), one can obtain the primal solution

$$\mathbf{w}_j = -\|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|^{\frac{2-p}{p-1}} \sum_{r=1}^s \alpha_r \mathbf{p}_j^r, \forall j = 1, \dots, m.$$

Let $\mu_j = \|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|^{\frac{2-p}{p-1}}$, $\forall j = 1, \dots, m$, so the primal solution can be simplified as $\mathbf{w}_j = -\mu_j \sum_{r=1}^s \alpha_r \mathbf{p}_j^r$. According to the definition of the mixed norm, one can have the following derivations,

$$\begin{aligned} \|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} &= \left(\sum_{j=1}^m \|\mathbf{w}_j\|^p \right)^{\frac{p-2}{p(p-1)}} \\ &= \left(\sum_{j=1}^m \left(\|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|^{\frac{1}{p-1}} \right)^p \right)^{\frac{p-2}{p(p-1)}} \\ &= \left(\|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} \right)^{\frac{p-2}{p-1}} \left(\sum_{j=1}^m \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|^{\frac{p}{p-1}} \right)^{\frac{p-2}{p(p-1)}}. \end{aligned}$$

Hence, one can obtain the representation of the mixed norm with respect to α as

$$\|\mathbf{w}\|_{2,p}^{\frac{p-2}{p-1}} = \left(\sum_{j=1}^m \left\| \sum_{r=1}^s \alpha_r \mathbf{p}_j^r \right\|^{\frac{p}{p-1}} \right)^{\frac{p-2}{p}}.$$

Substituting it to the μ and $\mathbf{w}$, one obtains the solutions in Proposition 2. The proof is complete. $\square$

Problem (3.12) is a smooth function with $p > 1$ and the feasible domain of α is in a simplex $\mathcal{A}_s$. One can employ projected gradient descent to solve (3.12) where the projection step can be readily solved by efficient projections onto simplex method [145]. Since the inference process is the same as Algorithm 1, the only difference is to solve the subproblem (3.12) instead of (3.6). Hence, Algorithm 1 can be easily extended to solve MTL^{struct} with p -block norm regularization.

3.4 Experiments

As aforementioned, templates are employed widely in various applications. To verify the proposed method, it is explored on three tasks: Chinese Word Segmentation [5, 146], Named Entity Recognition [8], and Dependency Parsing [10]. The first two tasks are popularly transformed into the structured prediction problem with the output as a sequence structure [5, 8], while dependency parsing is cast as the structured prediction problem with the output as a spanning tree [10]. In what follows, I will first give the general setting of these tasks, and then the task-specific settings. Their experimental results are shown separately.

3.4.1 Experimental Setting

In sequence learning, high-order dependence among labels and state-state transition with observation in the linear chain could be considered. But the state-state transition feature without observations are considered in this Chapter for two reasons. First, the inference problem (3.8) can be solved efficiently and effectively by the Viterbi algorithm [147] for Hamming loss [6]. Second, there exist toolboxes with state-of-the-art methods for the fair comparison. The methods for comparison are listed as follows:

- **CRF**(ℓ_2) (**CRF**(ℓ_1)) [9]: CRF with a Gaussian (Laplacian) prior defined by CRF++ toolbox² in terms of the command option “-a”.
- **IRW-CRF**: Solving weighted CRF(ℓ_2) by iterative re-weighted least square method with the update rule for weights as $(\|\mathbf{w}_j\|^2 + \epsilon)^{-1}$ [148] and warm start using **CRF**(ℓ_2) solution. To avoid numerical instabilities, $\epsilon = 0.001$ is fixed and the maximum iteration is set to 10.
- **SVM**^{hmm} [6]: Structural SVM for sequence learning by SVM^{hmm} toolbox³. The recommended termination condition “-e 0.5” is used in following experiments.
- **OnlineMKL** [18]: Online MKL for structured prediction with C++ implementation is provided by the authors. The number of training epochs is set to 20 following [18]⁴.
- **MTL**^{hmm}: The instantiated method of the proposed MTL^{struct} framework for sequence learning problem is implemented in C++. The termination condition is the same as **SVM**^{hmm}.

All above compared methods have the tradeoff parameter, so they are rescaled into the grid of $\{10^{-2}, 10^{-1}, 10^0, 10^1, 10^2\}$ such that the objectives of all methods are scaled with a constant. In the following experiments, C of MTL^{hmm} is tuned in the grid of $n \times \{10^{-2}, 10^{-1}, 10^0, 10^1, 10^2\}$, while the parameters of other methods are adjusted accordingly.

In dependency parsing, I follow the structured prediction framework with the loss defined as the number of words that have the incorrect parent [10]. According to [91], there are two algorithms which can solve Problem (3.8) in terms of two types of dependency trees, respectively. For projective dependency tree, Eisner algorithm [149] has a runtime of $O(l^3)$, while Chu-Liu-Edmonds [150] provides non-projective parsing complexity of $O(l^2)$ where l is the length of the input sentence. Non-projective case is adopted according to the property of datasets used in the following experiments. For fair comparison, the mainly focus is on comparing the following methods:

²<http://crfpp.sourceforge.net/>

³http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html

⁴In the following experiments, I will show that the large number of epochs is infeasible for the large scale high-dimensional data.

- **MSTParser** [10]: Online large margin method for dependency parsing with the available toolbox⁵. I vary the number of epochs in the set of $\{20, 30, 40\}$.
- **MTL^{parse}**: The instantiated method of the proposed MTL^{struct} framework for dependency parsing.

The setting of MTL^{parse} is similar to the task of sequence learning, but with a fixed $C=n \times 10^2$. Notice that more epochs do not affect the performance of MSTParser much.

The last experiments are conducted for the analysis of MTL^{struct} with p -block-norm regularization (p MTL^{struct}) in the setting of sequence labeling problems (p MTL^{hmm}). The same setting as that of MTL^{hmm} is adopted, as well as vary the parameter p in the set of $\{4/3, 2, 4, \infty\}$ [44] to compare with MTL^{hmm} such that I can obtain the influence of the p -block-norm regularization in the sequence labeling tasks.

For fair comparisons, all comparing methods have the same set of templates(features) as the input, and the best results of each method are reported on the split datasets by tuning the parameter in the given grid. In what follows, the detailed experiments will be shown individually.

Table 3.1: Benchmark datasets for Chinese word segmentation.

Data	# Sentence		#Token		#Features	
	Train	Test	Train	Test	TP1	TP2
AS	708,953	14,429	8,368,050	197,681	31,587,879	53,599,827
MSR	86,924	3,985	4,050,469	184,355	17,055,615	31,228,275
CityU	53,019	1,492	2,403,355	67,690	13,196,613	25,143,720
PKU	19,054	1,944	1,826,448	172,733	9,605,457	18,717,687

3.4.2 Chinese Word Segmentation

Chinese word segmentation has been an active area of research in computational linguistics for several decades. The second International Chinese Word Segmentation Bakeoff task⁶ provides the platform to evaluate different methods. It contains four corpora:

⁵<http://sourceforge.net/projects/mstparser/>

⁶<http://www.sighan.org/bakeoff2005/>

Table 3.2: Two sets of templates used for Chinese word segmentation task: TP1 and TP2. The symbol ‘-’ means that this template is not used in the setting.

Type	Index		Name	Template
	TP1	TP2		
uni-gram	1	1	U00	%x[-2,0]
	2	2	U01	%x[-1,0]
	3	3	U02	%x[0,0]
	4	4	U03	%x[1,0]
	5	5	U04	%x[2,0]
bi-gram	9	1	U08	%x[-1,0]/%x[0,0]
	10	10	U09	%x[0,0]/%x[1,0]
	-	11	U10	%x[-2,0]/%x[-1,0]
	-	12	U11	%x[1,0]/%x[2,0]
	-	13	U12	%x[-2,0]/%x[0,0]
	-	14	U13	%x[-1,0]/%x[1,0]
	-	15	U14	%x[0,0]/%x[2,0]
	-	16	U15	%x[-2,0]/%x[1,0]
	-	17	U16	%x[-1,0]/%x[2,0]
tri-gram	6	6	U05	%x[-2,0]/%x[-1,0]/%x[0,0]
	7	7	U06	%x[-1,0]/%x[0,0]/%x[1,0]
	8	8	U07	%x[0,0]/%x[1,0]/%x[2,0]
	11	18	B	state transition

Academia Sinica (AS), City University of Hong Kong (CityU), Peking University (PKU) and Microsoft Research (MSR). The detailed corpus information is referred to [151]. Table 3.1 shows the benchmark splits of data and their associated features in terms of templates in Table 3.2 following the definition of templates in CRF++. Results are reported by the following measures: test recall (Recall), test precision (Prec), balanced F score (F1), and recall on in-vocabulary words (R_{iv}). Three label notations are used such as B, I, E (B is the beginning character of a word; I is the inner character of a word; E is the end character of a word) to transform Chinese word segmentation task into a label sequence learning problem. To examine whether or not improper or redundant templates could affect sequence prediction performance, the set of templates are divided into two categories: TP1 and TP2. TP1 is frequently used in the literature [5]. In TP2, I deliberately enumerate all the possible bi-gram templates in the window size $[-2, +2]$.

Table 3.3: The performance (TP1/TP2) of Chinese word segmentation over five randomly sampled training and testing datasets. For each method, the first row is the mean and the second row is the corresponding standard deviation. The best results across the compared methods are shown in bold on TP1 and TP2, individually.

Dataset	Algorithm	Recall	Prec	F1	R_{iv}
PKU	CRF (ℓ_2)	96.4 / 96.0	96.7 / 96.0	96.6 / 96.0	97.2 / 96.8
		0.10 / 0.10	0.10 / 0.10	0.12 / 0.10	0.06 / 0.06
	CRF (ℓ_1)	95.7 / 95.5	95.9 / 95.3	95.8 / 95.4	96.4 / 96.3
		0.10 / 0.06	0.15 / 0.10	0.10 / 0.06	0.06 / 0.06
	IRW-CRF	97.3 / 95.9	96.5 / 95.8	96.4 / 95.8	97.1 / 96.7
		0.15 / 0.00	0.20 / 0.06	0.15 / 0.06	0.06 / 0.06
CityU	SVM ^{hmm}	96.6 / 96.5	96.7 / 96.5	96.6 / 96.5	97.4 / 97.2
		0.12 / 0.10	0.12 / 0.06	0.10 / 0.06	0.12 / 0.06
	OnlineMKL	95.9 / 96.2	95.9 / 96.3	95.9 / 96.3	96.6 / 97.0
		0.06 / 0.15	0.20 / 0.15	0.10 / 0.15	0.06 / 0.17
	MTL ^{hmm}	97.1 / 97.2	96.7 / 96.8	96.9 / 97.0	98.0 / 98.1
		0.10 / 0.10	0.10 / 0.06	0.10 / 0.10	0.06 / 0.00
CityU	CRF (ℓ_2)	96.3 / 95.9	96.6 / 95.8	96.4 / 95.9	96.9 / 96.6
		0.06 / 0.06	0.06 / 0.06	0.06 / 0.10	0.06 / 0.06
	CRF (ℓ_1)	97.0 / 96.5	97.2 / 96.5	97.1 / 96.5	97.6 / 97.1
		0.06 / 0.06	0.06 / 0.00	0.06 / 0.00	0.06 / 0.00
	IRW-CRF	97.0 / 96.4	97.2 / 96.4	97.1 / 96.4	97.6 / 97.0
		0.12 / 0.12	0.10 / 0.10	0.12 / 0.10	0.12 / 0.06
CityU	SVM ^{hmm}	97.0 / 96.9	97.1 / 96.8	97.0 / 96.8	97.6 / 97.4
		0.00 / 0.12	0.06 / 0.00	0.06 / 0.06	0.06 / 0.06
	MTL ^{hmm}	97.6 / 97.7	97.4 / 97.5	97.5 / 97.6	98.4 / 98.4
		0.06 / 0.00	0.00 / 0.00	0.06 / 0.00	0.06 / 0.06

This construction of templates has a high probability to include redundant information.

3.4.2.1 Model Stability and Training Time

I empirically study the sensitivity of performances and the time complexities of each algorithms on the two smallest datasets PKU and CityU. By merging the training and testing data in Table 3.1, I randomly generate 80% training and 20% testing splits five times. Results on PKU and CityU in Table 3.3 show that the standard deviations are less than 0.2%. The average CPU times on PKU dataset are shown in Figure 3.1. One

Table 3.4: The Chinese word segmentation results (TP1/TP2) of comparing methods on all the testing datasets. The best results across the compared methods are shown in bold on TP1 and TP2, individually.

Dataset	Method	Recall	Prec	F1	R_{iv}
AS	CRF (ℓ_2)	95.6 / 95.1	94.4 / 93.8	95.0 / 94.4	97.0 / 96.4
	CRF (ℓ_1)	95.4 / 95.0	94.1 / 93.4	94.8 / 94.2	96.7 / 96.4
	IRW-CRF	95.6 / 95.1	94.3 / 93.8	94.9 / 94.5	97.0 / 96.5
	SVM ^{hmm}	94.6 / 94.3	93.2 / 93.3	93.9 / 93.8	96.0 / 95.6
	MTL ^{hmm}	95.6 / 95.8	93.9 / 93.9	94.7 / 94.8	97.2 / 97.3
MSR	CRF (ℓ_2)	96.5 / 95.7	96.7 / 96.0	96.6 / 95.8	97.3 / 96.5
	CRF (ℓ_1)	96.2 / 95.7	96.2 / 95.6	96.2 / 95.6	96.9 / 96.5
	IRW-CRF	96.6 / 95.8	96.8 / 96.0	96.7 / 95.9	97.5 / 96.6
	SVM ^{hmm}	96.7 / 96.1	96.5 / 96.5	96.6 / 96.3	97.6 / 96.7
	MTL ^{hmm}	96.9 / 97.0	96.5 / 96.4	96.7 / 96.7	97.9 / 98.0
CityU	CRF (ℓ_2)	94.1 / 92.8	94.3 / 92.6	94.2 / 92.7	96.3 / 95.1
	CRF (ℓ_1)	93.4 / 92.9	93.6 / 92.9	93.5 / 92.9	95.6 / 95.1
	IRW-CRF	94.1 / 93.5	94.3 / 93.3	94.2 / 93.4	96.3 / 95.8
	SVM ^{hmm}	94.3 / 94.1	94.1 / 93.5	94.2 / 93.8	96.4 / 96.4
	MTL ^{hmm}	95.2 / 95.2	94.2 / 94.3	94.7 / 94.8	97.6 / 97.6
PKU	CRF (ℓ_2)	92.7 / 92.2	94.2 / 93.3	93.4 / 92.7	94.7 / 94.2
	CRF (ℓ_1)	91.6 / 91.8	93.1 / 92.4	92.4 / 91.7	93.8 / 93.4
	IRW-CRF	92.7 / 92.4	94.1 / 93.3	93.4 / 92.9	94.8 / 94.3
	SVM ^{hmm}	92.6 / 93.0	93.7 / 93.4	93.2 / 93.2	94.9 / 95.0
	OnlineMKL	89.9 / 91.0	89.2 / 90.5	89.6 / 90.7	91.7 / 92.7
	MTL ^{hmm}	93.8 / 94.0	93.3 / 93.6	93.6 / 93.8	96.1 / 96.2

can observe that the proposed method MTL^{hmm} is slower than CRFs and SVM^{hmm} in the range of large C values, but it is more than 10 times faster than OnlineMKL, and comparable to IRW-CRF. This is consistent with the complexity analysis in Section 4.4.7. Moreover, the experiments of OnlineMKL cannot be finished in the reasonable time for the other three large scale datasets.

3.4.2.2 Performance Analysis

The testing results on the standard split of the datasets are reported in Table 3.4. In TP1, except OnlineMKL, the rest four algorithms have similar F1 score with a difference

Figure 3.1: The average CPU time on PKU dataset by varying C on five randomly sample of training and testing datasets.

Figure 3.2: The performance and CPU time of OnlineMKL on PKU dataset with benchmark split and $C = 1$ by varying the number of epochs.

at most 0.5%, but CRF(ℓ_2) is 1% higher than SVM^{hmm} on AS. IRW-CRF slightly outperforms CRF(ℓ_2) in most cases. However, MTL^{hmm} always demonstrates the highest R_{iv} , which means weighting strategy is more useful to predict the known words correctly. Although MTL^{hmm} and SVM^{hmm} use the same set of features, MTL^{hmm} is better than SVM^{hmm} according to F1 score. This shows that the proposed weighting template strategy is helpful to boost the performance. However, OnlineMKL performs the worst among all algorithms. As 20 epoches are used in this setting as suggested in [18], the varied number of epochs are also studied in this Section. According to the complexity analysis in Section 3.2.4, OnlineMKL is extremely time-consuming in this task with empirical results shown in Figure 3.1. The varied number of epochs in the grid $\{5, 10, 15, 20, 30, 50, 70\}$

Figure 3.3: The learned template weights for Chinese word segmentation in TP1 and TP2 on PKU.

are studied. The experimental results of OnlineMKL on PKU dataset with $C = 1$ are reported in Figure 3.2. As observed, the performance of OnlineMKL only changes slightly with less than 1% when the number of epochs increases from 20 to 70, but the training time increases dramatically. In TP2, the similar phenomena could be observed, but the difference of F1 score among different methods are enlarged. For example, for CityU dataset, MTL^{hmm} is 2.1% higher than CRFs. MTL^{hmm} consistently shows the best F1 score.

By comparing the results between TP1 and TP2, one can find that adding more templates, such as templates U_{10} - U_{16} , degrades the performance of CRFs and SVM^{hmm} , but MTL^{hmm} and OnlineMKL obtain slight improvements. Therefore, the performance of weighting strategy in MTL^{hmm} and OnlineMKL is not greatly affected by the arbitrarily added templates. However, $CRF(\ell_2)$ and SVM^{hmm} do not consider this information, and $CRF(\ell_1)$ works poorly on CityU and PKU. Moreover, OnlineMKL demonstrates worse performance than MTL^{hmm} by more than 3% with respect to F1 on PKU. This is due to the fact that MTL^{hmm} can effectively identify the important templates from a given set of templates, and can remove unimportant templates simultaneously; while OnlineMKL may not be so effective in this case.

3.4.2.3 Weighting on Templates

The above statements can be justified by the learned template weights. Figure 3.3 shows an example of the learned weights of templates on PKU in the setting of TP1 and TP2.

One can observe that the weights learned by MTL^{hmm} has a higher standard deviation than that of OnlineMKL. The small standard deviation means the weights are more likely to be average weights. Hence, MTL^{hmm} prefers sparser weights than OnlineMKL. Even using more epochs, OnlineMKL still cannot improve the sparsity of weights or prediction performance. On the contrary, it takes more time for training. Together with the better performance, this implies that MTL^{hmm} trained in batch mode is more effective than OnlineMKL in online mode in this task.

Figure 3.4: F1 measure by varying C in TP2 on the Chinese word segmentation datasets.

Table 3.5: The overall performance of the comparing methods on Spanish and Dutch for named entity recognition. The best results across different methods are shown in bold.

Method	Validation set			Test set		
	Prec	Recall	F1	Prec	Recall	F1
CRF (ℓ_2)	74.38	55.97	63.88	78.16	60.72	68.34
CRF (ℓ_1)	73.32	58.34	64.98	77.41	64.23	70.21
IRW-CRF	74.08	56.23	63.93	77.17	61.03	68.39
SVM ^{hmm}	72.23	57.01	63.72	74.83	61.23	67.35
OnlineMKL	72.64	62.64	67.27	75.06	65.52	69.97
MTL ^{hmm}	77.47	63.21	69.62	79.26	68.39	73.42
(a) Spanish (ESP)						
CRF (ℓ_2)	71.56	40.79	51.96	74.53	46.41	57.20
CRF (ℓ_1)	72.64	43.85	54.68	74.93	51.10	60.76
IRW-CRF	71.62	41.09	52.22	73.68	46.54	57.05
SVM ^{hmm}	58.57	39.98	47.52	60.15	44.15	50.92
OnlineMKL	63.69	43.39	51.61	68.09	49.43	57.28
MTL ^{hmm}	82.72	48.32	61.00	85.63	56.08	67.77
(b) Dutch (NED)						

3.4.2.4 Sensitivity of Parameter C

Figure 3.4 shows the variations of testing F1 measure over the range of C used in the experimental setting ⁷. MTL^{hmm} obtain better results than the rest of algorithms in the range with the large C , i.e. $C \geq 1$. Therefore, weighting groups of features can boost the F1 measure on all four Chinese word segmentation datasets in the framework of MTL^{struct}, but OnlineMKL degrades the performance due to the online mode. CRF(ℓ_1) varies greatly in terms of different C 's, which implies that aggressively removing features could degrade prediction performance ⁸.

⁷Due to expensive training costs on large structured prediction tasks, one cannot afford a finer grid of C parameter in the experiments.

⁸A finer grid of C parameter for tuning may improve the prediction performance of CRF(ℓ_1) on these four datasets.

3.4.3 Named Entity Recognition

All methods are evaluated on the CoNLL-2002 language-independent named entity recognition shared task⁹. The data consists of three files per language: training, validation (testa), and testing (testb). Spanish (ESP) (8,323/1,915/1,517) and Dutch (NED) (15,806/2,895/5,159) with the number of sentences in each file are used in this experiment. Similar to TP2 in Table 3.2, I enumerate in the range of $[-3, 3]$ the combination of uni-gram and bi-gram templates in terms of words and part-of-speech (POS) respectively, the combination of tri-gram on POS only, and a state-state transition template. There are 134 templates in total, and the number of features are 5,926,876 and 4,710,332 for ESP and NED, respectively.

Figure 3.5: The CPU time on Spanish and Dutch dataset by varying C .

3.4.3.1 Performance Analysis on All Templates

Table 3.5 shows the overall performance of different methods on Spanish and Dutch data, respectively. One can observe that CRF (ℓ_2) and SVM^{hmm} perform worse than other methods. This implies that template (feature) selection improves performance by pruning the noisy or redundancy templates (features). Another observation is that MTL^{hmm} outperforms CRF (ℓ_1), IRW-CRF and OnlineMKL with more than 3% on ESP and 7% on NED in terms of F1 on the testing data. Figure 3.5 shows the training times of each method where MTL^{hmm} is slower than CRFs and SVM^{hmm}, but it is more than

⁹<http://www.cnts.ua.ac.be/conll2002/ner/>

Figure 3.6: The relative learned template weights for Named entity recognition on Spanish and Dutch. The horizontal line with 1 stands for methods with the average weights.

10 times faster than OnlineMKL. Figure 3.6 shows the weights learned by OnlineMKL and MTL^{hmm} on both datasets. If one considers the template with its weight less than a relative small value such as 10^{-5} as redundant, MTL^{hmm} can remove 75 and 97 templates for ESP and NED, respectively, but OnlineMKL cannot remove any template. On both datasets, MTL^{hmm} obtains much sparser weights than OnlineMKL, and also selects a quite different subset of templates for different language; while OnlineMKL selects similar subset of templates due to the similar curves. All the observations imply that MTL^{hmm} is more promising than other methods.

3.4.3.2 Performance on the Selected Templates by MTL^{hmm}

To further justify the effectiveness, a subset of 20 templates is selected according to the learned weights of MTL^{hmm} for all methods. The overall performances are shown in Table 3.6. One can observe that the overall performances on the small subset of templates greatly outperform that on the full set for both CRF (ℓ_2) and SVM^{hmm} by more than 5% over both datasets. CRF (ℓ_1) and OnlineMKL also have around 2-3% improvement. MTL^{hmm} on ESP has 1.02% improvement, while a reduction of 0.71% happens on NED. IRW-CRF in this case shows the best performance on ESP, but worse on NED compared with MTL^{hmm} . However, IRW-CRF performs worse on the full set of templates shown in Table 3.5 which implies that IRW-CRF cannot tackle well the noisy templates. All these

Table 3.6: The overall performance of the comparing methods on Spanish and Dutch for named entity recognition with the selected subset of templates. The best results across different methods are shown in bold.

Method	Validation set			Test set		
	Prec	Recall	F1	Prec	Recall	F1
CRF (ℓ_2)	78.00	61.03	68.48	81.76	67.52	73.96
CRF (ℓ_1)	76.58	60.34	67.50	80.75	66.70	73.00
IRW-CRF	76.36	63.90	69.58	81.04	70.50	75.40
SVM ^{hmm}	74.39	62.82	68.12	78.17	68.50	73.02
OnlineMKL	73.17	63.60	68.05	75.58	68.19	71.70
MTL ^{hmm}	76.09	62.32	68.52	79.96	69.74	74.50
(a) Spanish (ESP)						
CRF (ℓ_2)	77.66	43.31	55.61	80.79	50.57	62.20
CRF (ℓ_1)	77.53	43.92	56.08	80.05	51.31	62.53
IRW-CRF	79.22	48.09	59.85	82.19	54.78	65.74
SVM ^{hmm}	70.16	43.96	54.05	74.77	50.90	60.57
OnlineMKL	67.63	45.83	54.64	69.60	51.99	59.52
MTL ^{hmm}	80.64	47.94	60.13	85.06	55.34	67.06
(b) Dutch (NED)						

observations imply that MTL^{hmm} is effective for selecting templates and has a stable even improved performance in the case of noisy or non-informative templates.

3.4.4 Dependency Parsing

Comparisons on CoNLL-X shared task¹⁰ are conducted: Multi-lingual Dependency Parsing, on Danish and Swedish where the number of sentences in the given files (training / testing) are (5,125/313) and (11,092/389), respectively. I use the templates in MST-Parser including: (1) POS (including CPOSTAG and POSTAG) trigrams: the POS of the head, that of the modifier and that of a word in between, for all distinct POS tags for the words between the head and the modifier. Each relative position from the head to the modifier can be considered as a different type of template. (2) The form of POS 4-gram: the POS of the head, modifier, word before/after head and word before/after modifier. (3) Two items: each template consists of two observations, e.g. head word,

¹⁰http://nextens.uvt.nl/~conll/post_task_data.html

Table 3.7: The results (MSTParser / MTL^{parse}) of the dependency parsing on Danish and Swedish. The best results across the compared methods are shown in bold.

	ACC	UASP	UAS
Danish	31.06 / 32.61	86.93 / 87.08	88.80 / 88.80
Swedish	37.28 / 37.53	83.65 / 84.14	85.76 / 86.40

head POS/LEMMA, child word, and child POS/LEMMA. All templates are conjoined with the direction of attachment as well as the distance between the two words creating the dependency. For the distance between two words creating the dependency is longer than 10, it is set to 10; If between 10 and 5, it is 5, otherwise it does not change. There are 568 and 612 templates which generate 2, 137, 252 and 2, 292, 216 features for Danish and Swedish, respectively. Three measures are employed to evaluate the performance of comparing methods: complete sentence accuracy (ACC), unlabeled attachment score including punctuation (UASP), and unlabeled attachment score excluding punctuation (UAS).

Figure 3.7: The relative learned template weights for Dependency Parsing on Danish and Swedish. The horizontal line with 10^0 stands for methods with the average weights.

The experimental results of comparisons over Danish and Swedish are shown in Table 3.7, and the corresponding learned weights of templates are presented in Figure 3.7. If the same selection strategy is used in named entity recognition task, MTL^{parse} can remove 500 and 508 templates for Danish and Swedish, respectively. It implies that the huge

Table 3.8: The overall results of $p\text{MTL}^{hmm}$ on PKU and ESP by varying p . The best results across the different p are shown in bold.

p	PKU				ESP (Test set)		
	Recall	Prec	F1	R_{iv}	Prec	Recall	F1
1	93.8	93.3	93.6	96.1	79.96	69.74	74.50
4/3	92.9	94.1	93.5	94.7	80.22	70.08	74.81
2	92.6	93.4	93.0	94.9	79.08	67.35	72.75
4	90.8	91.3	91.1	93.4	74.59	59.54	66.22
100	87.4	89.7	88.5	90.0	71.92	54.34	61.91
(a) small set of templates							
1	94.0	93.6	93.8	96.2	79.26	68.39	73.42
4/3	93.5	94.0	93.8	95.3	78.29	64.65	70.82
2	92.6	93.7	93.1	94.3	75.52	61.11	67.56
4	90.0	92.5	91.3	92.2	72.74	53.93	61.58
100	89.7	91.7	90.7	91.9	65.08	54.51	59.33
(b) large set of templates							

redundancy exists in the given set of templates. The similar phenomena can be observed: MTL^{parse} obtains comparable or even better results than MSTParser, and also achieves very sparse weights of templates. The weights of patterns in some senses could be helpful for the interpretation of semantic meaning of each language [18].

3.4.5 MTL^{struct} with p -Block-Norm Regularization

Experiments of $p\text{MTL}^{hmm}$ are carried out on PKU and ESP (Test set), and their corresponding templates are employed (see Section 3.4.2 and Section 3.4.3, respectively). The empirical results are shown in Table 3.8, where $p = 1$ is MTL^{hmm} trained by Algorithm 1, and $p=100$ is used to approximate infinity-block norm. One can observe that F1 score decreases when increasing p , except that $p=3/4$ obtains the best performance on ESP (Test set). For large p ($p \geq 2$), the performance degrades greatly. For $1 \leq p < 2$, F1 score can be varied in terms of data and the set of templates and is higher than $p \geq 2$. As proved in [44], the p -block-norm regularizer with $1 \leq p \leq 2$ is equivalent to the q -norm on the weights of base kernels where $q \geq 1$. However, the case of $p > 2$ cannot be represented by the penalties on the weights of base kernels. As the penalties on the weights of

Figure 3.8: The normalized weights of $p\text{MTL}^{hmm}$ on the large set of templates by varying p values on PKU and ESP datasets, respectively.

templates are more reasonable at least for the interpretation of feature engineering, this implies that the p -block-norm regularizer with $1 \leq p \leq 2$ can obtain good F1 scores on both datasets, but that with $p > 2$ cannot because of lacking of the interpretation as the weights of templates. The corresponding weights on the large set of templates are shown in Figure 3.8 where the weights are normalized by l_1 norm due to the scaling problem. From these observations, $1 \leq p < 2$ is recommended for sequence learning tasks studied in this thesis.

3.5 Conclusion

Structured prediction is an important modeling strategy for various real world applications. Instead of modeling the structure of specific application, in this Chapter, I explore the underlying feature structure over the input-output pairs which contributes to the success of discriminative models, such as CRFs and Structural SVMs. Structured prediction algorithms take more effort on how to model the interdependence among the output variables, but less consideration is taken on the feature engineering which is a non-trivial and tedious task for general users. To alleviate this issue, I propose a Multiple Template Learning (MTL^{struct}) paradigm to learn both the weight of each template and the structured prediction model in the batch mode, simultaneously. Learning the weights of these groups is formulated as a Multiple Kernel Learning (MKL) problem. I propose

to solve this MKL problem by an efficient cutting plane algorithm, and its convergence is presented. I further extend it to the p -block norm regularization and the modified algorithm is readily obtained. Two special cases are explored in the proposed MTL^{struct} framework, i.e. sequence labeling and dependency parsing. Extensive experimental results demonstrate that learning structured prediction model with weighting template can automatically interpret the importance of each templates, so users can arbitrarily define numerous templates without cautions of performance reduction. Both theoretical analysis and empirical results verified that the MTL^{struct} is more efficient and performs much better than OnlineMKL.

Chapter 4

Structured-Prediction-Based Performance Measure

4.1 Introduction

Motivated by the requirement of specific measures in different applications of image annotation, in this Chapter, I propose a unified multi-label learning framework which can optimize a variety of objective specific measures for image annotation, named as objective-guided image annotation. To evaluate the performance of automatic image annotation, a variety of performance criteria from different considerations have been proposed in different objective/applications of image annotation. According to the categorization of performance measures in [11], this Chapter mainly focuses on a large family of measures called bipartition, including example-based measures, macro-averaging measures and micro-averaging measures. I particularly focus on example-based measures for two reasons: First, they are tailor-made for image annotation tasks but are seldom explored in the literature; Second, since the number of keywords is usually much smaller than the number of images in training datasets, optimizing example based measures is more efficient than micro-averaging measures or macro-averaging measures. The core contributions of this Chapter are listed as follows:

- (i) I summarize a variety of objective-guided performance measures based on contingency table under a unified representation. The analysis reveals that macro-averaging measures are very sensitive to infrequent keywords, which is rarely studied

in literatures. Moreover, hamming measure is easily affected by skewed distributions which lead to be a bit over-estimated. An example illustrates the properties of these measures.

- (ii) To optimize the objective-guided performance measure, I propose a hierarchical structure of multi-layer hypothesis to unify all the contingency table based measures. The unified loss functions can be defined correspondingly. And then, I formulate these loss functions as relaxed surrogate functions. The learning model with the relaxed surrogate function can be optimized efficiently. The proposed framework can optimize all the bipartition measures without additional effort. SVM and SVM^{perf} [152] can be shown as special cases of this framework. ReverseMLL [15] by optimizing the macro-averaging measures in the retrieval setting is also a special case of this framework.
- (iii) Experiments are conducted in two settings by comparing with various baselines. These baselines include κNN , SVM, SVM^{perf} [152], reverseMLL [15], ML-KNN [128] and the state-of-the-art image annotation methods [118, 121]. On the two frequently used multi-label datasets, I evaluate a number of instantiated methods for optimizing the objective-guided measures. The results are consistent with the theoretical analysis. Experiments on four image annotation datasets – UIUC-Sport, LabelMe, NUS-WIDE-Object, and Corel5K – demonstrate superior performance of the proposed method over baselines. I also observed that optimizing example-based F_1 obtains consistent good performance, while optimizing example-based Precision at k or Recall at k are sensitive to the number k and the keyword distribution of image annotation datasets.

The rest of this Chapter is organized as follows. The objective-guided performance measures and its property analysis are given in Section 4.2 and Section 4.3. Section 4.4 illustrates the unified multi-label learning framework. Experimental results are shown in Section 4.5. The conclusive remarks are given in Section 5.9.

4.2 Objective-guided Performance Measures

In the task of image annotation, each image may be assigned by multiple keywords where each keyword is associated with one object from the given set of object vocabularies. Suppose that a dataset contains N images with L objects (keywords) in total. Given an input image denoted by $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \dots, x_{i,D}] \in R^D$, its corresponding output keywords are denoted by $\mathbf{y}_i = [y_{i,1}, y_{i,2}, \dots, y_{i,L}]$ where $y_{i,l} = +1$ if the l^{th} object appears in the i^{th} image, otherwise $y_{i,l} = -1$, $\forall l = 1, \dots, L; i = 1, \dots, N$. We can obtain a concise representation as the pair $(\mathbf{X}, \mathbf{Y})$ where the feature matrix $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \dots; \mathbf{x}_N] \in R^{N \times D}$ and its corresponding keyword matrix $\mathbf{Y} = [\mathbf{y}_1; \mathbf{y}_2; \dots; \mathbf{y}_N] \in \{-1, +1\}^{N \times L}$.

Objective evaluation of the effectiveness is a cornerstone of a learning algorithm in the real world applications. This chapter studies a large family of performance measures based on the contingency table. A contingency table is a type of matrix where each entry of this matrix denotes the co-occurrence frequency of random variables t and p in terms of m samples for each random variable, i.e., $\mathbf{t} = [t_1, \dots, t_m]$ and $\mathbf{p} = [p_1, \dots, p_m]$. Suppose that $\mathbf{t}$ is the ground truth and $\mathbf{p}$ is the prediction. A variety of performance measures shown in Table 4.1 are readily defined in terms of the contingency table.

The evaluation criteria for image annotation are quite different, but the aforementioned measures can be readily adapted for specific applications. In the next subsections,

Table 4.1: Contingency table based measures with the entries of contingency table as the count of $t = 1 \& p = 1$, $t = -1 \& p = 1$, $t = 1 \& p = -1$, $t = -1 \& p = -1$ to be a, b, c, d , respectively.

Measure	Formula
Precision	$\text{Prec}(\mathbf{t}, \mathbf{p}) = \frac{a}{a+b}$
Recall	$\text{Rec}(\mathbf{t}, \mathbf{p}) = \frac{a}{a+c}$
F_1	$F_1(\mathbf{t}, \mathbf{p}) = \frac{2a}{2a+b+c}$
Precision at k	$\text{Prec}@k(\mathbf{t}, \mathbf{p}) = \frac{a}{a+b}$, s.t. $a + b = k$
Recall at k	$\text{Rec}@k(\mathbf{t}, \mathbf{p}) = \frac{a}{a+c}$, s.t. $a + b = k$
PRBEP	$\text{PRBEP}(\mathbf{t}, \mathbf{p}) = \frac{a}{a+c}$, s.t. $a + b = a + c$
Accuracy	$\text{Acc}(\mathbf{t}, \mathbf{p}) = \frac{a}{a+b+c}$
Hamming	$\text{Ham}(\mathbf{t}, \mathbf{p}) = \frac{a+d}{a+b+c+d}$

I will show that a variety of the adapted measures for a specific application are defined on its corresponding hypothesis and feature representation $\Phi(\mathbf{X}, \mathbf{Y})$.

4.2.1 Example-based Measures

Example-based measures evaluate the performance of image annotation according to its performance on each image. Therefore, it is usually preferred in image tagging or tag recommendation, which is also a kind of image annotation [126]. Given an image to be tagged, on one hand, the tags assigned to the image are expected to be as correct as possible; on the other hand, one also desires the tags assigned to the image as complete as possible, which means the image contents have been fully characterized. Therefore, example-based F_1 measure will be usually preferred [12, 120, 126, 127]. Such F_1 measure is calculated based on the precision at k and recall at k , where precision at k measures how correct of these k tags assigned to the image to be annotated, and recall at k measures to what extent the image has been characterized by these k tags [153]. Higher precision at k and recall at k will lead to higher F_1 measure (please refer to Table 4.1). Parameter k is usually predefined value, such as 5 widely used in the literature [12, 119].

According to the contingency table, the binary random variables t and p are associated with an image, and samples are the keywords of the ground truth and predictions of this image. Under this interpretation in the multi-label learning setting, for an image $\mathbf{x}$, we have the ground truth $\mathbf{t} = \mathbf{y}$ and its prediction $\mathbf{p} = h_-(\mathbf{x}; \mathbf{w})$ in terms of the hypothesis $h_- : R^d \mapsto \{-1, +1\}^L$ where each pair $(\mathbf{x}, \mathbf{y})$ is i.i.d. drawn from some unknown distribution and $\mathbf{w}$ is the model parameter.

By defining the joint feature map $\Phi(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^N \phi(\mathbf{x}_i, \mathbf{y}_i)$ with $\phi(\mathbf{x}_i, \mathbf{y}_i) = (\mathbf{x}_i \otimes \mathbf{y}_i)^T$ over both $\mathbf{x}_i$ and $\mathbf{y}_i$ where the operator $\otimes$ is the Kronecker product [6], and the i.i.d. assumption over images, the following equality holds:

$$\max_{\mathbf{Y} \in \{-1, +1\}^{N \times L}} \sum_{i=1}^N \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) = \sum_{i=1}^N \max_{\mathbf{y} \in \{-1, +1\}^L} \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}).$$

Hence, we can predict the keywords for each image individually by solving the following problems, $\forall i = 1, \dots, N$,

$$h_-(\mathbf{x}_i; \mathbf{w}) = \arg \max_{\mathbf{y} \in \{-1, +1\}^L} \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}). \quad (4.1)$$

All the measures in Table 4.1 can be adapted for each image individually. For convenience, denote a general representation of the measures in Table 4.1 by the symbol M . The example-based measures are defined as the average of the performance measures over images:

$$M_{example}(\mathbf{Y}, h_-(\mathbf{X}; \mathbf{w})) = \frac{1}{N} \sum_{i=1}^N M(\mathbf{y}_i, h_-(\mathbf{x}_i; \mathbf{w})). \quad (4.2)$$

4.2.2 Macro-averaging Measures

In tag-based image retrieval tasks [123], if the query is a keyword in the given set of vocabularies, the retrieval system needs to return k relevant images which contain this keyword. In this case, macro-averaging measures are frequently used in the literature [124, 125]. The mean of break-even point (mBEP) is used in information retrieval from multi-word queries [121]. Denote the l^{th} column of $\mathbf{Y}$ by Υ^l , i.e., $\mathbf{Y} = [\Upsilon^1, \Upsilon^2, \dots, \Upsilon^L]$. Different from the interpretation for the example-based measures, the binary random variables have different meanings: $\mathbf{t} = \Upsilon$ and $\mathbf{p} = h_l(\mathbf{x}; \mathbf{w})$ in terms of the hypothesis $h_l : \mathcal{X} \mapsto \{-1, +1\}^N$ where the pair $(\mathbf{X}, \Upsilon)$ is i.i.d. drawn from some unknown distribution.

Note that for the l^{th} keyword, there is an associated model parameter $\mathbf{w}^l$. By utilizing the feature representation $\Phi(\mathbf{X}, \mathbf{Y}) = [\psi(\mathbf{X}, \Upsilon^1); \dots; \psi(\mathbf{X}, \Upsilon^L)]$ with the joint feature map $\psi(\mathbf{X}, \Upsilon^l) = \sum_{i=1}^N y_{i,l} \mathbf{x}_i^T$ [152], $\mathbf{w} = [\mathbf{w}^1; \dots; \mathbf{w}^L]$, and the i.i.d. assumption over the keywords, we obtain the following equality:

$$\max_{\mathbf{Y} \in \{-1, +1\}^{N \times L}} \mathbf{w}^T \Phi(\mathbf{X}, \mathbf{Y}) = \sum_{l=1}^L \max_{\Upsilon \in \{-1, +1\}^N} \langle \mathbf{w}^l, \psi(\mathbf{X}, \Upsilon) \rangle.$$

Based on this assumption, the hypothesis turns out to be

$$h_l(\mathbf{X}; \mathbf{w}^l) = \arg \max_{\Upsilon \in \{-1, +1\}^N} \langle \mathbf{w}^l, \psi(\mathbf{X}, \Upsilon) \rangle, \quad (4.3)$$

$\forall l = 1, \dots, L$, independently. Hence, the macro-averaging measures are defined as

$$M_{macro}(\mathbf{Y}, h_l(\mathbf{X}; \mathbf{w})) = \frac{1}{L} \sum_{l=1}^L M(\Upsilon^l, h_l(\mathbf{X}; \mathbf{w}^l)). \quad (4.4)$$

For the l^{th} keyword, the learning algorithm learns the hypothesis h_l from a tuple of images $\mathbf{X} = (\mathbf{x}_1; \dots; \mathbf{x}_N)$ to a tuple of keywords $\Upsilon^l = [y_{1,l}; \dots; y_{N,l}]$. This hypothesis

is the same as that of SVM^{perf} [152] for binary classification problems where SVM is the special case of the hinge loss that is a convex upper bound of zero-one loss. The similar hypothesis is also used to optimize average precision [154]. However, the proposed formula is an extension to image annotation problem where each image may be assigned by multiple keywords.

4.2.3 Micro-averaging Measures

Similar to macro-averaging measures, micro-averaging measures are also used for the evaluation of image annotation [129]. Macro-averaging measures give equal weight to every keyword, while micro-averaging measures give equal weight to every image, i.e., an average over all the image/keyword pairs [138].

The corresponding measures are defined on the hypothesis $h : R^{N \times D} \mapsto \{-1, +1\}^{N \times L}$ as:

$$h(\mathbf{X}; \mathbf{w}) = \arg \max_{\mathbf{Y} \in \{-1, +1\}^{N \times L}} \mathbf{w}^T \Phi(\mathbf{X}, \mathbf{Y}), \quad (4.5)$$

since there is no independence assumption over images or keywords. As shown in Section 4.4.3, the specific design of $\Phi(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^N (\mathbf{y}_i \otimes \mathbf{x}_i)^T$ makes an easy and unified prediction.

The samples of the binary random variables for the contingency table are $\mathbf{t} = \text{vec}(\mathbf{Y})$ and $\mathbf{p} = \text{vec}(h(\mathbf{X}; \mathbf{w}))$ where $\text{vec}(A)$ is a function which transforms a matrix A to a vector by concatenating all the columns of this matrix A . Similarly, the micro-averaging measures are defined as:

$$M_{micro}(\mathbf{Y}, h(\mathbf{X}; \mathbf{w})) = M(\text{vec}(\mathbf{Y}), \text{vec}(h(\mathbf{X}; \mathbf{w}))). \quad (4.6)$$

4.2.4 Hamming Measure

Hamming measure is frequently used as the evaluation criterion in the multi-label classification problem [11, 15, 128] which is defined over the hypothesis $h_+ : R^D \mapsto \{-1, +1\}$. This assumes that the input-output pair $(\mathbf{x}_i, y_{i,l})$ for i^{th} image with l^{th} keyword is i.i.d. drawn from some unknown distribution. The training data is denoted by $\{(\mathbf{x}_i, y_{i,l}), \forall i =$

Figure 4.1: (a) The frequency of each keyword on UIUC-Sport dataset. (b) The effect on the bipartition measures with varying numbers of $\{a\}$ -Rare keywords (the number of columns).

$1, \dots, N; l = 1, \dots, L\}$ with the joint feature map $\varphi(\mathbf{x}, y) = y\mathbf{x}^T$. According to the independence assumption over both the image and the keyword, the following equality holds naturally:

$$\max_{\mathbf{Y} \in \{-1, +1\}^{N \times L}} \mathbf{w}^T \Phi(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^N \sum_{l=1}^L \max_{y \in \{-1, +1\}} \langle \mathbf{w}^l, \varphi(\mathbf{x}_i, y) \rangle \quad (4.7)$$

which is equivalent to predict the l^{th} keyword for the i^{th} image individually, $\forall i = 1, \dots, N; l = 1, \dots, L$,

$$h_+(\mathbf{x}_i; \mathbf{w}^l) = \arg \max_{y \in \{-1, +1\}} \langle \mathbf{w}^l, \varphi(\mathbf{x}_i, y) \rangle. \quad (4.8)$$

From (4.8), it is similar to learn a binary classifier for each keyword separately. The hamming measure is defined as the average of zero-one score over images and keywords

$$Ham(\mathbf{Y}, h_+(\mathbf{X}; \mathbf{w})) = \frac{1}{NL} \sum_{i=1}^N \sum_{l=1}^L \Pi(y_{i,l} = h_+(\mathbf{x}_i; \mathbf{w}^l)). \quad (4.9)$$

4.3 Analysis under Image Annotation Setting

In this section, the property of macro-averaging measures for image annotation tasks is analyzed. As shown in Figure 4.1(a), the keywords of image repositories usually follow the distribution of Power Law [155] where many keywords have very low keyword frequency.

For those keywords, due to the rareness of positive images, usually the corresponding classifier is not robust and predicts negative. Hence, the following definition for the rare keywords is presented.

Definition 5. **{a}-Rare keyword** is a keyword with low keyword frequency in the image collection, and its true positive a defined in the contingency table is very small.

Based on this definition, one can observe the following property:

Property 1. Since the numerator of all the measures in Table 4.1 except the hamming measure depend on the true positive a only, according to the formula in (4.4), the corresponding macro-averaging measures, which compute the average of measures over keywords, will decrease vastly if a **{a}-rare keyword** is added into the consideration.

4.3.1 An Illustrating Example

In this section, an example following Definition 5 is constructed. Given a set of images $\{i_1, i_2, \dots, i_9\}$ and its associated keyword vocabulary $\{k_1, k_2, \dots, k_{10}\}$, the ground truth keyword matrix is denoted by $\mathbf{Y}$

$\mathbf{Y}$	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}
i_1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1
i_2	+1	-1	+1	-1	-1	-1	-1	-1	-1	-1
...	...	...	...	...	...	...	...	...	...	...
i_9	+1	-1	-1	-1	-1	-1	-1	-1	-1	+1

where $\mathbf{Y}$ is a 9×10 matrix with all elements as -1 except the first column and $\mathbf{Y}_{j,j+1}, \forall j = 1, \dots, 9$ as $+1$, and its predicted keyword matrix is constructed as

$\mathbf{h}(\mathbf{X})$	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}
i_1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1
i_2	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1
...	...	...	...	...	...	...	...	...	...	...
i_9	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1

where all the rows are the same. For the keyword k_2 to k_{10} with $a = 0$, $b = 0$, $c = 1$, and $d = 8$ according to the contingency table of its corresponding columns in terms of $\mathbf{Y}$ and $\mathbf{h}(\mathbf{X})$, one can consider these keywords as the $\{0\}$ -rare keywords. According to the Property 1, one can conclude that macro-averaging measures are sensitive when adding more rare keywords.

For the example, Figure 4.1(b) shows the scores of different bipartition performance measures by varying the columns of $\mathbf{Y}$. One can observe that macro-averaging F_1 varies

Table 4.2: The summary of hypotheses and their corresponding score functions, loss functions and the set of i.i.d. samples.

Hypothesis $\mathfrak{h}$	Compatibility function	Loss function $c(\mathfrak{x}, \mathfrak{y}, \mathfrak{h})$	i.i.d. samples $(\mathfrak{x}, \mathfrak{y})$
h	$f(\mathbf{X}, \tilde{\mathbf{Y}}) = \mathbf{w}^T \Phi(\mathbf{X}, \tilde{\mathbf{Y}})$	$\Delta = 100(1 - M(\text{vec}(\tilde{\mathbf{Y}}), \text{vec}(h(\mathbf{X}; \mathbf{w}))))$	$(\mathbf{X}, \mathbf{Y})$
h_-	$f(\mathbf{x}_i, \mathbf{y}) = \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y})$	$\Delta_i = 100(1 - M(\mathbf{y}_i, h_-(\mathbf{x}_i; \mathbf{w})))$	$(\mathbf{x}_i, \mathbf{y}_i), \forall i = 1, \dots, N$
h_l	$f(\mathbf{X}, \Upsilon^l) = \langle \mathbf{w}^l, \psi(\mathbf{X}, \Upsilon^l) \rangle$	$\Delta_l = 100(1 - M(\Upsilon^l, h_l(\mathbf{X}; \mathbf{w}^l)))$	$(\mathbf{X}, \Upsilon^l), \forall l = 1, \dots, L$
h_+	$f(\mathbf{x}_i, y) = \langle \mathbf{w}^l, \varphi(\mathbf{x}_i, y) \rangle$	$\Delta_{i,l} = 1 - \Pi(y = h_+(\mathbf{x}_i; \mathbf{w}^l))$	$(\mathbf{x}_i, y_{i,l}), \forall i = 1, \dots, N; \forall l = 1, \dots, L$

greatly comparing with other measures. This verifies the sensitivity of macro-averaging measures when a lot of $\{0\}$ -rare keywords appear in the dataset. The Hamming score in this case is very high, while there is no image which is totally annotated correctly, so this score is a little over-estimated. It is because hamming loss also considers the negative-negative pair into contribution. It is also pointed out that hamming measure or zero-one loss is not appropriate in the case of imbalanced binary classification problem [152]. As will be shown in Section 4.4.4, optimizing micro-averaging measures is not efficient, so optimizing example-based measures is the main focus in the experiments.

4.4 Optimizing Objective-guided Performance Measures

In this section, I first unify the risks in terms of objective-guided measures. And then, I construct a generalized learning algorithm to optimize all the empirical risks for image annotation.

4.4.1 Objective-Guided Risks

As discussed in Section 4.2, the objective specific image annotation requires its corresponding performance measure to be the proper criterion to evaluate how successful the learning method is. Figure 4.2 summarizes the hypothesis space where the associated measure is defined and which sampling assumption is required to generate the training and testing data. Following the standard machine learning setup in the supervised scenario, the goal is to learn a function from its corresponding hypothesis space.

Figure 4.2: The hierarchical structure of multi-layer hypotheses where the rectangle stands for the keyword matrix $\mathbf{Y}$. The blank area in the rectangle means no independence assumption.

Table 4.2 summarizes the hypotheses in the different layer. Taking the example-based hypothesis h_- for example, the triplet $(h_-, (\mathbf{x}_i, \mathbf{y}_i)_{i=1}^N, \Delta_i)$ leads to the empirical risk

$$R_{emp}[h_-] = \frac{1}{N} \sum_{i=1}^N \Delta_i. \quad (4.10)$$

The rest of empirical risks can be similarly constructed based on their own i.i.d. sampling assumption.

4.4.2 Proposed Formulation

The goal of a learning method is to learn a hypothesis $\mathbb{h} : \mathcal{X} \mapsto \mathcal{Y}$ from some hypothesis space $\mathcal{H}$. The idea is to learn a compatibility function $f : \mathcal{X} \times \mathcal{Y} \mapsto R$ over input/output pairs such that the prediction can be defined for a given input $\mathbf{x}$ as

$$\mathbb{h}_{\mathbf{w}}(\mathbf{x}) = \arg \max_{\mathbf{y} \in \mathcal{Y}} f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = \arg \max_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}^T \Psi(\mathbf{x}, \mathbf{y}) \quad (4.11)$$

where $\mathbf{w}$ is a parameter vector and $\Psi(\mathbf{x}, \mathbf{y})$ is a joint feature vector over the input $\mathbf{x}$ and output $\mathbf{y}$.

The regularized risk minimization is employed to estimate the parameter vector $\mathbf{w}$ by solving the following optimization problem

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C R_{emp}[\mathbb{h}_{\mathbf{w}}] \quad (4.12)$$

However, empirical risk $R_{emp}[\mathbf{h}_{\mathbf{w}}]$ is usually non-convex and discontinuous, so it is difficult to directly optimize. Suppose that $\mathbf{w}$ satisfies Problem (4.11). The cost function has the following upper bound for all $(\mathbf{x}_i, \mathbf{y}_i), i = 1, \dots, m$

$$c(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_{\mathbf{w}}(\mathbf{x}_i)) \leq \max_{\mathbf{y} \in \mathcal{Y}} c(\mathbf{x}_i, \mathbf{y}_i, \mathbf{y}) + f_{\mathbf{w}}(\mathbf{x}_i, \mathbf{y}) - f_{\mathbf{w}}(\mathbf{x}_i, \mathbf{y}_i)$$

which is a piecewise linear convex function (margin rescaling). To optimize the upper bound instead of the empirical risk, following [14], we can formulate (4.12) as the following constrained optimization problem with 1-slack variable

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C\xi \\ \text{s.t.} \quad & \forall (\mathbf{y}'_1, \mathbf{y}'_2, \dots, \mathbf{y}'_m) \in \mathcal{Y}^m \\ & \frac{1}{m} \sum_{i=1}^m \mathbf{w}^T d\Psi_i(\mathbf{y}'_i) \geq \frac{1}{m} \sum_{i=1}^m c(\mathbf{x}_i, \mathbf{y}_i, \mathbf{y}'_i) - \xi \end{aligned} \tag{4.13}$$

where $d\Psi_i(\mathbf{y}'_i) = \Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \mathbf{y}'_i)$.

4.4.3 Joint Feature Map for Bipartition Measures

To optimize the performance measures for image annotation, one first needs to define the joint feature map for each hypothesis. In this Chapter, the following feature mapping is explored:

$$\Phi(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^N (\mathbf{y}_i \otimes \mathbf{x}_i)^T \tag{4.14}$$

where the operator $\otimes$ is the Kronecker product. Based on this, we can derive all the joint feature functions corresponding to each hypothesis in Figure 4.2 by the following equalities:

$$\mathbf{w}^T \Phi(\mathbf{X}, \mathbf{Y}) \stackrel{\substack{\text{i.i.d over} \\ \text{image \& keyword}}}{=} \sum_{i=1}^N \sum_{l=1}^L \langle \mathbf{w}^l, \varphi(\mathbf{x}_i, y_{i,l}) \rangle \tag{4.15}$$

$$\stackrel{\substack{\text{i.i.d over} \\ \text{image}}}{=} \sum_{i=1}^N \langle \mathbf{w}, \phi(\mathbf{x}_i, \mathbf{y}_i) \rangle \tag{4.16}$$

$$\stackrel{\substack{\text{i.i.d over} \\ \text{keyword}}}{=} \sum_{l=1}^L \langle \mathbf{w}^l, \psi(\mathbf{X}, \Upsilon^l) \rangle. \tag{4.17}$$

where the feature functions are $\varphi(\mathbf{x}, y) = y\mathbf{x}^T$, $\phi(\mathbf{x}, \mathbf{y}) = (\mathbf{y} \otimes \mathbf{x})^T$, and $\psi(\mathbf{X}, \Upsilon^l) = \sum_{i=1}^N y_{i,l} \mathbf{x}_i^T$.

The following proposition shows the good property of these feature functions for easy prediction.

Proposition 3. *Inference problems (4.1), (4.3), (4.5) and (4.8) to find the label configuration with the highest score have the same analytical solutions regardless of different hypotheses. Given an input $\mathbf{x}$, solutions are*

$$y_l = \begin{cases} +1, & \langle \mathbf{w}^l, \mathbf{x} \rangle > 0 \\ -1, & \text{otherwise.} \end{cases}, \forall l = 1, \dots, L. \quad (4.18)$$

Proof: According to the feature function $\Phi(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^N (\mathbf{y}_i \otimes \mathbf{x}_i)^T$, all the prediction rules are equal to the following problem,

$$\arg \max_{\mathbf{Y} \in \{-1, +1\}^{N \times L}} \sum_{i=1}^N \sum_{l=1}^L \langle \mathbf{w}^l, \varphi(\mathbf{x}_i, y_{i,l}) \rangle.$$

Keywords and images are independent in terms of the feature representations (4.14)-(4.17), so the corresponding solutions (4.18) are readily obtained. $\square$

4.4.4 Generalized Learning Algorithm

Given the training data $(\mathbf{X}, \mathbf{Y})$, the following proposition shows that there exists a generalized optimization problem for all hypotheses discussed in this Chapter, which is analogous to the 1-slack variable structural SVMs with margin rescaling formulation [14]. Correspondingly, the empirical risks defined in Section 4.4.1 for the measures of image annotation can be easily incorporated by replacing the cost functions.

Proposition 4. *Suppose that the joint feature functions are defined in the form of (4.14) - (4.17). The optimization problem over each hypothesis in Table 4.2 becomes a general form*

$$\begin{aligned} \min_{\mathbf{W}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{W}\|_F^2 + C\xi \\ \text{s.t.} \quad & \forall \tilde{\mathbf{Y}} \in \{-1, +1\}^{N \times L} : \\ & \xi \geq \Delta(\mathbf{Y}, \tilde{\mathbf{Y}}) + \frac{1}{\gamma} \text{trace} \left((\tilde{\mathbf{Y}} - \mathbf{Y}) \mathbf{W}^T \mathbf{X}^T \right) \end{aligned} \quad (4.19)$$

where $\mathbf{W} = [\mathbf{w}^1, \dots, \mathbf{w}^L] \in R^{D \times L}$, the Frobenius norm $\|\mathbf{W}\|_F = \sqrt{\text{trace}(\mathbf{W}\mathbf{W}^T)}$, and loss functions over different hypotheses

$$\Delta(\mathbf{Y}, \tilde{\mathbf{Y}}) = \begin{cases} \Delta & \gamma = 1, \\ \frac{1}{N} \sum_{i=1}^N \Delta_i & \gamma = N, \\ \frac{1}{L} \sum_{l=1}^L \Delta_l & \gamma = L, \\ \frac{1}{NL} \sum_{i=1}^N \sum_{j=1}^L \Delta_{i,l} & \gamma = NL. \end{cases} \quad (4.20)$$

Proof: According to (4.14) and some linear algebra operations, we obtain the equality $\mathbf{w}^T \Phi(\mathbf{X}, \mathbf{Y}) = \text{trace}(\mathbf{Y}\mathbf{W}^T \mathbf{X}^T)$. It is trivial for the case of $\gamma = 1$ where the micro-averaging measures are used in terms of hypothesis h since there is only one pair $(\mathbf{X}, \mathbf{Y})$.

For other cases, one can take the hypothesis h_- for example. According to [6], we can formulate an N -slack variable structural SVM with margin re-scaling

$$\begin{aligned} \min_{\mathbf{W}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{W}\|_F^2 + C \left\{ \frac{1}{N} \sum_{i=1}^N \xi_i \right\} \\ \text{s.t.} \quad & \forall i, \forall \tilde{\mathbf{y}}_i \in \{-1, +1\}^L \setminus \mathbf{y}_i : \\ & \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{w}^T \phi(\mathbf{x}_i, \tilde{\mathbf{y}}_i) + \Delta_i - \xi_i \end{aligned} \quad (4.21)$$

where the second term in the objective is the empirical risk,

$$\begin{aligned} \frac{1}{N} \sum_{i=1}^N \xi_i &= \frac{1}{N} \sum_{i=1}^N \max_{\tilde{\mathbf{y}}_i \in \{-1, +1\}^L \setminus \mathbf{y}_i} \{0, \Delta_i + \mathbf{w}^T (\phi(\mathbf{x}_i, \tilde{\mathbf{y}}_i) - \phi(\mathbf{x}_i, \mathbf{y}_i))\} \\ &= \max_{\tilde{\mathbf{Y}} \in \{-1, +1\}^{N \times L}} \left\{ 0, \frac{1}{N} \sum_{i=1}^N [\Delta_i + \mathbf{w}^T (\phi(\mathbf{x}_i, \tilde{\mathbf{y}}_i) - \phi(\mathbf{x}_i, \mathbf{y}_i))] \right\} \\ &= \max_{\tilde{\mathbf{Y}} \in \{-1, +1\}^{N \times L}} \left\{ 0, \frac{1}{N} \sum_{i=1}^N \Delta_i + \frac{1}{N} \text{trace} \left((\tilde{\mathbf{Y}} - \mathbf{Y}) \mathbf{W}^T \mathbf{X}^T \right) \right\} \\ &= \xi. \end{aligned}$$

The third equality in the derivation comes from the evidences that the hypothesis h_- assumes the i.i.d. samples over images, as well as $\Delta_i(\mathbf{y}_i, \mathbf{y}_i) = 0$ for all measures in Table 4.1. Following the similar derivation, we can obtain the proof for hypotheses h_l and h_+ . $\square$

Cutting plane algorithm has been shown efficiently for 1-slack structural SVM with margin-rescaling problem [14]. The algorithm iteratively constructs a working set $\mathcal{W}$ of

Algorithm 2 Training for general loss functions

- 1: Input: $\mathbb{S} = (\mathbf{X}, \mathbf{Y})$, C , ϵ , loss function Δ and γ
 - 2: $\mathcal{W} = \emptyset$, $\mathbf{W} = \mathbf{0}$
 - 3: Find the most violated $\tilde{\mathbf{Y}}$ based on $\mathbf{W}$ and Δ
 - 4: **repeat**
 - 5: $\mathcal{W} = \mathcal{W} \cup \{\tilde{\mathbf{Y}}\}$
 - 6: Solve Problem (4.19) over the reduced set $\mathcal{W}$ and obtain $(\mathbf{W}, \xi)$
 - 7: Find the most violated $\tilde{\mathbf{Y}}$ based on $\mathbf{W}$ and Δ
 - 8: **until** $\Delta(\mathbf{Y}, \tilde{\mathbf{Y}}) + \frac{1}{\gamma} \text{trace}((\tilde{\mathbf{Y}} - \mathbf{Y})\mathbf{W}^T \mathbf{X}^T) \leq \xi + \epsilon$
-

constraints. In each iteration, the algorithm computes the solution over the current $\mathcal{W}$, finds the most violated constraint, and adds it to $\mathcal{W}$. The algorithm stops once we cannot find any constraint which is violated by more than the desired precision ϵ . Algorithm 2 shows the cutting plane algorithm for solving Problem (4.19) with general loss functions.

4.4.5 The Most Violated Constraint for Bipartition Measures

Recall that finding the most violated constraint is specific to each application, and is also one of the most expensive procedure in cutting plane algorithms. In order to find the most violated constraint for bipartition measures, the following integer programming should be solved:

$$\mathbf{Y}^* = \arg \max_{\tilde{\mathbf{Y}} \in \{-1, +1\}^{N \times L}} \Delta(\mathbf{Y}, \tilde{\mathbf{Y}}) + \frac{1}{\gamma} \text{trace}((\tilde{\mathbf{Y}} - \mathbf{Y})\mathbf{W}^T \mathbf{X}^T). \quad (4.22)$$

The reverse procedure of the proof in Proposition 4 motivates us to decompose Problem (4.22) into several independent subproblems according to their corresponding hypothesis. Taking example-based measures for example, we can obtain the following equivalent problems, $\forall i = 1, \dots, N$,

$$\mathbf{y}_i^* = \arg \max_{\tilde{\mathbf{y}} \in \{-1, +1\}^L} 100(1 - M(\mathbf{y}_i, \tilde{\mathbf{y}})) + \mathbf{w}^T \delta \phi_i(\tilde{\mathbf{y}}) \quad (4.23)$$

where $\delta \phi_i(\tilde{\mathbf{y}}) = \phi(\mathbf{x}_i, \tilde{\mathbf{y}}) - \phi(\mathbf{x}_i, \mathbf{y}_i)$. For each individual problem with the measures defined in Table 4.1, it can be solved exactly based on contingency table by Algorithm 2 in [152]. Based on $\mathbf{Y}^* = [\mathbf{y}_1^*; \dots, \mathbf{y}_N^*]$, we obtain the most violated constraint efficiently,

$$\xi \geq \Delta(\mathbf{Y}, \mathbf{Y}^*) + \frac{1}{\gamma} \text{trace}((\mathbf{Y}^* - \mathbf{Y})\mathbf{W}^T \mathbf{X}^T). \quad (4.24)$$

4.4.6 Subproblem over the Set of Violated Constraints

At the r th iteration of Algorithm 2, we can obtain a collection of the most violated constraints based on $\mathcal{W}_r = \{\tilde{\mathbf{Y}}^1, \tilde{\mathbf{Y}}^2, \dots, \tilde{\mathbf{Y}}^r\}$. The reduced problem turns out to be

$$\begin{aligned} \min_{\mathbf{W}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{W}\|_F^2 + C\xi \\ \text{s.t.} \quad & \forall \tilde{\mathbf{Y}} \in \mathcal{W}_r : \\ & \xi \geq \Delta(\mathbf{Y}, \tilde{\mathbf{Y}}) + \frac{1}{\gamma} \text{trace} \left((\tilde{\mathbf{Y}} - \mathbf{Y}) \mathbf{W}^T \mathbf{X}^T \right). \end{aligned} \quad (4.25)$$

By constructing Lagrangian dual function with dual variables $\alpha \geq 0$, we can readily obtain its dual problem

$$\max_{\alpha \in \mathcal{A}} -\frac{1}{2} \sum_{\tilde{\mathbf{Y}} \in \mathcal{W}} \sum_{\tilde{\mathbf{Y}}' \in \mathcal{W}} \alpha_{\tilde{\mathbf{Y}}} \alpha_{\tilde{\mathbf{Y}}'} \mathbf{H}(\tilde{\mathbf{Y}}, \tilde{\mathbf{Y}}') + \sum_{\tilde{\mathbf{Y}} \in \mathcal{W}} \alpha_{\tilde{\mathbf{Y}}} \Delta(\mathbf{Y}, \tilde{\mathbf{Y}}) \quad (4.26)$$

where the feasible region $\mathcal{A} = \{\sum_{\tilde{\mathbf{Y}} \in \mathcal{W}} \alpha_{\tilde{\mathbf{Y}}} \leq C, \alpha \geq 0\}$, $\mathbf{H}(\tilde{\mathbf{Y}}, \tilde{\mathbf{Y}}') = \frac{1}{\gamma^2} \|\mathbf{X}^T(\tilde{\mathbf{Y}} - \tilde{\mathbf{Y}}')\|_F^2$, and primal variables can be recovered by $\mathbf{W} = \frac{1}{\gamma} \sum_{\tilde{\mathbf{Y}} \in \mathcal{W}} \alpha_{\tilde{\mathbf{Y}}} \mathbf{X}^T(\mathbf{Y} - \tilde{\mathbf{Y}})$.

4.4.7 Complexity Analysis

The optimization problems associated to different hypotheses are solved by the same algorithm, but the problem complexities are quite different. According to the Algorithm 2 in [152], if the size of the vector $\mathbf{t}$ is n , i.e., $|\mathbf{t}|=n$, then the complexity to find the optimal $\tilde{\mathbf{Y}}$ based on the contingency table constructed by $\mathbf{t}$ is $O(n^2)$. If the loss is based on the *micro-averaging* measures, the problem cannot be decomposed. We need to find $\tilde{\mathbf{Y}}$ on the contingency tables with $|\mathbf{t}|=LN$ in $O(L^2N^2)$. Similarly, we can solve Problem (4.22) in $O(LN^2)$ for *macro-averaging* loss functions with L independent contingency table with $|\mathbf{t}|=N$, $O(L^2N)$ for *example-based* loss functions with N independent contingency table with $|\mathbf{t}|=L$, and $O(LN)$ for *Hamming* loss since Problem (4.22) can be solved independently. Fast inference contributes to the capability of decomposition of different loss functions. Therefore, the complexity analysis shows that *example-based* measures are more efficient to be directly optimized for image annotation problems since L is generally much smaller than N . Though *Hamming loss* is the measure that can be optimized in the most efficient way, it is improper to evaluate a learning method in the case of imbalanced problem [152] such as image annotation.

Problem (4.26) is a Quadratic Programming (QP) problem with $|\mathcal{W}|$ variables and $|\mathcal{W}| + 1$ constraints, which can be efficiently solved by off-shelf optimization tools as $|\mathcal{W}|$ is generally not large. Non-linear kernel functions can be used in this framework, but the computational complexity of calculating $H(\tilde{\mathbf{Y}}, \tilde{\mathbf{Y}}')$ is $O(LN^2)$, instead of $O(NL)$ in the linear case. For large scale multi-label problems, it is extremely time consuming and impossible to pre-compute or even store the entire kernel matrix in the memory. Hence, the models in the linear case is used.

4.5 Experiments

In this section, the proposed generalized learning algorithm for optimizing the objective-guided performance measures is empirically evaluated by comparing with a variety of state-of-the-art methods on benchmark datasets for multi-label learning and the popular benchmark datasets for image annotation.

4.5.1 Methods for Comparisons

Most of the performance measures listed in TABLE 4.1 are evaluated by comparing the proposed methods with the following baselines:

- **κ NN**: Nearest Neighbor classifier with Euclidean distance is used to find κ nearest neighbors from the training set in terms of the given image and all the keywords of its neighbors are assigned to this image.
- **SVM**: Support Vector Machine with the linear kernel is employed to train one model for each keyword. Given a test image, we predict each keyword independently. LIBSVM toolbox ¹ is used [156].
- **SVM^{perf}** [152]: Similar to **SVM**, one model is trained for each keyword by optimizing F_1 score individually. **SVM^{perf}** toolbox ² is used.
- **ML-KNN** [128]: It is a multi-label lazy learning method which is derived from the traditional κ NN algorithm, and maximum a posteriori principle is utilized to determine the label set for the test image. The ML-KNN toolbox ³ is available.

¹<http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

²http://svmlight.joachims.org/svm_perf.html

³<http://lamda.nju.edu.cn/datacode/MLkNN.htm>

- **reverseMLL** [15]: Reverse multi-label learning method is designed to directly optimize macro-averaging F_1 . The toolbox ⁴ is available.
- **label-transfer** [118]: Greedy label transfer is to annotate the test image based on its κ nearest neighbors, tag frequency and tag co-occurrence information.
- **Tagprop** [121]: It is a discriminatively trained nearest neighbor model and the keywords of the test image is predicted by a weighed nearest neighbor model with the image similarity metrics learned by a metric learning method. The Tagprop toolbox ⁵ is available.
- **IA**: The proposed methods for optimizing each measure separately only based on the linear kernel. Specifically, denote **IAF**₁ for example-based F_1 score, **IAP** _{k} for example-based Prec@ k and **IAR** _{k} for example-based Rec@ k .

The detailed settings for each method are described in the following subsections since some parameters depends on the characteristics of the dataset used. For all baseline methods, the results of each specific performance measure are reported by model selection according to the specific measure.

4.5.2 Experiments on Multi-label Datasets

4.5.2.1 Datasets

The two widely used datasets, *Emotions* and *Medical* ⁶, are employed to demonstrate the effectiveness of the generalized learning methods for a variety of performance measures. Emotions dataset [157] is for the task of the automated detection of emotion in music. A piece of music can belong to more than one of the following 6 categories: amazed-surprised, happy-pleased, relaxing-calm, quiet-still, sad-lonely, and angry-fearful. The data includes 593 songs with annotations. Medical data consists of 978 instances with 45 labels. The other statistics of both datasets can be accessed in its website.

⁴<http://users.cecs.anu.edu.au/~jpettersen/>

⁵<http://lear.inrialpes.fr/people/guillaumin/code.php#tagprop>

⁶<http://mulan.sourceforge.net/datasets.html>

4.5.2.2 Experimental Setting

I attempt to empirically demonstrate the characteristics of various performance measures which are given by the analysis in Section 4.2. Since *Emotions* only has 6 categories, the results for precision@k and recall@k will not be reported. For nearest neighbor based methods (κ NN, ML-KNN, label-transfer, and Tagprop), κ is tuned in the grid of [1, 3, 5, 10, 20, 50, 100]. For large margin based methods (SVM, SVM^{perf}, reverseMLL, and the proposed methods), C is tuned in the range of [0.01, 0.1, 1, 10, 100]. the termination condition parameter $\epsilon = 0.01$ is fixed for the proposed methods. Results are reported by performing *five-fold cross validation* on each dataset for all the compared methods.

4.5.2.3 Experimental Results

Table 4.3 and 4.4 shows the experimental results of proposed methods by comparing with other methods in terms of a variety of performance measures discussed in Table 4.1 and Section 4.2. On Emotions data, the instantiations of the proposed framework on most of performance measures outperform others. However, on Medical data, we observe that IA performs worse than SVM method greatly in terms of macro-averaging measures. This is consistent with the analysis in Section 4.3 since there are many keywords with low keyword frequency in this data collection. This leads to the sensitivity of optimizing macro-averaging measures on this data. Although IA does not perform the best in micro-averaging measures in Table 4.4, it demonstrates the most stable performance compared with other methods. On all the four tested micro-averaging measures, IA is comparable to the one with the best performance. The hamming score on Medical is nearly 99%, but the rest of measures is far away from this score, so hamming loss is not a proper measure for multi-label problem with a large number of labels. Figure 4.3 shows the time cost for training IA in terms of the specific performance measure used. It clearly illustrates that optimizing example-based measures are much cheaper than macro-averaging and micro-averaging measures. This agrees with the time complexity analysis in Section 4.4.7.

Table 4.3: A variety of performance measures (mean $\pm$ standard deviation, %) by comparing different methods on Emotions by five-fold cross validation except that hamming loss is one minus hamming measure without percentage. The best results are in bold.

Measure	IA	κ NN	SVM	SVM ^{perf}	ML-KNN	reverseMLL	label-transfer	Tagprop
hamming loss	0.20 $\pm$ 0.01	0.27 $\pm$ 0.01	0.20 $\pm$ 0.01	0.27 $\pm$ 0.01	0.27 $\pm$ 0.01	0.31 $\pm$ 0.02	0.30 $\pm$ 0.02	0.30 $\pm$ 0.02
example-F1	65.96 $\pm$ 1.47	50.48 $\pm$ 3.71	62.00 $\pm$ 1.11	65.87 $\pm$ 1.43	41.20 $\pm$ 4.56	61.09 $\pm$ 1.47	46.78 $\pm$ 2.01	49.19 $\pm$ 3.66
example-precision	63.34 $\pm$ 2.48	56.02 $\pm$ 4.06	70.16 $\pm$ 2.58	55.04 $\pm$ 1.20	52.29 $\pm$ 4.83	52.09 $\pm$ 2.24	50.44 $\pm$ 2.63	52.26 $\pm$ 2.49
example-recall	87.47 $\pm$ 1.78	50.39 $\pm$ 4.32	59.66 $\pm$ 2.93	86.39 $\pm$ 1.76	37.23 $\pm$ 4.39	82.35 $\pm$ 3.35	48.40 $\pm$ 2.23	51.51 $\pm$ 4.45
example-accuracy	55.30 $\pm$ 2.21	34.61 $\pm$ 2.68	46.87 $\pm$ 0.92	50.09 $\pm$ 1.73	33.76 $\pm$ 4.31	48.71 $\pm$ 1.64	38.95 $\pm$ 2.19	41.05 $\pm$ 3.32
macro-F1	68.24 $\pm$ 1.67	49.68 $\pm$ 2.34	58.74 $\pm$ 2.74	64.62 $\pm$ 0.81	35.73 $\pm$ 2.84	63.23 $\pm$ 1.90	48.47 $\pm$ 1.86	50.58 $\pm$ 3.80
macro-precision	64.74 $\pm$ 1.84	54.30 $\pm$ 2.03	63.71 $\pm$ 3.31	55.88 $\pm$ 0.96	45.15 $\pm$ 4.55	53.52 $\pm$ 2.70	50.74 $\pm$ 1.84	51.27 $\pm$ 3.35
macro-recall	86.23 $\pm$ 2.36	51.42 $\pm$ 4.31	60.40 $\pm$ 3.18	86.60 $\pm$ 1.07	32.83 $\pm$ 4.54	80.68 $\pm$ 4.17	46.82 $\pm$ 1.66	50.50 $\pm$ 4.48
macro-accuracy	53.00 $\pm$ 2.22	41.91 $\pm$ 2.24	51.10 $\pm$ 2.43	52.91 $\pm$ 0.98	24.92 $\pm$ 2.54	47.00 $\pm$ 2.10	32.71 $\pm$ 1.57	34.65 $\pm$ 3.62
micro-F1	68.26 $\pm$ 0.89	52.65 $\pm$ 3.09	64.80 $\pm$ 1.51	66.13 $\pm$ 1.20	45.77 $\pm$ 4.17	62.73 $\pm$ 1.59	50.17 $\pm$ 2.39	51.85 $\pm$ 4.23
micro-precision	66.03 $\pm$ 2.63	66.95 $\pm$ 6.74	71.24 $\pm$ 4.15	54.61 $\pm$ 1.28	62.63 $\pm$ 4.43	50.69 $\pm$ 2.27	51.81 $\pm$ 2.96	52.09 $\pm$ 4.03
micro-recall	87.06 $\pm$ 3.76	51.53 $\pm$ 4.42	60.23 $\pm$ 2.43	87.13 $\pm$ 1.51	36.38 $\pm$ 5.13	82.49 $\pm$ 3.52	48.65 $\pm$ 1.98	51.62 $\pm$ 4.57
micro-accuracy	51.89 $\pm$ 1.22	35.78 $\pm$ 2.82	47.95 $\pm$ 1.65	49.41 $\pm$ 1.35	29.75 $\pm$ 3.52	45.71 $\pm$ 1.67	33.52 $\pm$ 2.10	35.08 $\pm$ 3.87

Table 4.4: A variety of performance measures (mean $\pm$ standard deviation, %) by comparing different methods on Medical by five-fold cross validation except that hamming loss is one minus hamming measure without percentage. The best results are in bold.

Measure	IA	κ NN	SVM	SVM ^{perf}	ML-KNN	reverseMLL	label-transfer	Tagprop
hamming loss	0.01 $\pm$ 0.00	0.02 $\pm$ 0.00	0.01 $\pm$ 0.00	0.10 $\pm$ 0.01	0.02 $\pm$ 0.00	0.07 $\pm$ 0.02	0.02 $\pm$ 0.00	0.02 $\pm$ 0.00
example-F1	75.21 $\pm$ 2.15	23.82 $\pm$ 2.67	35.93 $\pm$ 3.09	35.86 $\pm$ 0.80	52.61 $\pm$ 3.02	41.59 $\pm$ 7.97	45.48 $\pm$ 3.53	57.71 $\pm$ 3.70
example-precision	74.79 $\pm$ 1.50	27.61 $\pm$ 2.25	38.18 $\pm$ 2.49	30.92 $\pm$ 1.26	54.77 $\pm$ 2.55	29.75 $\pm$ 7.40	46.91 $\pm$ 4.01	59.52 $\pm$ 3.41
example-recall	91.49 $\pm$ 1.95	23.75 $\pm$ 2.90	35.70 $\pm$ 4.00	57.99 $\pm$ 2.51	52.38 $\pm$ 3.39	84.38 $\pm$ 2.33	46.82 $\pm$ 3.06	59.33 $\pm$ 4.01
example-accuracy	70.29 $\pm$ 1.46	18.36 $\pm$ 2.70	31.52 $\pm$ 2.87	29.54 $\pm$ 1.40	50.49 $\pm$ 3.14	29.33 $\pm$ 7.24	42.46 $\pm$ 3.46	54.22 $\pm$ 3.82
macro-F1	39.30 $\pm$ 2.55	55.53 $\pm$ 2.97	71.53 $\pm$ 3.04	58.35 $\pm$ 2.45	18.52 $\pm$ 1.96	23.10 $\pm$ 3.01	24.37 $\pm$ 3.43	24.63 $\pm$ 2.93
macro-precision	40.35 $\pm$ 4.49	58.09 $\pm$ 3.00	72.47 $\pm$ 3.37	49.74 $\pm$ 2.99	23.47 $\pm$ 3.55	18.57 $\pm$ 2.89	28.28 $\pm$ 4.19	29.04 $\pm$ 4.12
macro-recall	57.89 $\pm$ 2.90	56.06 $\pm$ 3.08	73.41 $\pm$ 2.93	91.36 $\pm$ 1.72	16.73 $\pm$ 1.57	46.91 $\pm$ 1.19	23.78 $\pm$ 3.80	24.82 $\pm$ 3.18
macro-accuracy	33.48 $\pm$ 2.86	52.36 $\pm$ 3.06	68.68 $\pm$ 3.04	48.44 $\pm$ 3.17	14.77 $\pm$ 1.63	17.14 $\pm$ 2.81	18.22 $\pm$ 2.85	19.42 $\pm$ 2.85
micro-F1	78.64 $\pm$ 1.86	63.92 $\pm$ 6.35	79.01 $\pm$ 1.71	56.44 $\pm$ 2.61	64.97 $\pm$ 3.05	41.05 $\pm$ 8.23	52.12 $\pm$ 3.59	65.00 $\pm$ 4.79
micro-precision	85.15 $\pm$ 1.63	85.33 $\pm$ 5.51	82.25 $\pm$ 2.30	41.73 $\pm$ 2.83	80.39 $\pm$ 2.94	26.98 $\pm$ 6.61	57.21 $\pm$ 3.74	78.06 $\pm$ 4.44
micro-recall	93.36 $\pm$ 1.87	57.96 $\pm$ 3.98	76.04 $\pm$ 1.63	95.72 $\pm$ 1.11	54.53 $\pm$ 3.09	88.45 $\pm$ 2.66	47.87 $\pm$ 3.53	61.89 $\pm$ 4.93
micro-accuracy	64.35 $\pm$ 2.85	47.23 $\pm$ 6.82	65.33 $\pm$ 2.33	39.35 $\pm$ 2.55	48.17 $\pm$ 3.28	26.09 $\pm$ 6.37	35.30 $\pm$ 3.29	48.30 $\pm$ 5.31

Figure 4.3: The training time of the proposed methods for optimizing the objective-guided performance measures on Medical data.

According to these observations, the example-based measures will only be investigated on the image annotation problems of the following experiments due to the following reasons: 1) image annotation data usually consists of hundreds of keywords and most of them are rare keywords, and macro-averaging measures are too sensitive to be optimized; 2) Optimizing micro-averaging measures is time-consuming. 3) Hamming score is over-estimated. 4) Example-based measures are tailor-made for image annotation tasks.

4.5.3 Experiments on Image Annotation Datasets

4.5.3.1 Datasets

Four popular benchmark datasets are considered, which have been used in the previous work for image annotation. UIUC-Sport [158] dataset contains 1792 images with 8 classes. The number of images in each class varies from 137 to 250. The images to 400 pixels are resized in terms of the maximum of the width or height and keep the aspect ratio. LabelMe is a subset from [159] with 2366 images in total. Similar to the work of [12, 119], images with 256×256 pixels are used. NUS-WIDE-Object dataset [160] contains 30,000 images and 31 classes where the photos and the associated tags are downloaded from

Figure 4.4: Histograms (a)-(d) show the frequency of images with the specific number of keywords on four datasets.

flickr.com. Following the process of [12], 26 classes with 4025 images are also used. Corel5K dataset [161] contains 5,000 images and 50 classes which are collected from the Corel CD set. On average, there are 7.5 keywords per image in the UIUC-Sport dataset with 330 keywords in all, 6.8 keywords per image in the LabelMe dataset with 812 keywords in all, 6.3 keywords per image in the NUS-WIDE-Object with 813 keywords in all, and 3.5 keywords per image in the Corel5K with 374 keywords in all. The detailed distribution of keywords in each dataset is shown in Figure 4.4. Following the feature representation in the work of [12], we adopt densely-sampled SIFT feature, whose step size and patch size are 4 and 24, respectively. Then we quantize all the features into 400 clusters using k-means clustering. Three-layer spatial pyramid representation [162] is used to preserve the spatial information.

Table 4.5: The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on UIUC-Sport by five-fold cross validation. The best results are in bold.

Method	F ₁	Prec@5	Rec@5
κ NN	49.32 $\pm$ 2.35	64.22 $\pm$ 1.35	47.37 $\pm$ 2.10
SVM	56.48 $\pm$ 1.78	69.76 $\pm$ 1.01	51.77 $\pm$ 1.98
SVM ^{perf}	46.70 $\pm$ 0.57	43.24 $\pm$ 2.84	30.83 $\pm$ 2.13
ML-KNN	49.29 $\pm$ 1.85	64.24 $\pm$ 1.57	47.56 $\pm$ 2.25
reverseMLL	23.29 $\pm$ 5.56	64.13 $\pm$ 1.55	47.63 $\pm$ 1.99
label-transfer	45.52 $\pm$ 1.69	53.70 $\pm$ 1.38	40.47 $\pm$ 1.98
Tagprop	51.06 $\pm$ 1.91	58.98 $\pm$ 2.41	43.47 $\pm$ 2.80
IAF ₁	58.24 $\pm$ 0.98	70.52 $\pm$ 0.94	52.38 $\pm$ 1.89
IAP ₅	60.07 $\pm$ 1.11	71.08 $\pm$ 0.76	52.77 $\pm$ 1.77
IAR ₅	4.69 $\pm$ 0.12	69.13 $\pm$ 1.25	51.26 $\pm$ 2.36

4.5.3.2 Experimental Setting

According to the results in Section 4.5.2, I particularly concentrate on optimizing the example-based measures for evaluating image annotation tasks in the following experiments. These measures are also widely employed in the literature [12, 120, 126, 127]. Following the work [12], k is set to 5 since the average number of keywords in all datasets is around 5, i.e., **IAP₅** and **IAR₅**. For nearest neighbor based methods (κ NN, ML-KNN, label-transfer, and Tagprop), κ is tuned in the grid of [1, 3, 5, 10, 20, 50, 100, 200, 500]. the termination condition parameter $\epsilon = 0.1$ is fixed for the proposed methods. The rest of settings are same as Section 4.5.2.

4.5.3.3 Experimental Results

For the concise representation, the measures discussed in the following of this section are all example-based. Experimental results on three datasets – UIUC-Sport, LabelMe and NUS-WIDE-Object – are shown in Table 4.5, 4.6 and 4.7, respectively. One can observe that the proposed method for optimizing F₁ score consistently outperforms all the baseline methods evaluated on the three performance measures in Table 4.1. The specially designed methods for optimizing macro-averaging F₁ such as SVM^{perf} and reverseMLL perform worse than the proposed methods because they optimize macro-averaging measures instead of example-based measures as the proposed methods do. Moreover, they

Table 4.6: The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on LabelMe by five-fold cross validation. The best results are in bold.

Method	F ₁	Prec@5	Rec@5
κ NN	39.55 $\pm$ 0.95	53.00 $\pm$ 1.37	44.59 $\pm$ 1.46
SVM	47.85 $\pm$ 1.00	54.52 $\pm$ 0.93	46.64 $\pm$ 1.53
SVM ^{perf}	42.79 $\pm$ 1.14	43.91 $\pm$ 1.17	36.11 $\pm$ 1.52
ML-KNN	39.46 $\pm$ 0.56	52.00 $\pm$ 1.10	43.84 $\pm$ 1.06
reverseMLL	12.87 $\pm$ 8.60	36.71 $\pm$ 17.82	30.05 $\pm$ 15.41
label-transfer	39.10 $\pm$ 0.98	41.00 $\pm$ 1.02	36.25 $\pm$ 0.93
Tagprop	39.32 $\pm$ 0.52	38.47 $\pm$ 0.83	33.53 $\pm$ 0.75
IAF ₁	51.28 $\pm$ 1.23	56.78 $\pm$ 1.38	48.31 $\pm$ 1.43
IAP ₅	51.05 $\pm$ 1.48	56.71 $\pm$ 1.54	48.13 $\pm$ 1.58
IAR ₅	1.81 $\pm$ 0.03	56.33 $\pm$ 1.51	47.91 $\pm$ 1.62

are much slower in the experiments than the proposed methods which is consistent with the complexity analysis in Section 4.4. The classical methods like κ NN and SVM are still the competitive methods after the proper model selection, e.g., five-fold cross validation for parameter selection, but they are worse than the proposed methods. The performances depend on datasets. SVM performs better than κ NN on UIUC-Sport and LabelMe, while κ NN is better than SVM on NUS-WIDE-Object. For fair comparison, the prior knowledge is not used to cover the different aspects of image contents so that all the methods are evaluated in the same setting. In this case, label transfer and TagProp perform poorly on all three datasets.

Another observation is that optimizing Rec@5 may not be helpful to get a good prediction of images since F₁ score of optimizing Rec@5 is much worse than those of optimizing F₁ and Prec@5, but the rest of measures are comparable. This is partially due to the extreme sparsity of keywords in each image. Optimizing Rec@5 leads to the situation that more keywords are preferred to be labeled as +1 in the training procedure. But actually, the average number of keywords in the testing image is small as shown in Figure 4.4(a)-(c). This contributes to the poor performance of F₁. Optimizing Prec@*k* however is the different case in which the main concern is the predicted keywords labeled as +1 in the testing image, so that the learned model prefers to label the top 5 keywords in an image correctly. Together with the sparsity of keywords in an image around 5, optimizing Prec@5 can obtain good results on all datasets and performance measures.

Table 4.7: The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on NUS-WIDE-Object by five-fold cross validation. The best results are in bold.

Method	F ₁	Prec@5	Rec@5
κ NN	7.88 $\pm$ 0.85	15.08 $\pm$ 0.32	12.90 $\pm$ 0.18
SVM	7.38 $\pm$ 0.44	8.87 $\pm$ 0.21	7.82 $\pm$ 0.29
SVM ^{perf}	8.83 $\pm$ 0.21	6.63 $\pm$ 0.33	5.58 $\pm$ 0.30
ML-KNN	0.68 $\pm$ 0.29	14.43 $\pm$ 0.34	12.31 $\pm$ 0.31
reverseMLL	6.24 $\pm$ 1.47	10.32 $\pm$ 1.23	8.67 $\pm$ 1.16
label-transfer	7.12 $\pm$ 0.56	7.67 $\pm$ 0.69	6.67 $\pm$ 0.57
Tagprop	7.88 $\pm$ 0.85	9.58 $\pm$ 1.06	8.35 $\pm$ 1.00
IAF ₁	10.48 $\pm$ 0.46	16.35 $\pm$ 0.23	13.93 $\pm$ 0.35
IAP ₅	9.41 $\pm$ 0.52	11.28 $\pm$ 0.60	9.45 $\pm$ 0.72
IAR ₅	1.55 $\pm$ 0.01	12.28 $\pm$ 0.33	10.62 $\pm$ 0.32

Even though the proposed methods outperform the baseline methods, especially for optimizing F₁ score, it seems that the results have a bit deviation from the initial objective. By comparing methods IAF₁ and IAP₅, one can observe that optimizing the target measure may not be better than optimizing other measures. For example, IAP₅ is better than IAF₁ in terms of F₁ score on UIUC-Sport, while IAF₁ is better than IAP₅ for Prec@5 on LabelMe and NUS-WIDE-Object. To explain this phenomenon, one can take a look at the histograms of the frequency of images with respect to the number of keywords shown in Figure 4.4(a)-(c). One can observe that there are around 90% images which contain more than or equal to 5 keywords on UIUC-Sport, while it is less than 60% on LabelMe and NUS-WIDE-Object. I conjecture that the reason why IAP₅ performs better than IAF₁ on UIUC-Sport is due to the keyword distribution and the choice of k . Figure 4.5 gives the evidence that F₁ score of IAP _{k} decreases when k increases gradually on UIUC-Sport. As k approaches to 7 where near 60% of images have keywords more than or equal to k , F₁ score of IAP _{k} begins to be worse than that of IAF₁. These observations imply that the choice of k affects F₁ of IAP _{k} according to the keyword distribution. Under the same situation (for example, 60%), they demonstrate the similar results that optimizing one specific measure is able to improve this measure on the testing dataset.

As shown in Figure 4.4, the frequency of images with the specific number of keywords on Corel5K is very different from the other three datasets since there is at most five

Figure 4.5: The example-based F_1 score from IAP_k method by varying k on UIUC-Sport.

keywords in each image on Corel5K, but there may exist tens of keywords appeared in one image in other three datasets with the long tail distribution in the right-hand side. As shown in the previous experiments, the characteristics of Corel5K will affect $Prec@5$ and $Rec@5$. Experimental results on Corel5K are shown in Table 4.8. One can observe that IAF_1 still outperforms others in terms of F_1 score. Another observation is that optimizing the objective-guided measure can achieve the higher score of this measure

Table 4.8: The example-based measures (mean $\pm$ standard deviation, %) of comparing different methods on Corel5k by five-fold cross validation. The best results are in bold.

Method	F_1	$Prec@5$	$Rec@5$
κNN	9.32 ± 0.58	6.08 ± 0.42	9.45 ± 0.67
SVM	1.66 ± 0.04	0.89 ± 0.13	0.30 ± 0.26
SVM^{perf}	23.32 ± 0.77	22.41 ± 0.28	32.04 ± 0.46
ML-KNN	7.05 ± 0.42	25.32 ± 0.12	36.23 ± 0.45
reverseMLL	3.66 ± 0.99	9.21 ± 3.13	13.06 ± 4.67
label-transfer	28.40 ± 0.53	21.20 ± 0.25	29.89 ± 0.56
tagprop	22.72 ± 0.57	22.94 ± 0.63	32.95 ± 0.64
IAF_1	29.48 ± 0.69	28.77 ± 0.44	40.66 ± 0.57
IAP_5	19.13 ± 1.77	29.98 ± 0.49	42.38 ± 0.71
IAR_5	25.03 ± 0.85	29.78 ± 0.67	42.42 ± 1.07

climber,grass,hook,rock,rope,tree	ball,athlete,auditorium,grass,horse,mallet,roadbed,spectator,tree	athlete,building,oar,rowboat,sky,tree,water	athlete,bridge,house,plant,sailing boat,sky,water
climber,grass,hook,rock,rope,sky,tree	ball,athlete,auditorium,grass,horse,mallet,roadbed,tree	athlete,building,oar,rowboat,sky,tree,water	athlete,house,plant,sailing boat,sky,water
climber,plant,rock	athlete,grass,horse,mallet,tree	athlete,oar,rowboat,tree,water	athlete,sailing boat,sky,water
athlete,croquet,grass,mallet,plant,stuff,wicket	ball,athlete,auditorium,grass,horse,mallet,roadbed,tree	athlete,building,oar,rowboat,tree,water	athlete,sailing boat,sky,water

Figure 4.6: Example annotations on UIUC-sport dataset by different methods: ground truth, IAF₁, κ NN, and Tagprop from top to bottom.

than others. This is still consistent with the previous results since there is no long tail distribution affecting Prec@5 and Rec@5.

These observations imply that optimizing Prec@ k and Rec@ k may be affected by the characteristics of datasets, but optimizing F₁ can consistently obtain very competitive results on all settings by comparing with other methods. All these evidences show that optimizing a specific measure, especially for F₁ score, is able to improve this measure on the testing dataset. some annotation results on UIUC-Sport are listed by the proposed methods. Figure 4.6 and Figure 4.7 show the example annotations from IAF₁ and IAP₅ respectively compared with κ NN and Tagprop.

4.6 Conclusion

To address the issue that many image annotation methods neglect optimizing the objective-guided performance measures, in this Chapter, I attempt to optimize a variety of objective specific measures in a unified multi-label learning framework. I present a multilayer hierarchical structure of learning hypothesis for multi-label problems based on how a variety of loss functions with respect to objective-guided measures are defined. And then,

climber, hook, knapsack, rock, rope, sky	athlete, audience, battledore, door, floor, spectator, stuff, wall	ball, door, flowerpot, ground, parapet, plant, player, spectator, tree, wall	athlete, ball, grass, lawn, mallet, parterre, wicket
climber, hook, rock, rope, sky	athlete, battledore, floor, stuff, wall	ball, ground, player, tree, wall	athlete, ball, grass, mallet, wicket
athlete, ski, skier, sky, water	athlete, battledore, floor, net, wall	athlete, ball, climber, sky, tree	athlete, mallet, oar, plant, rowboat
climber, hook, knapsack, rock, rope	(basketball) frame, athlete, battledore, door, mirror	audience, ball, door, flowerpot, ground	athlete, ball, bench, plant, wicket

Figure 4.7: Example annotations on UIUC-sport dataset by different methods on the top 5 keywords: ground truth, IAP₅, κ NN, and Tagprop from top to bottom.

the unified learning framework is presented. The analysis reveals that macro-averaging measures are very sensitive to infrequent keywords, micro-averaging measures are time-consuming, and hamming measure is easily affected by skewed distributions. The experimental results on four image annotation datasets demonstrate that optimizing the objective-guided performance measure is able to improve this performance measure, especially for F_1 score, which consistently shows very competitive results over other three measures on all four datasets.

Chapter 5

Structured-Prediction-Based Feature Selection for Performance Measures

5.1 Introduction

In this Chapter, I first propose a generalized sparse regularizer for feature selection. After that, a unified feature selection framework is presented for general loss functions based on the proposed regularizer. Particularly, in this thesis, optimizing multivariate performance measures is studied in this framework. To the best of my knowledge, this is the first work to optimize multivariate performance measures for feature selection. Due to exponential number of constraints brought by non-smooth multivariate loss functions [13, 14] and exponential number of feature subset combinations [62], the resultant optimization problem is very challenging for high-dimensional data. To tackle this challenge, I propose a two-layer cutting plane algorithm, including *group feature generation* (see Section 5.5.1) and *group feature selection* (see Section 5.5.2), to solve this problem effectively and efficiently. Specifically, Multiple Kernel Learning (MKL) by cutting plane algorithm is proposed to deal with exponential size of constraints induced by multivariate losses. The main contributions of this Chapter are listed as follows.

- A unified feature selection framework is presented for general loss functions based on the proposed generalized sparse regularizer. To solve the resultant optimization problem, a two-layer cutting plane algorithm is proposed. MKL algorithm is presented. The implementation details and the convergence proof of the proposed two-layer cutting plane algorithm are also given.

- Connections to a variety of the state-of-the-art feature selection methods including SKM [33], NMMKL [36], l_1 -SVM [56], l_0 -SVM [57] and FGM [62] are discussed in details. By comparing with these methods, the advantages of the proposed methods are summarized as follows:
 - (i) The tradeoff parameter C in l_1 SVM [56] is too sensitive to be tuned properly since it controls both margin loss and the sparsity of $\mathbf{w}$. However, the proposed method alleviates this problem by introducing an additional parameter B to control the sparsity of $\mathbf{w}$. This separation makes parameter tuning for the proposed methods much easier than those of SKM [33] and l_1 SVM.
 - (ii) NMMKL [36] uses the similar parameter separation strategy, but it is intractable for this method to handle high-dimensional datasets, let alone optimize multivariate losses. The proposed method can readily optimize multivariate losses for high-dimensional problems.
 - (iii) FGM [62] is a special case of the proposed framework when optimizing square hinge loss with indicator variables in integer domain. The proposed framework is formulated in the real domain for general loss functions. In particular, I provide a natural extension of FGM for multivariate losses.
 - (iv) The proposed framework can be interpreted by l_0 -norm constraint, so it can be considered as one of l_0 methods. This gives another interpretation of the additional parameter B .
- Recall that Multiple-Instance Learning via Embedded instance Selection (MILES) [163], which transforms multiple instance learning (MIL) into a feature selection problem by embedding bags into an instance-based feature space and selecting the most important features, achieves state-of-the-art performance for multiple instance learning problems. Under the proposed unified feature selection framework, I extend MILES and study MIL for multivariate performance measure. This is seldom studied in MIL scenarios, but it is important for the real world applications of MIL tasks [164–167].

- Extensive experiments on several challenging and very high-dimensional real world datasets show that the proposed method yields better performance than the state-of-the-art feature selection methods, and outperforms SVM^{perf} using all features in terms of multivariate performance measures. The experimental results on the multiple instance dataset show that the proposed method achieves promising results.

The rest of the Chapter is organized as follows: I briefly review SVM^{perf} in Section 5.2, and then introduce the proposed generalized sparse regularizer in Section 5.3. In particular, I study the feature selection framework for multivariate performance measures, its algorithm and its application to multiple instance learning in Section 5.4, 5.5 and 5.7, respectively. Section 5.6 gives the analysis of connections to a variety of feature selection methods. The extensive empirical results are shown in Section 5.8. Finally, conclusive remarks are presented in the last section.

5.2 SVM for Multivariate Performance Measure

Given a training sample of input-output pairs $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ for $i = 1, \dots, n$ drawn from some fixed but unknown probability distribution with $\mathcal{X} \subseteq R^m$ and $\mathcal{Y} \in \{-1, +1\}$, the learning problem is treated as a multivariate prediction problem by defining the hypotheses $\bar{h} : \bar{\mathcal{X}} \rightarrow \bar{\mathcal{Y}}$ that map a tuple $\bar{\mathbf{x}} \in \bar{\mathcal{X}}$ of n feature vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ to a tuple $\bar{y} \in \bar{\mathcal{Y}}$ of n labels $\bar{y} = (y_1, \dots, y_n)$ where $\bar{\mathcal{X}} = \mathcal{X} \times \dots, \mathcal{X}$ and $\bar{\mathcal{Y}} \subseteq \{-1, +1\}^n$. The linear discriminative function of SVM^{perf} is defined as

$$\bar{h}_{\mathbf{w}}(\bar{\mathbf{x}}) = \arg \max_{\bar{y}' \in \bar{\mathcal{Y}}} f(\bar{\mathbf{x}}, \bar{y}') = \arg \max_{\bar{y}' \in \bar{\mathcal{Y}}} \sum_{i=1}^n y'_i \mathbf{w}^T \mathbf{x}_i, \quad (5.1)$$

where $\mathbf{w} = [w_1, \dots, w_m]^T$ is the weight vector.

To learn the hypothesis (5.1) from training data, large margin method is employed to obtain the good generalization performance by enforcing the constraints that the decision value of the ground truth labels $\bar{y}$ should be larger than any possible labels $\bar{y}' \in \bar{\mathcal{Y}} \setminus \{\bar{y}\}$, i.e., $f(\bar{\mathbf{x}}, \bar{y}) \geq f(\bar{\mathbf{x}}, \bar{y}') + \Delta(\bar{y}, \bar{y}')$, where $\Delta(\bar{y}, \bar{y}')$ is some type of multivariate loss functions (several instantiated losses are presented in Section 5.5.4). Structural SVMs [6, 14] are

proposed to solve the corresponding soft-margin case by 1-slack variable formula as,

$$\begin{aligned} & \min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \|\mathbf{w}\|_2^2 + C\xi \\ & \text{s.t. } \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y} : \mathbf{w}^T \sum_{i=1}^n (y_i - y'_i) \mathbf{x}_i \geq \Delta(\bar{y}, \bar{y}') - \xi, \end{aligned} \quad (5.2)$$

where C is a regularization parameter that trades off the empirical risk and the model complexity.

The optimization problem (5.2) is convex, but there is exponential size of constraints. Fortunately, this problem can be solved in polynomial time by adopting the sparse approximation algorithm of structural SVMs. As shown in [13], optimizing the learning model subject to one specific multivariate measure can really boost the performance of this measure.

5.3 Generalized Sparse Regularizer

In this Chapter, The focus is on minimizing the regularized empirical loss functional as

$$\min_{\mathbf{w}} \Omega(\mathbf{w}) + C\ell(\mathbf{w}), \quad (5.3)$$

where $\Omega(\cdot)$ is a regularization function and $\ell(\cdot)$ is any loss function, including multivariate performance measure losses.

Since l_2 -norm regularization is used in (5.2), the learned weight vector $\mathbf{w}$ is non-sparse, and so the linear discriminant function in (5.1) would involve many features for the prediction. As discussed in Section 2.1.4, selecting a small set of discriminative features is crucial to many real applications. In order to enforce the sparsity on $\mathbf{w}$, I propose a new sparse regularizer

$$\Omega(\mathbf{w}) = \min_{\mathbf{d} \in \mathcal{D}} \frac{1}{2} \sum_{j=1}^m \frac{|w_j|^p}{d_j},$$

where $\mathbf{d}$ is in the real domain of $\mathcal{D} = \{\mathbf{d} | \sum_{j=1}^m d_j = B, 0 \leq d_j \leq 1, \forall j = 1, \dots, m\}$, $p > 0$ and $B > 0$ are two parameters. The optimal solution of the new proposed regularizer should satisfy $w_j = 0$ if $d_j = 0$ since $|w_j|^p = 0$ with $p > 0$ induces $w_j = 0$, otherwise the objective value approaches to infinite. The l_1 -norm constraint $\sum_{j=1}^m d_j = B$ and

$0 \leq d_j \leq 1$ will force some d_j to be zero, so the corresponding w_j is zero, $\forall j = 1, \dots, m$. Hence, the parameter B is interpreted as a budget to control the sparsity of $\mathbf{w}$.

This regularizer is similar to SimpleMKL [35] with each feature corresponding to one kernel, but SimpleMKL is a special case of $\mathcal{D}$ with $B = 1$, which also can be interpreted by the quadratic variational formulation of l_1 norm [168]. However, it is different from l_1 when $B \neq 1$. To explain the difference, the problem (5.2) under the general framework (5.3) is considered. In the separable case, parameter C does not affect the optimum solution since the error $\xi = 0$. If l_1 norm is applied to replace l_2 in Problem (5.2), the sparsity of $\mathbf{w}$ will be fixed once optimal solution is reached. Hence, parameter B in $\mathcal{D}$ now can be considered as the only factor to enforce sparsity on $\mathbf{w}$. However, in the non-separable case where errors are allowed, parameter C will also influence the sparsity of $\mathbf{w}$, but B is expected to enforce the sparsity of $\mathbf{w}$ more explicitly when C becomes larger. This argument will be empirically justified in Section 5.8.1.

The learning algorithm with the proposed generalized sparse regularizer is formulated as

$$\min_{\mathbf{d} \in \mathcal{D}} \min_{\mathbf{w}} \frac{1}{2} \sum_{j=1}^m \frac{|w_j|^p}{d_j} + C\ell(\mathbf{w}). \quad (5.4)$$

This formulation is more general for feature selection. For the comprehensive understanding of the proposed generalized sparse regularizer, we give the following Lemma and its proof according to [29] for complete consideration.

Lemma 1. *If and only if $p \geq 2$, Problem (5.4) is jointly convex with respect to $\mathbf{w}$ and $\mathbf{d}$.*

Proof: One only needs to prove that, if $p \geq 2$, $g(w_j, d_j) = \frac{|w_j|^p}{d_j}$ where $d_j > 0$ is jointly convex with respect to w_j and d_j . The convexity of g in its domain is established when the following holds: $\nabla^2 g = \begin{bmatrix} \frac{2|w_j|^p}{d_j^3} & -\frac{p|w_j|^{p-1}}{d_j^2} \\ -\frac{p|w_j|^{p-1}}{d_j^2} & \frac{p(p-1)|w_j|^{p-2}}{d_j} \end{bmatrix} \succeq \mathbf{0} \Leftrightarrow \begin{bmatrix} 2|w_j|^2 & -p|w_j|d_j \\ -p|w_j|d_j & p(p-1)d_j^2 \end{bmatrix} \succeq \mathbf{0}$, which is equivalent to $\mathbf{v}^T \nabla^2 g \mathbf{v} \geq 0$ for any nonzero vector $\mathbf{v}$. *WLOG*, we assume $\mathbf{v} = [1 \ a]^T$ where a is any real number, then this condition is reduced to: $2|w_j|^2 - 2ap|w_j|d_j + a^2p(p-1)d_j^2 \geq 0 \Leftrightarrow 2\left(|w_j| - \frac{apd_j}{2}\right)^2 \geq \frac{a^2d_j^2p(2-p)}{2}$. This condition always holds when $p \geq 2$, which completes the proof. $\square$

In what follows, I focus on the convex formulation with $p = 2$. The proposed method specially caters for the case $p = 2$ for the ease of derivation. It seems the special case limits the power of the generalized regularizer, but I will show the relationships of the proposed method with a variety of the state-of-the-art feature selection methods in Section 5.6. And the analysis of the relationships demonstrates that the learning method for the proposed formulation with $p = 2$ already covers or surpasses many state-of-the-art methods. The generalized regularizer with $p > 2$ will be studied in the future work.

5.4 Feature Selection for Multivariate Performance Measures

To optimize the multivariate loss functions and learn a sparse feature representation simultaneously, I propose to solve the following jointly convex problem over $\mathbf{d}$ and $(\mathbf{w}, \xi)$ in the case of $p = 2$,

$$\begin{aligned} \min_{\mathbf{d} \in \mathcal{D}} \min_{\mathbf{w}, \xi \geq 0} & \frac{1}{2} \sum_{j=1}^m \frac{|w_j|^2}{d_j} + C\xi \\ \text{s.t. } & \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y} : \mathbf{w}^T \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) \mathbf{x}_i \geq \Delta(\bar{y}, \bar{y}') - \xi. \end{aligned} \quad (5.5)$$

The partial dual with respect to $(\mathbf{w}, \xi)$ is obtained by Lagrangian function $\mathcal{L}(\mathbf{w}, \xi, \alpha, \tau)$ with dual variables $\alpha \geq 0$ and $\tau \geq 0$ as follows:

$$\frac{1}{2} \sum_{j=1}^m \frac{|w_j|^2}{d_j} + C\xi - \tau\xi - \sum_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} \alpha_{\bar{y}'} \left(\mathbf{w}^T \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) \mathbf{x}_i - \Delta(\bar{y}, \bar{y}') + \xi \right).$$

As the gradients of Lagrangian function with respect to $(\mathbf{w}, \xi)$ vanish at the optimal points, one can obtain the KKT conditions: $w_j = d_j \sum_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} \alpha_{\bar{y}'} \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) x_{j,i}$ and $\sum_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} \alpha_{\bar{y}'} \leq C$. By substituting KKT conditions back to $\mathcal{L}(\mathbf{w}, \xi, \alpha, \tau)$, one can obtain the dual problem as

$$\min_{\mathbf{d} \in \mathcal{D}} \max_{\alpha \in \mathcal{A}} -\frac{1}{2} \sum_{\bar{y}'} \sum_{\bar{y}''} \alpha_{\bar{y}'} \alpha_{\bar{y}''} Q_{\bar{y}', \bar{y}''}^{\mathbf{d}} + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'}, \quad (5.6)$$

where $\Delta(\bar{y}, \bar{y}) = 0$, $\Delta(\bar{y}, \bar{y}') > 0$ if $\bar{y} \neq \bar{y}'$,

$$\begin{aligned} Q_{\bar{y}', \bar{y}''}^{\mathbf{d}} &= \sum_{j=1}^m d_j \left(\sum_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} \alpha_{\bar{y}'} \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) x_{j,i} \right)^2 \\ &= \sum_{j=1}^m \left(\sum_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} \alpha_{\bar{y}'} \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) x_{j,i} \sqrt{d_j} \right)^2 \\ &= \langle \mathbf{a}_{\bar{y}'}, \mathbf{a}_{\bar{y}''} \rangle, \end{aligned}$$

$\mathbf{a}_{\bar{y}'} = \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) (\mathbf{x}_i \odot \sqrt{\mathbf{d}})$, $b_{\bar{y}'} = \frac{1}{n} \Delta(\bar{y}, \bar{y}')$, and $\mathcal{A} = \{\alpha \mid \sum_{\bar{y}'} \alpha_{\bar{y}'} \leq C, \alpha \geq 0\}$. Problem (5.6) is a challenging problem because of the exponential size of α and high-dimensional vector $\mathbf{d}$ for high-dimensional problems.

5.5 Two-Layer Cutting Plane Algorithm

In this section, a two-layer cutting plane algorithm is proposed to solve Problem (5.6) efficiently and effectively. The two layers, namely group feature generation and group feature selection, will be described in Section 5.5.1 and 5.5.2, respectively. The two-layer cutting plane algorithm will be presented in Section 5.5.3 and 5.5.4.

5.5.1 Group Feature Generation

By denoting $S(\alpha, \mathbf{d}) = -\frac{1}{2} \sum_{\bar{y}'} \sum_{\bar{y}''} \alpha_{\bar{y}'} \alpha_{\bar{y}''} Q_{\bar{y}', \bar{y}''}^{\mathbf{d}} + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'}$, Problem (5.6) turns out to be

$$\min_{\mathbf{d} \in \mathcal{D}} \max_{\alpha \in \mathcal{A}} S(\alpha, \mathbf{d}).$$

Since domains $\mathcal{D}$ and $\mathcal{A}$ are nonempty, the function $S(\alpha^*, \mathbf{d})$ is closed and convex for all $\mathbf{d} \in \mathcal{D}$ given any $\alpha^* \in \mathcal{A}$, and the function $S(\alpha, \mathbf{d}^*)$ is closed and concave for all $\alpha \in \mathcal{A}$ given any $\mathbf{d}^* \in \mathcal{D}$, the saddle-point property: $\min_{\mathbf{d} \in \mathcal{D}} \max_{\alpha \in \mathcal{A}} S(\alpha, \mathbf{d}) = \max_{\alpha \in \mathcal{A}} \min_{\mathbf{d} \in \mathcal{D}} S(\alpha, \mathbf{d})$ holds [169].

Denote $\mathcal{F}_{\mathbf{d}}(\alpha) = -S(\alpha, \mathbf{d})$. The equivalent optimization problems are obtained as

$$\min_{\alpha \in \mathcal{A}} \max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha) \text{ or } \min_{\alpha \in \mathcal{A}, \gamma} \gamma : \gamma \geq \mathcal{F}_{\mathbf{d}}(\alpha), \forall \mathbf{d} \in \mathcal{D}. \quad (5.7)$$

Cutting plane algorithm [63] could be used here to solve this problem. Since $\max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha) \geq \mathcal{F}_{\mathbf{d}^t}(\alpha), \forall \mathbf{d}^t \in \mathcal{D}$, the lower bound approximation of (5.7) can be obtained by $\max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha) \geq$

$\max_{t=1,\dots,T} \mathcal{F}_{\mathbf{d}^t}(\alpha)$. Then one can minimize Problem (5.7) over the set $\{\mathbf{d}^t\}_{t=1}^T$ by,

$$\min_{\alpha \in \mathcal{A}} \max_{t=1,\dots,T} \mathcal{F}_{\mathbf{d}^t}(\alpha) \text{ or } \min_{\alpha \in \mathcal{A}, \gamma} \gamma : \gamma \geq \mathcal{F}_{\mathbf{d}^t}(\alpha), \forall t=1,\dots,T \quad (5.8)$$

As from [170], such cutting plane algorithm can converge to a robust optimal solution within tens of iterations with the exact worst-case analysis. Specifically, for a fixed α^t , the worst-case analysis can be done by solving,

$$\mathbf{d}^t = \arg \max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha^t), \quad (5.9)$$

which is referred to as the group generation procedure. Even though Problem (5.8) and (5.9) cannot be solved directly due to the exponential size of α , I will show that they are readily solved in Section 5.5.2 and Section 5.5.4, respectively.

5.5.2 Group Feature Selection

By introducing dual variables $\mu = [\mu_1, \mu_2, \dots, \mu_T]^T \geq 0$, one can transform (5.8) to an MKL problem as follows,

$$\max_{\alpha \in \mathcal{A}} \min_{\mu \in \mathcal{M}_T} -\frac{1}{2} \sum_{\bar{y}'} \sum_{\bar{y}''} \alpha_{\bar{y}'} \alpha_{\bar{y}''} \left(\sum_{t=1}^T \mu_t Q_{\bar{y}', \bar{y}''}^{\mathbf{d}^t} \right) + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'}, \quad (5.10)$$

where $\mathcal{M}_T = \{\sum_{t=1}^T \mu_t = 1, \mu_t \geq 0, \forall t = 1, \dots, T\}$.

However, due to the exponential size of α , the complexity of Problem (5.13) remains. In this case, state-of-the-art multiple kernel learning algorithms [34, 35, 171] do not work any more. The following proposition shows that one can indirectly solve Problem (5.13) in the primal form.

Proposition 5. *The primal form of Problem (5.13) is*

$$\begin{aligned} \min_{\mathbf{w}_1, \dots, \mathbf{w}_T, \xi \geq 0} & \frac{1}{2} \left(\sum_{t=1}^T \|\mathbf{w}_t\|_2 \right)^2 + C\xi \\ \text{s.t. } & \xi \geq b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle, \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}. \end{aligned} \quad (5.11)$$

According to KKT conditions, the solution of (5.13) is

$$\mathbf{w}_t = \mu_t \sum_{\bar{y}'} \alpha_{\bar{y}'} \mathbf{a}_{\bar{y}'}^t \quad (5.12)$$

where μ_t is a dual value of the t^{th} constraint of (5.8).

Proof: Since the loss term $\Delta(\bar{y}', \bar{y}') = 0$ for all $\bar{y}' \in \mathcal{Y}$, one can equivalently transform Problem

$$\begin{aligned} \min_{\mathbf{w}_1, \dots, \mathbf{w}_T, \xi \geq 0} & \frac{1}{2} \left(\sum_{t=1}^T \|\mathbf{w}_t\|_2 \right)^2 + C\xi \\ \text{s.t.} & \xi \geq b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle, \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}, \end{aligned}$$

into the following optimization problem

$$\begin{aligned} \min_{\mathbf{w}_1, \dots, \mathbf{w}_T, \xi \geq 0} & \frac{1}{2} \left(\sum_{t=1}^T \|\mathbf{w}_t\|_2 \right)^2 + C\xi \\ \text{s.t.} & \xi \geq b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle, \forall \bar{y}' \in \bar{\mathcal{Y}}. \end{aligned}$$

By introducing a new variable $u \in \mathbb{R}$ and moving out summation operator from objective to be a constraint, one can obtain the equivalent optimization problem as

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} & \frac{1}{2} u^2 + C\xi \\ \text{s.t.} & \xi \geq b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle, \forall \bar{y}' \in \bar{\mathcal{Y}} \\ & \sum_{t=1}^T \|\mathbf{w}_t\| \leq u. \end{aligned}$$

one can further simplify above problem by introducing another variables $\rho_t \in \mathbb{R}$ such that $\|\mathbf{w}_t\| \leq \rho_t, \forall t = 1, \dots, T$, to be

$$\begin{aligned} \min_{\mathbf{w}, u, \rho, \xi \geq 0} & \frac{1}{2} u^2 + C\xi \\ \text{s.t.} & \xi \geq b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle, \forall \bar{y}' \in \bar{\mathcal{Y}} \\ & \sum_{t=1}^T \rho_t \leq u \\ & \|\mathbf{w}_t\| \leq \rho_t, \forall t = 1, \dots, T. \end{aligned}$$

It is known that for each t , $\|\mathbf{w}_t\| \leq \rho_t$ is a second-order cone constraint. Following the recipe of [29], the self-dual cone $\|\mathbf{v}_t\|_2 \leq \eta_t, \forall t = 1, \dots, T$ can be introduced to form the

Lagrangian function as follows

$$\begin{aligned} & \mathcal{L}(\mathbf{w}, \xi, u, \rho; \alpha, \tau, \gamma, \mathbf{v}, \eta) \\ &= \frac{1}{2}u^2 + C\xi - \sum_{\bar{y}'} \alpha_{\bar{y}'} \left(\xi - b_{\bar{y}'} + \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle \right) - \tau\xi \\ & \quad + \gamma \left(\sum_{t=1}^T \rho_t - u \right) - \sum_{t=1}^T (\langle \mathbf{v}_t, \mathbf{w}_t \rangle + \eta_t \rho_t), \end{aligned}$$

with dual variables $\alpha_t \in \mathbb{R}_+$, $\tau \in \mathbb{R}_+$, $\gamma \in \mathbb{R}_+$. The derivatives of the Lagrangian with respect to the primal variables have to vanish which leads to the following KKT conditions:

$$\begin{aligned} \mathbf{v}_t &= - \sum_{\bar{y}'} \alpha_{\bar{y}'} \mathbf{a}_{\bar{y}'}^t, \forall t = 1, \dots, T \\ C - \sum_{\bar{y}'} \alpha_{\bar{y}'} - \tau &= 0 \\ u &= \gamma \\ \gamma &= \eta_t, \forall t = 1, \dots, T \end{aligned}$$

By substituting all the primal variables with dual variables by above KKT conditions, one can obtain the following dual problem,

$$\begin{aligned} \max_{\alpha, \gamma} \quad & -\frac{1}{2}\gamma^2 + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'} \\ \text{s.t.} \quad & \left\| \sum_{\bar{y}'} \alpha_{\bar{y}'} \mathbf{a}_{\bar{y}'}^t \right\| \leq \gamma, \forall t = 1, \dots, T \\ & \sum_{\bar{y}'} \alpha_{\bar{y}'} \leq C, \alpha_{\bar{y}'} \geq 0, \forall \bar{y}' \in \bar{\mathcal{Y}} \end{aligned}$$

By setting $\theta = \frac{1}{2}\gamma^2$ and $\mathcal{A} = \{\sum_{\bar{y}'} \alpha_{\bar{y}'} \leq C, \alpha_{\bar{y}'} \geq 0, \forall \bar{y}' \in \bar{\mathcal{Y}}\}$, one can reformulate above problem as

$$\begin{aligned} \max_{\theta, \alpha \in \mathcal{A}} \quad & -\theta + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'} \\ \text{s.t.} \quad & \frac{1}{2}\alpha^T Q^t \alpha \leq \theta, \forall t = 1, \dots, T \end{aligned}$$

where $Q_{\bar{y}', \bar{y}''}^t = \langle \mathbf{a}_{\bar{y}'}^t, \mathbf{a}_{\bar{y}''}^t \rangle$. According to the property of self-dual cone [33], one can obtain the primal solution from its dual as $\mathbf{w}_t = -\mu_t \mathbf{v}_t = \mu_t \sum_{\bar{y}'} \alpha_{\bar{y}'} \mathbf{a}_{\bar{y}'}^t$ where μ_j is the dual

variable of the j^{th} quadratic constraint such that $\sum_{j=1}^m \mu_j = 1, \mu_j \in \mathbb{R}_+, \forall j = 1, \dots, m$. By constructing Lagrangian with dual variables μ with respect to θ , one can recover Problem

$$\max_{\alpha \in \mathcal{A}} \min_{\mu \in \mathcal{M}_T} -\frac{1}{2} \sum_{\bar{y}'} \sum_{\bar{y}''} \alpha_{\bar{y}'} \alpha_{\bar{y}''} \left(\sum_{t=1}^T \mu_t Q_{\bar{y}', \bar{y}''}^{\mathbf{d}^t} \right) + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'}, \quad (5.13)$$

where $\mathcal{M}_T = \{\sum_{t=1}^T \mu_t = 1, \mu_t \geq 0, \forall t = 1, \dots, T\}$. This completes the proof. $\square$

Here, define the regularization term as $\Omega(\bar{\mathbf{w}}) = \frac{1}{2} \left(\sum_{t=1}^T \|\mathbf{w}_t\|_2 \right)^2$ with $\bar{\mathbf{w}} = [\mathbf{w}_1, \dots, \mathbf{w}_T]^T$ and the empirical risk function as

$$R_{emp}(\bar{\mathbf{w}}) = \max \left(0, \max_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle \right), \quad (5.14)$$

which is a convex but non-smooth function w.r.t $\bar{\mathbf{w}}$. Then one can apply the bundle method [85] to solve this primal problem. Problem (5.13) is transformed as

$$\min_{\bar{\mathbf{w}}} \mathcal{J}(\bar{\mathbf{w}}) = \Omega(\bar{\mathbf{w}}) + C R_{emp}(\bar{\mathbf{w}}).$$

Since $R_{emp}(\bar{\mathbf{w}})$ is a convex function, its subgradient exists everywhere in its domain [172]. Suppose $\bar{\mathbf{w}}^k$ is a point where $R_{emp}(\bar{\mathbf{w}})$ is finite. One can formulate the lower bound according to the definition of subgradient,

$$\begin{aligned} R_{emp}(\bar{\mathbf{w}}) &\geq R_{emp}(\bar{\mathbf{w}}^k) + \langle \bar{\mathbf{w}} - \bar{\mathbf{w}}^k, \mathbf{p}^k \rangle \\ &= \langle \bar{\mathbf{w}}, \mathbf{p}^k \rangle + R_{emp}(\bar{\mathbf{w}}^k) - \langle \bar{\mathbf{w}}^k, \mathbf{p}^k \rangle \end{aligned}$$

where subgradient $\mathbf{p}^k \in \partial_{\bar{\mathbf{w}}} R_{emp}(\bar{\mathbf{w}}^k)$ is at $\bar{\mathbf{w}}^k$. In order to obtain $\mathbf{p}^k$, one needs to solve the following inference problem

$$\bar{y}^k = \arg \max_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} b_{\bar{y}'} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}'}^t \rangle \quad (5.15)$$

which is a problem of integer programming. The discussion of this problem is referred to Section 5.5.4. After that, one can obtain the subgradient $\mathbf{p}_t^k = -\mathbf{a}_{\bar{y}^k}^t$, so that $R_{emp}(\bar{\mathbf{w}}^k) = b_{\bar{y}^k} - \sum_{t=1}^T \langle \mathbf{w}_t, \mathbf{a}_{\bar{y}^k}^t \rangle = b_{\bar{y}^k} + \langle \bar{\mathbf{w}}^k, \mathbf{p}^k \rangle$.

Given the subgradient sequence $\mathbf{p}^1, \mathbf{p}^2, \dots, \mathbf{p}^K$, the tighter lower bound for $R_{emp}(\bar{\mathbf{w}})$ can be reformulated as follows,

$$R_{emp}(\bar{\mathbf{w}}) \geq R_{emp}^K(\bar{\mathbf{w}}) = \max \left(0, \max_{1 \leq k \leq K} \langle \bar{\mathbf{w}}, \mathbf{p}^k \rangle + q^k \right),$$

where $q^k = R_{emp}(\bar{\mathbf{w}}^k) - \langle \bar{\mathbf{w}}^k, \mathbf{p}^k \rangle = b_{\bar{y}^k}$. Following the bundle method [85], the criterion for selecting the next point $\bar{\mathbf{w}}^{K+1}$ is to solve the following problem,

$$\begin{aligned} \min_{\mathbf{w}_1, \dots, \mathbf{w}_T, \xi \geq 0} & \frac{1}{2} \left(\sum_{t=1}^T \|\mathbf{w}_t\|_2 \right)^2 + C\xi \\ \text{s.t. } & \xi \geq \langle \bar{\mathbf{w}}, \mathbf{p}^k \rangle + q^k, \forall k = 1, \dots, K. \end{aligned} \quad (5.16)$$

The following Corollary shows that Problem (5.16) can be easily solved by QCQP solvers, and the number of variables is independent of the number of examples.

Corollary 1. *In terms of Proposition 5, the dual form of Problem (5.16) is*

$$\begin{aligned} \max_{\alpha \in \mathcal{A}_K} \max_{\theta} & -\theta + \sum_{k=1}^K \alpha_k q^k \\ \text{s.t. } & \frac{1}{2} \left\| \sum_{k=1}^K \alpha_k \mathbf{p}_t^k \right\|_2^2 \leq \theta, \forall t = 1, \dots, T, \end{aligned} \quad (5.17)$$

where $\mathcal{A}_K = \{\sum_{k=1}^K \alpha_k \leq C, \alpha_k \geq 0, \forall k = 1, \dots, K\}$, and which is a QCQP problem with $T + 1$ constraints and $K + 1$ variables.

The proof of Corollary 1 follows the same derivation of Proposition 5 with $\mathbf{p}_t^k = -\mathbf{a}_{\bar{y}^k}^t$, $q^k = b_{\bar{y}^k}$ and the size of α_k as K . Consequently, the primal variables are recovered by $\mathbf{w}_t = -\mu_t \sum_k \alpha_k \mathbf{p}_t^k$.

Algorithm 3 Group Feature Selection

- 1: Input: $\bar{\mathbf{x}} = (\mathbf{x}_1, \dots, \mathbf{x}_n), \bar{y} = (y_1, \dots, y_n)$, an initial group set $\mathcal{W}$, ϵ , C
 - 2: $\bar{\mathcal{Y}} = \emptyset, k = 0$
 - 3: **repeat**
 - 4: $k = k + 1$
 - 5: Finding the most violated $\bar{y}'$
 - 6: Compute $\mathbf{p}^k$ and q^k
 - 7: $\bar{\mathcal{Y}} = \bar{\mathcal{Y}} \cup \{\bar{y}'\}$
 - 8: Solving Problem (5.17) over $\mathcal{W}$ and $\bar{\mathcal{Y}}$
 - 9: **until** ϵ -optimal
-

Let $\mathcal{J}_K(\bar{\mathbf{w}}) = \Omega(\bar{\mathbf{w}}) + CR_{emp}^K(\bar{\mathbf{w}})$, the ϵ -optimal condition in Algorithm 3 is $\min_{0 \leq k \leq K} \mathcal{J}(\bar{\mathbf{w}}^k) - \mathcal{J}_K(\bar{\mathbf{w}}^K) \leq \epsilon$. The convergence proof in [85] does not apply in this case as the Fenchel

dual of $\Omega(\bar{\mathbf{w}})$ fails to satisfy the strong convexity assumption if $K > 1$. As $K = 1$, Algorithm 3 is exactly the bundle method [85]. When $K \geq 2$, one can adapt the proof of Theorem 5 in [14] for the following convergence results.

Theorem 3. *For any $0 < C, 0 < \epsilon \leq 4R^2C$ and any training example $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$, Algorithm 3 converges to the desired precision ϵ after at most,*

$$\left\lceil \log_2 \left(\frac{\Delta}{4R^2C} \right) \right\rceil + \left\lceil \frac{16R^2C}{\epsilon} \right\rceil$$

iterations. $R^2 = \max_{\mathbf{d}, \bar{y}'} \left\| \frac{1}{n} \sum_{i=1}^n (y_i - y'_i)(\mathbf{x}_i \odot \sqrt{\mathbf{d}^T}) \right\|^2$, $\Delta = \max_{\bar{y}} \Delta(\bar{y}', \bar{y})$ and $\lceil \cdot \rceil$ is the integer ceiling function.

Proof: For a given set $\mathcal{W}_T$, the dual objective of (5.8) can be reformulated as

$$\max_{\alpha \in \mathcal{A}} \min_{\mathbf{d} \in \mathcal{W}_T} \Theta_{\mathbf{d}}(\alpha) = -\frac{1}{2} \sum_{\bar{y}'} \sum_{\bar{y}''} \alpha_{\bar{y}'} \alpha_{\bar{y}''} Q_{\bar{y}', \bar{y}''}^{\mathbf{d}} + \sum_{\bar{y}'} \alpha_{\bar{y}'} b_{\bar{y}'}.$$

Since there are T constrained quadratic problems, one can consider each $\mathbf{d} \in \mathcal{W}_T$ at one time as $\max_{\alpha \in \mathcal{A}} \Theta_{\mathbf{d}}(\alpha)$, where $Q^{\mathbf{d}}$ is positive semi-definite, and derivative $\partial \Theta_{\mathbf{d}}(\alpha) = \mathbf{b} - Q^{\mathbf{d}} \alpha$. The Lemma 2 in [14] states that a line search starting at α and along an ascent direction η with maximum step-size $C > 0$ improves the objective by at least $\max_{0 \leq \beta \leq C} \{ \Theta_{\mathbf{d}}(\alpha + \beta \eta) - \Theta_{\mathbf{d}}(\alpha) \} \geq \frac{1}{2} \min \left\{ C, \frac{\partial \Theta_{\mathbf{d}}(\alpha)^T \eta}{\eta^T Q^{\mathbf{d}} \eta} \right\} \partial \Theta_{\mathbf{d}}(\alpha)^T \eta$. If one considers sub-gradient descent method, the line search along the subgradient of objective is $\partial \Theta_{\mathbf{d}^*}(\alpha)$ where $\mathbf{d}^* = \min_{\mathbf{d} \in \mathcal{W}_T} \Theta_{\mathbf{d}}(\alpha)$. Therefore, the maximum improvement is

$$\begin{aligned} & \max_{0 \leq \beta \leq C} \{ \Theta_{\mathbf{d}^*}(\alpha + \beta \eta) - \Theta_{\mathbf{d}^*}(\alpha) \} \\ & \geq \frac{1}{2} \min \left\{ C, \frac{\partial \Theta_{\mathbf{d}^*}(\alpha)^T \eta}{\eta^T Q^{\mathbf{d}^*} \eta} \right\} \partial \Theta_{\mathbf{d}^*}(\alpha)^T \eta \\ & \geq \frac{1}{2} \min_{\mathbf{d} \in \mathcal{W}_T} \left\{ C, \frac{\partial \Theta_{\mathbf{d}}(\alpha)^T \eta}{\eta^T Q^{\mathbf{d}} \eta} \right\} \partial \Theta_{\mathbf{d}}(\alpha)^T \eta. \end{aligned} \quad (5.18)$$

One can see that it is a special case of [14] if $T = 1$. According to Theorem 5 in [14], for a newly added constraint $\hat{y}$ and some $\gamma_{\mathbf{d}} > 0$, one can obtain $\partial \Theta_{\mathbf{d}}(\alpha)^T \eta = \gamma_{\mathbf{d}}$ by setting the ascent direction $\eta_{\hat{y}} = 1$ for the newly added $\hat{y}$ and $\eta_{\bar{y}} = -\frac{1}{C} \alpha_{\bar{y}}$ for the others. Here, set $\gamma = \min_{\mathbf{d} \in \mathcal{W}_T} \gamma_{\mathbf{d}}$ so as to be the lower bound of $\partial \Theta_{\mathbf{d}}(\alpha)^T \eta, \forall \mathbf{d} \in \mathcal{W}_T$. In addition, the upper bound for $\eta^T Q^{\mathbf{d}} \eta \leq 4R^2, \forall \mathbf{d} \in \mathcal{W}_T$ can also be obtained by the fact

Algorithm 4 The Two-Layer Cutting Plane Algorithm

- 1: Input: $\bar{\mathbf{x}} = (\mathbf{x}_1, \dots, \mathbf{x}_n), \bar{y} = (y_1, \dots, y_n), \epsilon, C$
 - 2: $\mathcal{W} = \emptyset, t = 0$
 - 3: **repeat**
 - 4: $t = t + 1$
 - 5: Finding the most violated $\mathbf{d}^t$
 - 6: $\mathcal{W} = \mathcal{W} \cup \{\mathbf{d}^t\}$
 - 7: Call Algorithm 3 with inputs $(\bar{\mathbf{x}}, \bar{y}, \mathcal{W}, \epsilon, C)$
 - 8: **until** ϵ -optimal
-

that $\eta^T Q^{\mathbf{d}} \eta = Q_{\hat{y}, \hat{y}}^{\mathbf{d}} - \frac{2}{C} \sum_{\bar{y}'} \alpha_{\bar{y}'} Q_{\bar{y}', \hat{y}}^{\mathbf{d}} + \frac{1}{C^2} \sum_{\bar{y}'} \sum_{\bar{y}''} \alpha_{\bar{y}'} \alpha_{\bar{y}''} Q_{\bar{y}', \bar{y}''}^{\mathbf{d}} \leq R^2 + \frac{2}{C} C R^2 + \frac{1}{C^2} C^2 R^2 = 4R^2, \forall \mathbf{d} \in \mathcal{W}_T$. By substituting them back to (5.18), the similar result shows the increase of the objective is at least

$$\min \left\{ \frac{C\gamma}{2}, \frac{\gamma^2}{8R^2} \right\}.$$

Moreover, the initial optimality gap is at most $C\Delta$. The following proof is the same as that in Theorem 2 in Chapter 3. $\square$

5.5.3 The Proposed Algorithm

Algorithm 3 can obtain the ϵ -optimal solution for the original dual problem (5.8). By denoting $\mathcal{G}_d(\alpha) = \frac{1}{2} \|\sum_{k=1}^K \alpha_k \mathbf{p}^k\|_2^2 - \sum_{k=1}^K \alpha_k q^k$, the group feature generation layer can directly use the ϵ -optimal solution of the objective $\mathcal{G}_d(\alpha)$ to approximate the original objective $\mathcal{F}_d(\alpha)$. The two-layer cutting plane algorithm is presented in Algorithm 4. From the description of Algorithm 4, it is clear to see that groups are dynamically generated and augmented into active set $\mathcal{W}$ for group selection. The following remarks shows the advantages of the proposed two-layer cutting plane method compared to other optimization strategies:

Remark 1: Problem (5.16) is similar to Support Kernel Machine (SKM) [33] in which the multiple Gaussian kernels are built on random subsets of features, with varying widths. However, the proposed method can automatically choose the most violated subset of features as a group instead of a subset of random features. Such random features lead to a local optimum; while the proposed method could guarantee the ϵ -optimality stated in Theorem 3. However, due to the extra cost of computing nonlinear kernel, the current model are only implemented for linear kernel with learned subsets of features.

Remark 2: The original Problem (5.7) could be easily formulated as a QCQP problem with exponential size of variables α that needs to be optimized and huge number of base kernels in the quadratic term. Unfortunately, the standard MKL methods cannot handle Problem (5.7) even for a small dataset, let alone the standard QCQP solver. However, Corollary 1 makes it practical to solve a sequence of small QCQP problems directly using standard off-line QCQP solvers, such as Mosek. Note that state-of-the-art MKL solvers can also be used to solve the small QCQP problems, but they are not preferred because their solutions are less accurate than that of standard QCQP solvers, which can solve Problem (5.17) more accurately in this case.

In terms of the convergence proof of FGM in [62] and Theorem 3, one can obtain the following theorem to illustrate the approximation with an ϵ -optimal solution to the original problem.

Theorem 4. *After Algorithm 4 stops in a finite number of steps, the difference between optimal solution $(\mathbf{d}^*, \alpha^*)$ of Problem (5.13) and the solution $(\mathbf{d}, \alpha)$ of Algorithm 4 is $\mathcal{F}_{\mathbf{d}}(\alpha) - \mathcal{F}_{\mathbf{d}^*}(\alpha^*) \leq \epsilon$.*

Proof:

Given the Problem

$$\min_{\alpha \in \mathcal{A}} \max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha) \text{ or } \min_{\alpha \in \mathcal{A}, \gamma} \gamma : \gamma \geq \mathcal{F}_{\mathbf{d}}(\alpha), \forall \mathbf{d} \in \mathcal{D}, \quad (5.19)$$

one has the equivalent optimization problem as

$$\begin{aligned} \max_{\alpha \in \mathcal{A}, \gamma} \quad & -\gamma \\ \text{s.t.} \quad & \gamma \geq \mathcal{F}_{\mathbf{d}}(\alpha), \forall \mathbf{d} \in \mathcal{D}. \end{aligned}$$

The outer layer of Algorithm 2 can generate a sequence of configurations of $\mathbf{d}$ as $\{\mathbf{d}^1, \dots, \mathbf{d}^k\}$ after k iterations. In the k th iteration, the most violated constraint d^{k+1} is found in terms of α_k , so that $\mathcal{F}_{\mathbf{d}^{k+1}}(\alpha_k) = \max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha_k)$ according to Problem $\mathbf{d}^t = \arg \max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha^t)$. Hence, one can construct two sequences $\{\underline{\gamma}_k\}$ and $\{\bar{\gamma}_k\}$ such that

$$\underline{\gamma}_k = \max_{1 \leq t \leq k} \mathcal{F}_{\mathbf{d}^t}(\alpha_t) \quad (5.20)$$

$$\bar{\gamma}_k = \min_{1 \leq t \leq k} \mathcal{F}_{\mathbf{d}^{t+1}}(\alpha_t) = \min_{1 \leq t \leq k} \max_{\mathbf{d} \in \mathcal{D}} \mathcal{F}_{\mathbf{d}}(\alpha_t) \quad (5.21)$$

Suppose that one can solve $\min_{\alpha \in \mathcal{A}} \max_{1 \leq t \leq k} \mathbf{F}_{\mathbf{d}^t}(\alpha)$ exactly. Due to the equivalence to Problem (5.13), it means that one can obtain the exact solution of the problem (5.13). Based on this assumption, equation (5.20) can be further reformed as

$$\underline{\gamma}_k = \max_{1 \leq t \leq k} \mathcal{F}_{\mathbf{d}^t}(\alpha_t) = \min_{\alpha \in \mathcal{A}} \max_{1 \leq t \leq k} \mathcal{F}_{\mathbf{d}^t}(\alpha_t). \quad (5.22)$$

This turns out to be the same problem of FGM [62]. For self-completeness, the theorem is given as follows,

Theorem 5 ([62]). *Let (α^*, γ^*) be the globally optimal solution pair of Problem (5.19), sequences $\{\underline{\gamma}_k\}$ and $\{\bar{\gamma}_k\}$ have the following property*

$$\underline{\gamma}_k \leq \gamma_k \leq \bar{\gamma}_k. \quad (5.23)$$

As k increases, $\{\underline{\gamma}_k\}$ is monotonically increasing and $\{\bar{\gamma}_k\}$ is monotonically decreasing.

Based on above theorem, global optimal solution can be obtained after a finite number of iterations. However, the assumption of the accurate solution for (5.13) usually has no formal guarantee. We have already proven in Theorem 3 that the inner problem of Algorithm 2 can reach the desired precision ϵ after a finite number of iterations by Algorithm 1. Therefore, according to Algorithm 2, one can construct the following sequence

$$\underline{\gamma}'_k = \max_{1 \leq t \leq k} \mathcal{F}_{\mathbf{d}^t}(\alpha_t) \leq \min_{\alpha \in \mathcal{A}} \max_{1 \leq t \leq k} \mathcal{F}_{\mathbf{d}^t}(\alpha_t) + \epsilon. \quad (5.24)$$

By combining inequalities (5.23) and (5.24), one can obtain the following inequalities

$$\underline{\gamma}'_k - \epsilon \leq \underline{\gamma}_k \leq \gamma_k \leq \bar{\gamma}_k. \quad (5.25)$$

After a finite number of iterations, the global optimal solution is $\gamma^* = \underline{\gamma}_k = \gamma_k = \bar{\gamma}_k$. Hence, the solution of the Algorithm 2 may be not less than the lower bound $\underline{\gamma}'_k$ by ϵ . The proof is completed. $\square$

For a test pattern $\mathbf{x}$, the discriminant function can be obtained by $f(\mathbf{x}) = \langle \mathbf{w} \odot \tilde{\mathbf{d}}, \mathbf{x} \rangle$ where $\mathbf{w} = \sum_{i=1}^n \beta_i \mathbf{x}_i$, $\beta_i = \frac{1}{n} \sum_{k=1}^K \alpha_k (y_i - y_i^k)$, and $\tilde{\mathbf{d}} = \sum_{t=1}^T \mu_t \sqrt{\mathbf{d}^t}$. This leads to the faster prediction since only a few of the selected features are involved. After computing $\mathbf{p}^k$, the matrices of Problem (5.17) can be incrementally updated, so it can be done totally in $O(TK^2)$.

5.5.4 Finding the Most Violated $\bar{y}'$ and $\mathbf{d}$

Algorithm 3 and Algorithm 4 need to find the most violated $\bar{y}'$ and $\mathbf{d}$, respectively. This section will discuss how to obtain these quantities efficiently. Algorithm 3 needs to calculate the subgradient of the empirical risk function $R_{emp}^K(\bar{\mathbf{w}})$. Since $R_{emp}^K(\bar{\mathbf{w}})$ is a pointwise supremum function, the subgradient should be in the convex hull of the gradient of the decomposed functions with the largest objective. Here, one of these subgradients is taken by solving

$$\bar{y}^k = \arg \max_{\bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}} \Delta(\bar{y}', \bar{y}) - \sum_{i=1}^n (y_i - y'_i) v_i, \quad (5.26)$$

where $v_i = \sum_{t=1}^T \mathbf{w}_t^T (\mathbf{x}_i \odot \sqrt{\mathbf{d}^t})$. After obtaining $\bar{y}^k$, it is easy to compute $\mathbf{p}_t^k = -\frac{1}{n} \sum_{i=1}^n (y_i - y_i^k) (\mathbf{x}_i \odot \sqrt{\mathbf{d}^t})$ and $q^k = \frac{1}{n} \sum_{i=1}^n \Delta(\bar{y}^k, \bar{y})$.

For finding the most violated $\bar{y}'$, it depends on how to define the loss $\Delta(\bar{y}, \bar{y}')$ in Problem (5.26). One of the instances is the Hamming loss which can be decomposed and computed independently, i.e., $\Delta(\bar{y}, \bar{y}') = \sum_{i=1}^n \delta(y_i, y'_i)$, where δ is an indicator function with $\delta(y_i, y'_i) = 0$ if $y_i = y'_i$, otherwise 1. However, there are some multivariate performance measures which could not be solved independently. Fortunately, there are a series of structured loss functions, such as Area Under ROC (AUC), Average Precision (AP), ranking and contingency table scores and other measures listed in [13, 85, 173], which can be implemented efficiently in the proposed algorithms. In this thesis, several multivariate performance measures based on contingency table are studied as the showcases and their finding $\bar{y}^k$ could be solved in time complexity $O(n^2)$ [13].

Given the true labels $\mathbf{y}$ and predicted labels $\mathbf{y}'$, the contingency tables is defined as follows

	y=1	y=-1
y'=1	a	b
y'=-1	c	d

Based on this table, various performance measures for binary classification can be defined, such as F_1 -score, Precision/Recall @k, Precision/Recall Break-Even Point (PRBEP). The detailed descriptions of these performance measures are shown as follows:

F_1 -score: The F_β -score is a weighted harmonic average of Precision and Recall. According to the contingency table, one can obtain $F_\beta = \frac{(1+\beta^2)a}{(1+\beta^2)a+b+\beta^2c}$. The most common choice is $\beta = 1$. The corresponding balanced F_1 measure loss can be written as $\Delta_{F_1}(a, b, c, d) = 100(1 - F_1)$. Then, Algorithm 2 in [13] can be directly applied.

Precision/Recall@k: In search engine systems, most users scan only the first few links that are presented. In this situation, $\text{Prec}@k$ and $\text{Rec}@k$ measure the precision and recall of a classifier that predicts exactly k documents, i.e., $\text{Prec}@k = \frac{a}{a+b}$ and $\text{Rec}@k = \frac{a}{a+c}$, subject to $a+b = k$. The corresponding loss could be defined as $\Delta_{\text{Prec}@k} = 100(1 - \text{Prec}@k)$ and $\Delta_{\text{Rec}@k} = 100(1 - \text{Rec}@k)$. And the procedure of finding most violated $\mathbf{y}$ is similar to that of finding most violated constraints for F-score, while the only difference is keeping constraint $a + b = k$ and removing $a + b \neq k$.

Precision/Recall Break-Even Point (PRBEP): The Precision/Recall Break-Even Point requires that the precision and its recall are equal. According to above definition, one can see PRBEP only adds a constraint $a + b = a + c$, or $b = c$. The corresponding loss is defined as $\Delta_{\text{PRBEP}} = 100(1 - \text{PRBEP})$. Finding the most violated $\mathbf{y}$ should enforce the constraint $b = c$.

After t iterations in Algorithm 4, one transforms α in Problem (5.9) from the exponential size to a small size α^t . Now, finding the most violated $\mathbf{d}$ becomes

$$\begin{aligned}
 \mathbf{d}^t &= \arg \max_{\mathbf{d} \in \mathcal{D}} \mathcal{G}_{\mathbf{d}}(\alpha^t) \\
 &= \arg \max_{\mathbf{d} \in \mathcal{D}} \frac{1}{2} \left\| \sum_{k=1}^K \alpha_k^t \mathbf{p}^k \right\|_2^2 - \sum_{k=1}^K \alpha_k^t q^k \\
 &= \arg \max_{\mathbf{d} \in \mathcal{D}} \frac{1}{2} \left\| \frac{1}{n} \sum_{k=1}^K \alpha_k^t \sum_{i=1}^n (y_i - y_i^k) (\mathbf{x}_i \odot \sqrt{\mathbf{d}}) \right\|^2 \\
 &= \arg \max_{\mathbf{d} \in \mathcal{D}} \frac{1}{2n^2} \sum_{j=1}^m c_j^2 d_j
 \end{aligned} \tag{5.27}$$

where $c_j = \sum_{k=1}^K \alpha_k^t \sum_{i=1}^n (y_i - y_i^k) \mathbf{x}_{i,j}$. With the budget constraint $\sum_{i=1}^m d_i = B$ in $\mathcal{D}$, (5.27) can be solved by first sorting c_j^2 's in the descent order and then setting the first B numbers corresponding to d_j^t to 1 and the rest to 0. This takes only $O(m \log m)$ operations.

5.6 Relations To Existing Methods

In this section, I will discuss the relationships between the proposed method for multi-variate loss (5.5) and the state-of-the-art feature selection methods including SKM [33], NMMKL [36], l_1 -SVM [56], l_0 -SVM [57] and FGM [62]. It can be easily adapted to the general framework (5.4).

5.6.1 Connections to SKM and l_1 SVM

Let $\mathcal{D}_1 = \{\mathbf{d} | \sum_{j=1}^m d_j = 1, d_j \geq 0, \forall j = 1, \dots, m\}$ be in the real domain. One can observe that $\mathcal{D} = \mathcal{D}_1$ when $B = 1$. According to [35], one can transform Problem (5.5) in the special case of $B = 1$ to the following equivalent optimization problem,

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} & \frac{1}{2} \left(\sum_{j=1}^m |w_j| \right)^2 + C\xi \\ \text{s.t. } & \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y} : \mathbf{w}^T \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) \mathbf{x}_i \geq \Delta(\bar{y}, \bar{y}') - \xi. \end{aligned} \quad (5.28)$$

SKM [33] attempts to obtain the sparsity of $\mathbf{w}$ by penalizing the square of a weighted block l_2 -norm $(\sum_{j=1}^k \gamma_j \|\mathbf{w}_j\|_2)^2$ where k is the number of groups and $\mathbf{w}_j$ is the weight vector for the features in the j th group. The regularizer $(\sum_{j=1}^m |w_j|)^2$ used in (5.28) is the square of the l_1 norm $(\|\mathbf{w}\|_1)^2$, which is a special case of SKM when $k = m$ and $\gamma_j = 1$, i.e., each group contains only one feature. Minimizing the square of the l_1 -norm is very similar to l_1 -norm SVM [56] by setting $\Omega(\mathbf{w}) = \|\mathbf{w}\|_1$ with the non-negative (convex) loss function.

Regardless of l_1 -norm or the square of l_1 -norm, the parameter C is too sensitive to be tuned properly since it controls both margin loss and the sparsity of $\mathbf{w}$. However, the proposed method alleviates this problem by two parameters C and B which control margin loss and sparsity of $\mathbf{w}$, respectively. This separation makes parameter tuning of the proposed method easier than those of SKM and l_1 SVM.

5.6.2 Connection to NMMKL

Instead of directly solving Problem (5.28), one can formulate a more general problem (5.5) by introducing an additional budget parameter B , which directly controls the sparsity of

$\mathbf{w}$. The advantage is to make parameter tuning easily done since C is not sensitive to the sparsity of $\mathbf{w}$. This strategy is also used in NMMKL [36], but one feature corresponding to one base kernel makes NMMKL intractable for high-dimensional problems. The multivariate loss is even hard to be optimized by NMMKL since there are exponential dual variables in the dual form of NMMKL from the exponential number of constraints. What is more, the proposed method can readily optimize multivariate loss on high-dimensional data.

5.6.3 Connection to FGM

According to the work [174], one can reformulate Problem (5.28) as an equivalent optimization problem

$$\begin{aligned} \min_{\mathbf{d} \in \mathcal{D}_1} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} \sum_{j=1}^m d_j |w_j|^2 + C\xi \\ \text{s.t. } \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}: \quad & \frac{1}{n} \sum_{j=1}^m d_j w_j \sum_{i=1}^n (y_i - y'_i) \mathbf{x}_{j,i} \geq \Delta(\bar{y}, \bar{y}') - \xi. \end{aligned} \quad (5.29)$$

After the substitutions of $v_j = \sqrt{d_j} w_j, \forall j = 1, \dots, m$ and the general case of $\mathcal{D}$, one can obtain the following problem

$$\begin{aligned} \min_{\mathbf{d} \in \mathcal{D}} \min_{\mathbf{v}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{v}\|_2^2 + C\xi \\ \text{s.t. } \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}: \quad & \mathbf{v}^T \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) (\mathbf{x}_i \odot \sqrt{\mathbf{d}}) \geq \tilde{\Delta}(\bar{y}, \bar{y}') - \xi, \end{aligned} \quad (5.30)$$

where $\mathbf{v} = [v_1, \dots, v_m]^T$. After deriving Lagrangian dual problem of (5.30), one can observe that it is same as Problem (5.6). Problem (5.27) always finds the most violated $\mathbf{d}$ in the integer domain $\{0, 1\}^m$, so the solutions of the following problem solved by the proposed two-layer cutting plane algorithm is the same as the solutions of Problem (5.6)

$$\begin{aligned} \min_{\mathbf{d} \in \mathcal{D}_2} \min_{\mathbf{v}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{v}\|_2^2 + C\xi \\ \text{s.t. } \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \bar{y}: \quad & \mathbf{v}^T \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) (\mathbf{x}_i \odot \mathbf{d}) \geq \tilde{\Delta}(\bar{y}, \bar{y}') - \xi, \end{aligned} \quad (5.31)$$

where the integer domain $\mathcal{D}_2 = \{\mathbf{d} \mid \sum_{j=1}^m d_j \leq B, \mathbf{d} \in \{0, 1\}^m\}$. This formula can be equally derived as the extension of FGM for multivariate performance measures by defining the new hypotheses

$$\tilde{h}_{\mathbf{v}}(\mathbf{x}) = \arg \max_{\bar{\mathbf{y}}' \in \bar{\mathcal{Y}}} \sum_{i=1}^n y'_i (\mathbf{v} \odot \mathbf{d})^T \mathbf{x}_i, \quad (5.32)$$

where $\tilde{h}_{\mathbf{v}} : \bar{\mathcal{X}} \rightarrow \bar{\mathcal{Y}}$ and $\mathbf{d} \in \mathcal{D}_2$. It is not trivial to perform the extension of FGM to optimize multivariate loss because original FGM method [62] cannot directly apply to solve the exponential number of constraints. And the proposed domain of $\mathbf{d}$ is in real domain $\mathcal{D}$ which is more general than the integer domain $\mathcal{D}_2$ used in FGM and the proposed extension (5.31), even though the final solutions of (5.5) and (5.31) are the same.

5.6.4 Connection to l_0 SVM

The following Lemma indicates that the proposed formula can be interpreted by l_0 -norm constraint.

Lemma 2. *(5.31) is equivalent to the following problem*

$$\begin{aligned} \min_{\tilde{\mathbf{w}}, \xi \geq 0} \quad & \frac{1}{2} \|\tilde{\mathbf{w}}\|_2^2 + C\xi \\ \text{s.t.} \quad & \forall \bar{\mathbf{y}}' \in \bar{\mathcal{Y}} \setminus \bar{\mathbf{y}} : \tilde{\mathbf{w}}^T \frac{1}{n} \sum_{i=1}^n (y_i - y'_i) \mathbf{x}_i \geq \tilde{\Delta}(\bar{\mathbf{y}}, \bar{\mathbf{y}}') - \xi, \\ & \|\tilde{\mathbf{w}}\|_0 \leq B. \end{aligned} \quad (5.33)$$

Proof: Note, at the optimality of (5.30), WLOG, suppose $d_j = 0$, the corresponding v_j must be 0. Thus, $\|\mathbf{v}\|_0 \leq \|\mathbf{d}\|_0$. Let $\tilde{\mathbf{w}} = \mathbf{v} \odot \mathbf{d}$. One has $\|\tilde{\mathbf{w}}\|_0 = \|\mathbf{v} \odot \mathbf{d}\|_0 \leq \min\{\|\mathbf{v}\|_0, \|\mathbf{d}\|_0\} \leq \|\mathbf{d}\|_0 = \sum_{j=1}^m d_j \leq B$. Moreover, $\|\tilde{\mathbf{w}}\|_2^2 = \|\mathbf{v} \odot \mathbf{d}\|_2^2 = \|\mathbf{v}\|_2^2$ at the optimality. Therefore, the optimal solution of (5.30) is a feasible solution of (5.33). On the other hand, for the optimal $\tilde{\mathbf{w}}$ in (5.33), let $\mathbf{v} = \tilde{\mathbf{w}}$ and $d_i = \delta(\tilde{w}_i)$ where $\delta(t) = 1$ if $t \neq 0$; otherwise, 0. So, the optimal solution of (5.33) is a feasible solution of (5.30). $\square$

This gives another interpretation of parameter B from the perspective of l_0 -norm. Since l_0 -norm $\|\tilde{\mathbf{w}}\|_0$ represents the number of non-zero entries of $\tilde{\mathbf{w}}$, so B in the proposed method can be considered as the parameter which directly controls the sparsity of $\mathbf{w}$.

5.7 Multiple Instance Learning for Multivariate Performance Measures

Section 5.4 has already illustrated the proposed framework by optimizing multivariate performance measures for feature selection. In this section, I extend this approach to solve multiple instance learning problems which have been employed to solve a variety of learning problems, e.g., drug activity prediction [164], image retrieval [165], natural scene classification [175] and text categorization [167], but it is seldom optimized for multivariate performance measures in the literature. However, it is crucial to optimize the task specific performance measures, e.g., F score is widely considered as the most important evaluation criterion for a learning method in image retrieval.

Multi-instance learning was formally introduced in the context of drug activity prediction [164]. In this learning scenario, a bag is represented by a set of instances where each instance is represented by a feature vector. The classification label is only assigned to each bag instead of the instances in this bag. One can name a bag as a positive bag if there is at least one positive instance in this bag, otherwise it is called negative bag. The learning problem is to decide whether the given unlabeled bag is positive or not. By defining a similarity measure between a bag and an instance, Multiple-Instance Learning via Embedded instance Selection (MILES) [163] successfully transforms multiple instance learning into a feature selection problem by embedding bags into an instance-based feature space and selecting the most important features.

Before discussing the transformation in MILES, I first give the notations of multiple instance learning problem. Following the notations in [163], denote i th positive bags as $\mathbf{B}_i^+ = \{\mathbf{x}_{i,j}^+\}_{j=1}^{n_i^+}$ which consists of n_i^+ instances $\mathbf{x}_{i,j}^+, j = 1, \dots, n_i^+$. Similarly, the i th negative bags is denoted as $\mathbf{B}_i^- = \{\mathbf{x}_{i,j}^-\}_{j=1}^{n_i^-}$. All instances belongs to the same feature space $\mathcal{X}$. The number of positive bags and negative bags are ℓ^+ and ℓ^- , respectively. The instances in all bags are rearranged as $\{\mathbf{x}^1, \dots, \mathbf{x}^n\}$ where $n = \sum_{i=1}^{\ell^+} n_i^+ + \sum_{i=1}^{\ell^-} n_i^-$.

By considering each instance in the training bags as a candidate for target concepts, the embedded feature space is represented as

$$\hat{\mathbf{x}}_i = [s(\mathbf{x}^1, \mathbf{B}_i), \dots, s(\mathbf{x}^n, \mathbf{B}_i)]^T \in R^n, \quad (5.34)$$

where the similarity measure between the bag $\mathbf{B}_i$ and the instance $\mathbf{x}^k$ is defined as the most-likely-cause estimator

$$s(\mathbf{x}^k, \mathbf{B}_i) = \max_j \exp \left(-\frac{\|\mathbf{x}_{i,j} - \mathbf{x}^k\|^2}{2\sigma^2} \right). \quad (5.35)$$

It follows the intuition that the similarity between a concept and a bag is determined by the concept and the closest instance in this bag. The corresponding labels are constructed as follows: $\hat{y}_i = 1$ if $\mathbf{B}_i$ is a positive bag, otherwise $\hat{y}_i = -1$. For a given ℓ^+ positive bags and ℓ^- negative bags, one can form a new classification representation of the multiple instance learning problem as $\{\hat{\mathbf{x}}_i, \hat{y}_i\}_{i=1}^{\ell^+ + \ell^-}$. According to (5.34), the feature representations for all the $\ell_+ + \ell_-$ bags are constructed as

$$[\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_{\ell^+}, \hat{\mathbf{x}}_{\ell^++1}, \dots, \hat{\mathbf{x}}_{\ell^++\ell^-}] = \begin{bmatrix} s(\mathbf{x}^1, \mathbf{B}_1^+) & \dots & s(\mathbf{x}^1, \mathbf{B}_{\ell^-}^-) \\ s(\mathbf{x}^2, \mathbf{B}_1^+) & \dots & s(\mathbf{x}^2, \mathbf{B}_{\ell^-}^-) \\ \vdots & \ddots & \vdots \\ s(\mathbf{x}^n, \mathbf{B}_1^+) & \dots & s(\mathbf{x}^n, \mathbf{B}_{\ell^-}^-) \end{bmatrix}.$$

For each instance $\mathbf{x}^k$, the new feature representation corresponds to the values of the k th feature variable is

$$s(\mathbf{x}^k, \cdot) = [s(\mathbf{x}^k, \mathbf{B}_1^+), \dots, s(\mathbf{x}^k, \mathbf{B}_{\ell^+}^+), s(\mathbf{x}^k, \mathbf{B}_1^-), \dots, s(\mathbf{x}^k, \mathbf{B}_{\ell^-}^-)].$$

The advantage of this feature representation is to provides the useful information for separating the positive and negative bags if $\mathbf{x}^k$ achieves high similarity to some positive bags and low similarity to some negative bags. The linear discriminant function

$$\hat{y} = \text{sign}(\langle \mathbf{w}, \hat{\mathbf{x}} \rangle + b), \quad (5.36)$$

where $\mathbf{w}$ and b are the model parameters. The embedding induces a possible high-dimensional space when the number of instances in the training set is large. Since some instances may not be responsible for the label of the bags or might be similar to each other, many features are redundant or irrelevant, so MILES employs L_1 -SVM to select a subset of mapped features that is most relevant to the classification problem. However, L_1 -SVM cannot fulfill to obtain a high performance over the task-specific measures because it only focuses on optimizing zero-one loss function. The proposed Algorithm 4 is a

Algorithm 5 Learning a bag classifier

- 1: Input: positive bags $\{\mathbf{B}_i^+\}_{i=1}^{\ell^+}$, negative bags $\{\mathbf{B}_i^-\}_{i=1}^{\ell^-}$, C , and ϵ
- 2: Construct the embedding representation of training data

$$\{(\hat{\mathbf{x}}_i, \hat{y}_i)\}, \forall i = 1, \dots, \ell^+ + \ell^-$$

- 3: $\bar{\mathbf{x}} = [\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_{\ell^+ + \ell^-}]$ and $\bar{\mathbf{y}} = [\hat{y}_1, \dots, \hat{y}_{\ell^+ + \ell^-}]$
 - 4: call Algorithm 4 with arguments $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, C, \epsilon)$
 - 5: Output: parameters $\mathbf{w}$
-

Table 5.1: Datasets used in the experiments

Dataset	#classes	#features	#train points	#test points
News20.binary	2	1,355,191	11,997	7,999
URL1	2	3,231,961	20,000	20,000
Image	5	10,800	1,200	800
Sector	105	55,197	6,412	3,207
News20	20	62,061	15,935	3,993

natural alternative feature selection method for multi-variate performance measures. The proposed algorithm for multiple instance learning to optimize multivariate measures is shown in Algorithm 5.

According to Algorithm 5, one does not need the model parameter b since the structural SVM is irrelevant to the relative offset b , i.e., $\hat{y} = \arg \max_{y \in \{-1, +1\}} y \langle \hat{\mathbf{w}}, \hat{\mathbf{x}} \rangle$.

5.8 Experiments

In this Section, extensive experiments are conducted to evaluate the performance of the proposed method and state-of-the-art feature selection methods: 1) SVM-RFE [61]; 2) l_1 -SVM; 3) FGM [62]; 4) l_1 -bmm-F₁¹, which is l_1 regularized SVM for optimizing F₁ score by bundle method [85]. SVM-RFE and FGM use Liblinear software ² as the QP solver for their SVM subproblems. For l_1 -SVM, Liblinear software is used, which implements the state-of-the-art l_1 -SVM algorithm [56]. In addition to the comparison for

¹<http://users.cecs.anu.edu.au/~chteo/BMRM.html>

²<http://www.csie.ntu.edu.tw/~cjlin/liblinear/>

0-1 loss, experiments are also performed on image data for F1 measure. Furthermore, several specific measures on the contingency table are investigated on Text datasets by comparing with SVM^{perf} [13]. All the datasets shown in Table 5.1 are of high dimensions.

For convenience, the proposed two-layer cutting plane algorithm is denoted as FS_{multi}^{Δ} , where Δ represents different type of multivariate performance measures. Algorithm 4 is implemented in MATLAB for all the multivariate performance measures listed above, using Mosek as the QCQP solver for Problem (5.17) which yields a worse-case complexity of $O(KT^2)$. Removing inactive constraints from the working set [14] in the inner layer is employed for speedup the QCQP problem. Since the values of both K and T are much smaller than the number of examples n and its dimensionality m , the QCQP is very efficient as well as more accurate for large-scale and high-dimensional datasets. Furthermore, the codes simultaneously solve the primal and its dual form. So the optimal μ and α can be obtained after solving Problem (5.17).

5.8.1 Parameter Sensitivity Analysis

Before comparing FS_{multi}^{Δ} with other methods, I first conduct empirical studies for the parameter sensitivity analysis on *News20.binary*. The goal is to examine the relationships among parameters C and B , performance measures and the number of selected features with the range of C in $[0.1, 1, 10, 100] \times n$ and B in $[2, 5, 10, 50, 100, 150, 200, 250]$.

Figure 5.1(a-b) show the testing accuracy and F1 scores as well as the number of selected features by varying C and B . One can observe that the results are very sensitive to C when B is very small. This indicates that the l_1 model, which is equivalent to the proposed method in the case of $B = 1$, is vulnerable to the choice of C . On the other hand, the results are rather insensitive to C when B is large. Hence, the proposed method is less sensitive to C than l_1 model. One also can observe that the proposed method prefers a large C value for better performances. Figure 5.1(c-d) demonstrate the corresponding relationships among parameters B , C and the number of selected features of Figure 5.1(a-b). One observes that B and the number of selected features always exhibits a linear trend with a constant slope. Moreover, the slope remains the same when $C \geq 10$, but a small C will increase the slope. This means that, compared with B , parameter C has less influence on the sparsity of $\mathbf{w}$, and the learned feature selection

Figure 5.1: First row (a-b): Testing Accuracy and F_1 scores as well as the number of selected features of the proposed method $\text{FS}_{\text{multi}}^{\Delta}$ on *News20.binary* dataset by varying C and the number of selected features. Second row (c-d): The corresponding relationship among parameters B , C , and the number of selected features.

model becomes stabilized when $C \geq 10$. These empirical results are consistent to the discussions of parameter B in Section 5.3.

Since large C needs more iterations to converge according to Theorem 3, the compromise is to set C not too large and let B dominate the selection of features. According to these observations, one can safely fix C and study the results by varying B to compare with other methods in the following experiments.

Figure 5.2: Training time on different datasets

5.8.2 Time Complexity Analysis

The time complexity of $FS_{multi}^{F_1}$ is empirically studied by comparing with other methods. Two datasets *News20.binary* and *Image (Desert)* are used for illustration. The detailed settings are shown in Section 5.8.3 and Section 5.8.4, respectively. Figure 5.2 gives the training time over five different methods. On *News20.binary* dataset, the training time cannot be reported for l_1 -bmrm- F_1 since l_1 -bmrm- F_1 cannot terminate after more than two days with the maximum iteration 1000 and parameter $\lambda \in [10^{-7}, 10^2]$ due to the extremely high dimensionality. One can observe that the proposed methods are slower than l_1 -SVM, but much faster than SVM-RFE and l_1 -bmrm- F_1 . In addition, on *Image* dataset, when the termination condition with the relative difference between the objective and its convex linear lower bound lower than 0.1 is set, l_1 -bmrm- F_1 also cannot converge after the maximum iteration, which is consistent with the discussion in Appendix C of [85] that bundle method with l_1 regularizer cannot guarantee the convergence. This leads to the similar number of selected features (e.g., 98 in Figure 5.2(b)) even though λ is decreasing gradually.

These observations implies that the proposed two-layer cutting plane method needs less time for training with guaranteed convergence than bundle method. Moreover, our method can work on large scale and high dimensional data for optimizing user-specified measure, but bundle method cannot. As aforementioned, l_1 -bmrm- F_1 is much slower on the high dimensional datasets in the experiments, so its results in Section 5.8.4 can only be reported.

Figure 5.3: Testing accuracy on different datasets

5.8.3 Feature Selection for Accuracy

citeJoachims05 has proven that SVM_{multi}^{Δ} with Hamming loss, namely $\Delta_{Err}(\bar{y}, \bar{y}') = 2(b + c)$, is the same as SVM. In this subsection, the accuracy performances of FS_{multi}^{Δ} for Hamming loss function is evaluated, namely $FS_{multi}^{hamming}$ as well as other state-of-the-art feature selection methods. These methods are compared on two binary datasets, *News20.binary*³ and *URL1* in Table 5.1. Both datasets are used in [62], and they are already split into training and testing sets.

FGM and SVM-RFE are tested in the grid $C_{FGM} = [0.001, 0.01, 0.1, 1, 5, 10]$ and choose $C_{FGM} = 5$ which gives good performance for both FGM and SVM-RFE. This is the same as [62]. For $FS_{multi}^{hamming}$, the experiments are conducted by fixing $C_{FGM_{multi}}$ as $0.1 \times n$ for *URL1* and $1.0 \times n$ for *News20.binary*. The setting for budget parameter $B = [2, 5, 10, 50, 100, 150, 200, 250]$ for *News20.binary*, and $B = [2, 5, 10, 20, 30, 40, 50, 60]$ for *URL1*. The elimination scheme of features for SVM-RFE method can be referred to [62]. For l_1 -SVM, the results of different C values are reported so as to obtain different number of selected features.

Figure 5.3 reports testing accuracy on different datasets. The testing accuracy is comparable among different methods, but both $FS_{multi}^{hamming}$ and FGM can obtain better prediction performances than SVM-RFE in a small number (less than 20) of selected features on both *News20.binary* and *URL1*. These results show that the proposed method

³<http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets>

Figure 5.4: The sparsity of features of $FS_{multi}^{hamming}$ with varying B on different datasets. Each row bar with different color represents the different subset of features selected under current B , where the white region means the features are not selected.

with Hamming loss can work well on feature selection tasks especially when choosing only a few features. $FS_{multi}^{hamming}$ also performs better than l_1 -SVM on *News20.binary* in most range of selected features. This is possibly because l_1 models are more sensitive to noisy or redundant features on *News20.binary* dataset.

Figure 5.4 shows that the proposed method with the small B will select smaller number of features than the large B . One can also observed that most of features selected by the small B also appeared in the subset of features using the large B . This phenomenon can be obviously observed on *News20.binary*. This leads to the conclusion that $FS_{multi}^{hamming}$ can select the important features in the given datasets due to the insensitivity of parameter B . However, One can notice that not all the features in the selected subset of features with smaller B fall into that of subset of features with the large B , so the proposed method is non-monotonic feature selection. This argument is consistent with the test accuracy in Figure 5.3. *News20.binary* seems to be monotonic datasets from Figure 5.4, since $FS_{multi}^{hamming}$, FGM and SVM-RFE demonstrate similar performance. However, *URL1* is more likely to be non-monotonic, as the proposed method and FGM can do better than SVM-RFE. All the facts imply that the proposed method is comparable with FGM and SVM-RFE. And it also demonstrates the non-monotonic property for feature selection.

5.8.4 Feature Selection for Multiple Instance Learning

This subsection will demonstrate the specific multivariate performance measures are important to select features for real applications. In particular, F_1 measure (commonly used performance measure) is evaluated for the task of image categorization. Due to the success of transforming multiple instance learning into a feature selection problem by embedded instance selection, the same strategy is used in Algorithm 4.1 of [163] to construct a dense and high-dimensional dataset on a preprocessed image data ⁴. This dataset is used in [176] for multi-instance learning. It contains five categories and 2,000 images. Each image is represented as a bag of nine instances generated by the SBN method [175]. Each image bag is represented by a collection of nine 15-dimensional feature vectors. After that, following [163], the natural scene image categorization problem turns out to be a feature selection task to select relevant embedded instances for prediction. The *Image* dataset are split randomly with the proportion of 60% for training and 40% for testing (Table 5.1). The number of features in the original dataset is 15. After embedding, the number of feature becomes 10800. Since F_1 -score is used for performance metric, FS_{multi}^{Δ} is performed for F_1 -score, namely $FS_{multi}^{F_1}$ as well as other state-of-the-art feature selection methods. As mentioned above, FGM and $FS_{multi}^{hamming}$ have similar performances, the results of FGM will not be reported here. $FS_{multi}^{hamming}$ and FS_{multi}^{Δ} use the fixed $C = 10 \times n$.

In what follows, two settings of experiments are conducted. In the first part of the experiments, various feature selection methods are evaluated in the framework of MILES for multiple instance learning. This aims to empirically justify the proposed feature selection method is better than baseline feature selection methods. The baseline methods in the previous Section is reused in this setting. In the second part of the experiments, the performance of multiple instance learning with the proposed feature selection methods are evaluated by comparing to other state-of-the-art methods, such as Iterated-discrim APR [164], EM-DD [177], Citation-kNN [178], Bag MI-SVM [179], Bagging-EM-DD [180], and miGraph [181] ⁵. Notice that MILES [163] uses l1-SVM as

⁴http://lamda.nju.edu.cn/data_MIMLimage.ashx

⁵The packages used for all the compared methods are from <http://www.cs.cmu.edu/~juny/MILL/> and http://lamda.nju.edu.cn/code_miGraph.ashx. All the parameters are set by default. Ensemble method Bagging-EM-DD is implemented based on EM-DD in the MILL toolbox with the number of trials of bootstrap sampling is 5.

Figure 5.5: Testing F_1 scores on *Image* dataset.

the solver for instance selection, so the MILES for multiple instance learning is equivalent to the l_1 -SVM in the first part of experiments, as well as the proposed methods, $FS_{multi}^{hamming}$ and FS_{multi}^{Δ} .

In the first setting, the testing F_1 values of all methods on each category are reported in Figure 5.5. From Figure 5.5, one can observe that $FS_{multi}^{F_1}$ and $FS_{multi}^{hamming}$ achieve significantly improved performance over l_1 -SVM in term of F_1 -score especially when choosing less than 100 features. Moreover, SVM-RFE also outperforms l_1 -SVM on three categories out of five. This verifies that ℓ_1 penalty does not perform as well as ℓ_0 methods like $FS_{multi}^{F_1}$ and $FS_{multi}^{hamming}$ on dense and high-dimensional datasets. It is possibly because ℓ_1 -norm penalty is very sensitive to dense and noisy features. One also can observe that $FS_{multi}^{F_1}$ performs better than $FS_{multi}^{hamming}$ and SVM-RFE on four over five categories. l_1 -bmm-F₁ performs competitively but it is unstable and time-consuming as shown in Section 5.8.2. All these facts imply that directly optimizing F_1 measure is useful to boost F_1 performance measure, and the proposed $FS_{multi}^{F_1}$ is efficient and effective.

In the second setting, we evaluate the proposed methods $FS_{multi}^{hamming}$ and FS_{multi}^{Δ} by comparing with other multiple instance learning methods in terms of both accuracy and F_1 score. The experimental results are reported in Figure 5.6. According to accuracy, Figure 5.6(a) shows that MILES outperforms others, but comparable to $FS_{multi}^{hamming}$. However, this implies the advantages of $FS_{multi}^{hamming}$ for optimizing accuracy since MILES uses l_1 -SVM as the inner solver and the parameter $C = [0.01 : 0.05 : 1, 1.1 : 0.1 : 2]$ is tuned in finer grid so as to form Figure 5.5, while the parameter C in $FS_{multi}^{hamming}$ is fixed as $10 \times n$. Figure 5.6(b) demonstrates that $FS_{multi}^{F_1}$ outperforms other methods except on Sunset in terms of F_1 score. This implies that optimizing F_1 is beneficial for evaluating on this measure. By comparing two subfigures, it is easy to observe that the compared multiple instance methods vary much more in terms of F_1 than that of accuracy. Most of the compared methods are designed to obtain good accuracy, so it is suboptimal for F_1 score. All these observations justify that our proposed methods are beneficial when optimizing different performance measures.

(a) Accuracy

(b) F score

Figure 5.6: Testing Accuracy and F_1 scores on *Image* dataset by comparing the proposed methods with other multiple instance learning methods.

5.8.5 Multivariate Performance Measures for Document Retrieval

This subsection focuses on feature selection for different multivariate performance measures on imbalanced text data shown in Table 5.1. For multiclass classification problems, one vs. rest strategy is used. The comparing model is SVM^{perf} ⁶. Following [13], the same notation SVM_{multi}^{Δ} is used for different multivariate performance measures. The command used for training SVM^{perf} can work for different measures by $-l$ option⁷. The experiments search C_{perf} in the same range $[2^{-6}, \dots, 2^6]$ as in [13]. The one which demon-

⁶www.cs.cornell.edu/People/tj/svm_light/svm_perf.html

⁷`svm_perf_learn -c  $C_{perf}$  -w 3 -b 0 train_file train_model`

Table 5.2: The macro-average testing performance comparisons among different methods. The quantities in the parentheses represent won/lost of the current method comparing with FS_{multi}^{Δ} . The last column indicates the average number of features is actually used in the current method for a specific measure. The symbol '*' indicates the level of significance at 0.95 according to t-test applied to pairs of results over classes

Dataset	method	F_1	$Rec@2p$	$PRBEP$	#selected features
Sector	FS_{multi}^{Δ}	92.07	95.77	93.25	787.6/658.9/508.3
	$FS_{multi}^{hamming}$	84.99 (12/91)*	90.01 (0/71)*	85.54 (0/86)*	689.2
	SVM_{multi}^{Δ}	33.35 (1/104)*	95.52 (11/19)	91.24 (11/47)*	55,197
News20	FS_{multi}^{Δ}	77.56	91.21	81.46	1,301 / 1,186 / 931
	$FS_{multi}^{hamming}$	49.61 (0/20)*	66.32 (0/20)*	52.14 (0/20)*	485.1
	SVM_{multi}^{Δ}	55.53 (0/20)*	93.08 (16/2)	80.83 (6/11)	62,061

strates the best performance of SVM_{multi}^{Δ} to each multivariate performance measure is chosen for comparison. FS_{multi}^{Δ} and $FS_{multi}^{hamming}$ fix $C_{FGM_{multi}} = 0.1 \times n$ for *News20* except $5.0 \times n$ for *Sector*. For $Rec@k$, k as twice the number of positive examples is used, namely $Rec@2p$. The evaluation for this measure uses the same strategy to label twice the number of positive examples as positive in the test datasets, and then calculate $Rec@2p$.

Table 5.2 shows the macro-average of the performance over all classes in a collection in which both FS_{multi}^{Δ} and $FS_{multi}^{hamming}$ at $B = 250$ are listed. The improvement of FS_{multi}^{Δ} over $FS_{multi}^{hamming}$ and SVM_{multi}^{Δ} with respect to different B values are reported in Figure 5.7. From Table 5.2, FS_{multi}^{Δ} is consistently better than $FS_{multi}^{hamming}$ on all multivariate performance measures and two multiclass datasets. Similar results can be obtained comparing with SVM_{multi}^{Δ} , while the only exception is the measure $Rec@2p$ on *News20* where SVM_{multi}^{Δ} is a little better than FS_{multi}^{Δ} . The largest gains are observed for F_1 score on all two text classification tasks. This implies that a small number of features selected by FS_{multi}^{Δ} is enough to obtain comparable or even better performances for different measures than SVM_{multi}^{Δ} using all features.

From Figure 5.7, FS_{multi}^{Δ} consistently performs better than $FS_{multi}^{hamming}$ for all of the multivariate performance measures from the figures in the left-hand side. Moreover, the figures in the right-hand side show that the small number of features are good for F_1 measures, but poor for other measures. As the number of features increases, $Rec@2p$

Figure 5.7: The average performance improvement of FS_{multi}^{Δ} with varying B on different datasets.

Figure 5.8: Testing F_1 in terms of the average number of selected features on Sector and News20.

and *PRBEP* can approach to the results of $\text{SVM}_{multi}^{\Delta}$ and all curves become flat. The performance of *PRBEP* and *Rec@2p* is relatively stable when sufficient features are selected, but the proposed method can choose very few features for fast prediction. For F_1 measure, The method is consistently better than $\text{SVM}_{multi}^{\Delta}$, and the results show significant improvement over all range of B . This improvement may be due to the reduction of noisy or non-informative features. Furthermore, $\text{FS}_{multi}^{\Delta}$ can achieve better performance measures than $\text{FS}_{multi}^{hamming}$.

Different feature selection algorithms such as SVM-RFE and l_1 -SVM are also compared on *Sector* and *News20* in the same setting as the previous sections. The results in terms of F1 measure are reported in Figure 5.8. One can clearly observe that $\text{FS}_{multi}^{\Delta}$ outperforms l_1 -SVM on both datasets, and comparable or even better than SVM-RFE. For a small number of features, $\text{FS}_{multi}^{\Delta}$ can still demonstrate very good F1 measure.

5.9 Conclusion

In this Chapter, I propose a generalized sparse regularizer for feature selection, and the unified feature selection framework for general loss functions. I particularly study in details for multivariate losses. To solve the resultant optimization problem, a two-layer cutting plane algorithm was proposed. The convergence property of the proposed algorithm is studied. Moreover, connections to a variety of state-of-the-art feature selection methods are discussed in details. A variety of analyses by comparing with the various feature selection methods show that the proposed method is superior to others. Experimental results show that the proposed method is comparable with FGM and SVM-RFE and better than l_1 models on feature selection task, and outperforms SVM for multivariate performance measures on full set of features.

Chapter 6

Conclusions and Future Work

This thesis studies two important components in the framework of structured prediction model, that is feature engineering and performance evaluation. For feature engineering, two kind of feature selections are considered e.g., group of features in terms of templates in structured prediction problems and conventional feature selection in binary classification problems. For performance evaluation, I also consider to optimize various multivariate performance measures for either binary classification or structured prediction problem, specifically for image annotation with multiple labels. A wide range of experiments are conducted, including Chinese word segmentation, named entity recognition, dependency parsing, automatic image annotation, and image/document retrieval. In the future work, I will study the extensions of the proposed methods to other learning settings, such as structured output regression, semi-supervised learning, and optimizing other performance measures.

6.1 Summary of Contributions

In this thesis, three work has been done based on structured output prediction model, but they focus on different components of the structured output prediction framework. Similar to the diagram illustration in Figure 1.6, the brief summary of contributions of the proposed methods in this thesis is shown in Figure 6.1, where the main points of each proposed method is listed at the bottom of the diagram. The detailed contributions of each proposed method are summarized as follows:

Figure 6.1: The diagram illustrates the relationships of the proposed methods with three main components of the general structured prediction models, and the summary of contributions for each proposed method is illustrated at the bottom.

- The first work is to deal with feature engineering problem (template selection) in the structured output prediction. A multiple template learning paradigm is proposed to learn the weight of each template and the structured prediction model in the batch mode, simultaneously. An adapted cutting plane algorithm is proposed to tackle the MKL problem with exponential number of constraints, and the convergence is guaranteed. Detailed time complexities reveal that the proposed MTL^{struct} is much more efficient than OnlineMKL in the case of very sparse and high-dimensional data. And the training efficiency of the proposed MTL^{struct} is also very comparable with Structural SVM. Moreover, since MTL^{struct} can automatically remove redundant templates, the testing efficiency of MTL^{struct} is much improved over structured SVM and various CRFs without performance reduction. The proposed learning paradigm is further extended to solve structured output problems with p -block norm regularization. Experimental results on sequence labeling tasks

show competitive results when $p \in [1, 2)$; otherwise, the overall performance may degrade vastly.

- In the second work, optimizing multivariate performance measures is the main concern based on structured output prediction, especially for the automatic image annotation problems. I propose a unified multi-label learning framework which can optimize a variety of objective specific measures for image annotation. The analysis of these measures reveals that macro-averaging measures are very sensitive to infrequent keywords, which is rarely studied in literatures. Moreover, hamming measure is easily affected by skewed distributions which lead to be a bit over-estimated. Experiments on four image annotation demonstrate superior performance of the proposed method over baselines. I also observed that optimizing example-based F_1 obtains consistent good performance, while optimizing example-based Precision at k or Recall at k are sensitive to the number k and the keyword distribution of image annotation datasets.
- The third work considers to perform both feature selection and optimizing multivariate performance measures for binary classification problems based on structured output prediction. A unified feature selection framework with a generalized sparse regularizer is presented based on the structured output prediction. Particularly, optimizing multivariate performance measures is studied in this framework. this is the first work to optimize multivariate performance measures for feature selection. A two-layer cutting plane algorithm is proposed to solve this problem effectively and efficiently. Connections to a variety of the state-of-the-art feature selection methods indicate that the proposed framework covers or surpass many state-of-the-art methods. The proposed framework is further extended to optimizing multivariate performance measures for multi-instance learning problems. Extensive experiments on several challenging and very high-dimensional real world datasets show that the proposed method yields better performance than the state-of-the-art feature selection methods, and outperforms SVM^{perf} using all features in terms of multivariate performance measures. The experimental results on the multiple instance dataset show that the proposed method achieves promising results.

Figure 6.2: The diagram illustrates the relationships of the proposed methods with three main components of the general structured prediction models, and the future work based on the proposed method is illustrated at the bottom.

6.2 Future Work

In this thesis, we have done three work based on structured output prediction. Although they demonstrate some advantages over the baseline methods on a variety of real world applications, there still are some limitations or drawbacks which need to be solved in the future work. Figure 6.2 illustrates some of the future work which can either deal with the limitations of the current methods or further are extended to more general settings.

6.2.1 Structured Output Regression

In this thesis, I study structured output prediction for classification problems. However, there are many real world applications whose outputs are real values [30, 182, 183]. For example, many problems in computer vision are formulated as multivariate output representations, including visual pos estimation, segmentation or object and action recognition. The key difference between classification and regression is the loss function. For example, in structured output-associative regression (SOAR) [182], the square loss and the ϵ -insensitive loss are employed. Instead of generalized linear function, the ℓ_2 norm function is used for prediction [30, 183]. The difference of the two methods lies in either

linear feature map or joint feature map. As mentioned in [183], the representation with joint feature map is better than linear feature map because the dependencies between labels are captured. In addition, their input features are also interdependent and generally extracted from a given image by multiple feature extractors. In this scenario, the proposed multiple template learning for structured prediction could be naturally extended for structured regression problems. The learning algorithm shown in Chapter 3 does not require much more modifications if finding most violated outputs and the prediction problem are given for the regression case. In addition, the extension to structured output regression with ordered outputs [184] might also be interested as the future work.

6.2.2 Semi-supervised Multiple Template Learning

In this thesis, I mainly focus on structured prediction model in supervised setting. However, in the real world, there are many problems where the labeled data is hard to be obtained due to limited financial support or lack of domain annotator, but the unlabeled data are easily accessible. For example, for dependency parsing problem, the text document can be easily obtained from Internet, but the annotator must be trained for consistent and precise labeling. To explore the unlabeled data for structured prediction, several methods have been proposed in the literature, following manifold assumption [185], clustering assumption [174, 186] and co-training assumption [187]. These assumptions are generally used in the conventional semi-supervised learning for connecting the input to its optimal hypothesis. Extensions of these assumptions to structured prediction problem make the optimization problem even harder than that in conventional case due to the exponential number of constraints. The above mentioned methods are all proposed based on structured SVM, so they can be also extended directly to the case of features generated by multiple templates. In the further work, I will extend the proposed multi-template learning method for structured prediction problems in the semi-supervised setting.

6.2.3 Extensions for Other Multivariate Performance Measures

In the work of Chapter 5, I tackle the binary classification problems with multivariate performance measures based on the contingency table, such as F measure and PRBEP.

Table 6.1: Various performance measures are widely used in the literature for evaluating learning methods in different learning settings. The contingency-table based measures belong to a large family which has been mentioned in Chapter 4 and Chapter 5. The rest of performance measures are listed but not the least. The column “Concerned” stands for the measures used in this thesis.

Multivariate Performance Measure	Learning Setting	Concerned	Reference
Contingency-Table-Based measures	Binary Classification	Yes	[21, 22, 111]
Area Under roc Curve	Binary Classification	No	[13, 188]
Mean Average Precision	Ranking	No	[154, 189]
Normalize Discount Cumulative Gain	Ranking	No	[110, 112]
Ordinal Regression	Regression	No	[111]
Contingency-Table-Based measures	Multi-label Classification	Yes	[15, 20, 141]
Coverage	Multi-label Classification	No	[137]
Ranking loss	Multi-label Classification	No	[137, 139]
PRO loss	Multi-label Classification	No	[190]

And in the work of Chapter 4, the multivariate performance measures for multi-label classification are also based on contingency tables, such as macro-averaging F1 and micro-averaging F1. However, there exist many other multivariate performance measures. For binary classification or ranking, there are many other measures, such as normalize discount cumulative gain (NDCG) [110], ordinal regression [111], area under the Receiver Operating Characteristic curve (AUC) [13] and average precision [154]. These performance measures can be optimized in the proposed feature selection method FGM^{perf} . The two-layer cutting plane method are maintained, but the subproblems will be modified according to the above specific measures. For multi-label classification, there are many measures which do not belong to the contingency based measures, including PRO loss [190], Ranking loss [139], Coverage [137], and other measures as referred to [11]. The most difficult part for the proposed unified multi-label classification method on these measures is to find the most violation constraint. Various performance measures used in different learning settings are summarized in Table 6.1, where inference problems used in the literature can be readily adapted for feature selection problem as done in Chapter 5. But other measures such as Coverage and PRO loss might need special design for the inference problem in the proposed unified multi-label classification method. In the future work, I will attempt to design the exact or approximate (loss-augmented) inference for

optimizing these measures. It is also critical to make the inference fast for large scale image annotation problems.

6.2.4 Exploring the Generalized Sparse Regularizer

In Chapter 5, the generalized sparse regularizer is proposed. But mostly due to the difficulty in deriving the dual problem for the generalied sparse regularizer with $p > 2$. The work in Chapter 5 only study the special case of $p = 2$. This limitation restricts the power of the generalized sparse regularizer. As proved in Lemma 1, the formulation with $p > 2$ is jointly convex over $\mathbf{d}$ and $\mathbf{w}$. A naive way is to solve this problem by alternating method. The convergence of this method might be slow or cannot be guaranteed. In the future work, these problems will be first explored. And then, faster optimization method to tackle the general $p > 2$ will be designed.

Publication

- (i) **Qi Mao** and Ivor W. Tsang. A Feature Selection Method for Multivariate Performance Measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 35(9): 2051 - 2063, Sept 2013..
- (ii) **Qi Mao**, Ivor W. Tsang, and Shenghua Gao. Objective-guided Image Annotation. *IEEE Transactions on Image Processing (TIP)*, 22(4):1585-1597, April 2013.
- (iii) **Qi Mao** and Ivor W. Tsang. Efficient Multi-Template Learning for Structured Prediction. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 24(2): 248 - 261, Feb 2013.
- (iv) Chun-Wei Seah, Ivor W. Tsang, Yew-Soon Ong, and **Qi Mao**. Learning Target Predictive Function without Target Labels. *Proceedings of the IEEE International Conference on Data Mining (ICDM 2012)*, 2012.
- (v) Joey Tianyi Zhou, Sinno Jianlin Pan, **Qi Mao**, and Ivor W. Tsang. *Multi-view Positive and Unlabeled Learning*. The Asian Conference on Machine Learning (ACML2012), Journal of Machine Learning Research W & CPs. 2012.
- (vi) Jian Bo Yang, **Qi Mao**, Qiao Liang Xiang, Ivor W. Tsang, Kian Ming Adam Chai, and Hai Leong Chieu. *Domain Adaptation for Coreference Resolution: An Adaptive Ensemble Approach*. *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Conference on Natural Language Learning (EMNLP-CoNLL2012)*, 2012.

- (vii) Qiaoliang Xiang, **Qi Mao**, Kian Ming Chai, Hai Leong Chieu, Ivor W. Tsang, and Zhengdong Zhao. *A Split-Merge Framework for Comparing Clusterings*. Proceedings of the 29th International Conference on Machine Learning (ICML2012), 2012.
- (viii) **Qi Mao** and Ivor W. Tsang. *Optimizing Performance Measures for Feature Selection*. Proceedings of the IEEE International Conference on Data Mining (IEEE ICDM 2011), Vancouver, Canada, Dec 2011.
- (ix) **Qi Mao** and Ivor W. Tsang. *Parameter-Free Spectral Kernel Learning*. Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI 2010), Catalina Island, California, July 2010.

References

- [1] V. Vapnik, *The nature of statistical learning theory*. New York, Springer-Verlag, 1995.
- [2] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.
- [3] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. The MIT Press, 2012.
- [4] B. Schölkopf and A. Smola, *Learning with Kernels*. The MIT Press, 2002.
- [5] F. Peng, F. Feng, and A. McCallum, “Chinese segmentation and new word detection using conditional random fields,” in *COLING*, 2004.
- [6] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, “Large margin methods for structured and interdependent output variables,” *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.
- [7] F. Sha and F. Pereira, “Shallow parsing with conditional random fields,” in *HLT-NAACL*, 2003.
- [8] A. McCallum and W. Li, “Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons,” in *CoNLL*, 2003.
- [9] J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *ICML*, 2001.

REFERENCES

- [10] R. McDonald, K. Crammer, and F. Pereira, “Online large-margin training of dependency parsers,” in *ACL*, 2005.
- [11] G. Tsoumakas, I. Katakis, and I. Vlahavas, “Mining multi-label data,” *Data Mining and Knowledge Discovery Handbook*, vol. Springer, 2nd edition, 2010.
- [12] S. Gao, L. Chia, and I. W. Tsang, “Multi-layer group sparse coding - for concurrent image classification and annotation,” in *CVPR*, 2011.
- [13] T. Joachims, “A support vector method for multivariate performance measures,” in *ICML*, 2005.
- [14] T. Joachims, T. Finley, and C.-N. Yu, “Cutting-plane training of structural svms,” *Machine Learning*, vol. 77(1), pp. 27–59, 2009.
- [15] J. Petterson and T. Caetano, “Reverse multi-label learning,” in *NIPS*, 2010.
- [16] G. Tsoumakas and I. Vlahavas, “Random k-labelsets: an ensemble method for multilabel classification,” in *ECML*, 2007.
- [17] B. Taskar, C. Guestrin, and D. Koller, “Max-margin markov networks,” in *NIPS*, 2003.
- [18] A. F. T. Martins, N. A. Smith, E. Xing, P. M. Q. Aguiar, and M. A. T. Figueiredo, “Online learning of structured predictors with multiple kernels,” in *AISTATS*, 2011.
- [19] Q. Mao and I. W. Tsang, “Efficient multi-template learning for structured prediction,” *IEEE Transaction on Neural Networks and Learning Systems*, vol. 24(2), pp. 248–261, 2013.
- [20] Q. Mao, I. W. Tsang, and S. Gao, “Objective-guided image annotation,” *IEEE Transaction on Image Processing*, vol. 22(4), pp. 1585–1597, 2013.
- [21] Q. Mao and I. W. Tsang, “A feature selection method for multivariate performance measures,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35(9), pp. 2051–2063, 2013.

REFERENCES

- [22] Q. Mao and I. W. Tsang, “Optimizing performance measures for feature selection,” in *ICDM*, 2011.
- [23] B. Scholkopf and A. Smola, *Learning with Kernels*. MIT Press, 2002.
- [24] R. Memisevic, “Dual optimization of conditional probability models,” in *NIPS Workshop on Kernel Methods and Structured Domains*, 2006.
- [25] J. Weston and C. Watkins, “Multi-class support vector machines,” tech. rep., CSD-TR-98-04, 1998.
- [26] K. Crammer and Y. Singer, “On the algorithmic implementation of multiclass kernel-based vector machines,” *Journal of Machine Learning Research* 2, pp. 265–292, 2001.
- [27] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, 1995.
- [28] B. Taskar, *Learning structured prediction models: A large margin approach*. PhD thesis, Stanford University, 2004.
- [29] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Cambridge, UK., 2004.
- [30] J. Weston, B. Schölkopf, and O. Bousquet, “Joint kernel maps,” in *MPI Technical Report, TR-131*, 2003.
- [31] S. S. Keerithi, K. B. Duan, S. K. Shevade, and A. N. Poo, “A fast dual algorithm for kernel logistic regression,” *Machine Learning*, vol. 61, pp. 151–165, 2005.
- [32] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, “Learning the kernel matrix with semidefinite programming,” *Journal of Machine Learning Research*, vol. 5, pp. 27–72, 2004.
- [33] F. R. Bach, R. G. Lanckriet, and M. I. Jordan, “Multiple kernel learning, conic duality, and the SMO algorithm,” in *ICML*, 2004.
- [34] S. Sonnenburg, G. Rätsch, B. Schölkopf, and G. Rätsch, “Large scale multiple kernel learning,” *Journal of Machine Learning Research*, 2006.

REFERENCES

- [35] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, “SimpleMKL,” *Journal of Machine Learning Research*, vol. 9, pp. 2491–2521, 2008.
- [36] Z. Xu, R. Jin, I. King, and M. Lyu, “An extended level method for efficient multiple kernel learning,” in *NIPS*, 2009.
- [37] A. Zien and C. Ong, “Multiclass multiple kernel learning,” in *ICML*, 2007.
- [38] S. Ji, L. Sun, and J. Ye, “Multiple-label multiple kernel learning,” in *NIPS*, 2009.
- [39] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, “Lp-norm multiple kernel learning,” *Journal of Machine Learning Research*, vol. 12, pp. 953–997, 2011.
- [40] C. Cortes, M. Mohri, and A. Rostamizadeh, “ l_2 regularization for learning kernels,” in *UAI*, 2009.
- [41] C. Cortes, M. Mohri, and A. Rostamizadeh, “Generalization bounds for learning kernels,” in *ICML*, 2010.
- [42] Z. Xu, R. Jin, H. Yang, I. King, and M. Lyu, “Simple and efficient mutiple kernel learning by group lasso,” in *ICML*, 2010.
- [43] M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K. M uller, and A. Zien, “Efficient and accurate lp-norm multiple kernel learning,” in *NIPS*, 2009.
- [44] M. Kloft, U. Rückert, and P. Bartlett, “A unifying view of multiple kernel learning,” in *ECML PKDD*, 2010.
- [45] M. Varma and B. R. Babu, “More generality in efficient multiple kernel learning,” in *ICML*, 2009.
- [46] M. Girolami and S. Rogers, “Hierarchic bayesian models for kernel learning,” in *ICML*, 2005.
- [47] R. Tomioka and T. Suzuki, “Regularization strategies and empirical bayesian learning for MKL.” arXiv:1011.3090v2, 2011.
- [48] M. Gönen, “Bayesian efficient multiple kernel learning,” in *ICML*, 2012.

- [49] J. Bi, T. Zhang, and K. Bennett, “Column-generation boosting methods for mixture of kernels,” in *SIGKDD*, 2004.
- [50] C. Cortes, M. Mohri, and A. Rostamizadeh, “Ensembles of kernel predictors,” in *UAI*, 2011.
- [51] M. Gönen and E. Alpaydin, “Multiple kernel learning algorithms,” *JMLR*, vol. 12, pp. 2211–2268, 2011.
- [52] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff, “Embedded methods,” in *Feature Extraction: Foundations and Applications, Studies in Fuzziness and Soft Computing* (I. Guyon, S. Gunn, M. Nikraves, and L. A. Zadeh, eds.), no. 207, pp. 137–165, Springer, 2006.
- [53] J. Zhu, S. Rossett, T. Hastie, and R. Tibshirani, “1-norm support vector machine,” in *NIPS*, 2003.
- [54] G. M. Fung and O. L. Mangasarian, “A feature selection newton method for support vector machine classification,” *Computational Optimization and Applications*, vol. 28, pp. 185–202, 2004.
- [55] A. Y. Ng, “Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance,” in *ICML*, 2004.
- [56] G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin, “A comparison of optimization methods and software for large-scale l_1 -regularized linear classification,” *JMLR*, vol. 11, pp. 3183–3234, 2010.
- [57] J. Weston, A. Elisseeff, and B. Schölkopf, “Use of the zero-norm with linear models and kernel methods,” *JMLR*, vol. 3, pp. 1439–1461, 2003.
- [58] Z. Liu, F. Jiang, G. Tian, S. Wang, F. Sato, S. J. Meltzer, and M. Tan, “Sparse logistic regression with l_p penalty for biomarker identification,” *Statistical Applications in Genetics and Molecular Biology*, vol. 6, no. 1, 2007.
- [59] D. Lin, D. P. Foster, and L. H. Ungar, “A risk ratio comparison of l_0 and l_1 penalized regressions,” tech. rep., University of Pennsylvania, 2010.

REFERENCES

- [60] T. Zhang, “Analysis of multi-stage convex relaxation for sparse regularization,” *JMLR*, vol. 11, pp. 1081–1107, Mar 2010.
- [61] I. Guyou, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine Learning*, vol. 46, pp. 389–422, 2002.
- [62] M. Tan, L. Wang, and I. W. Tsang, “Learning sparse SVM for feature selection on very high dimensional datasets,” in *ICML*, 2010.
- [63] J. E. Kelley, “The cutting plane algorithm for solving convex programs,” *Journal of the Society for Industrial and Applied Mathematics*, vol. 8(4), pp. 703–712, 1960.
- [64] B. Taskar, C. Guestrin, and D. Koller, “Max-margin markov networks,” in *NIPS*, 2003.
- [65] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.
- [66] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.
- [67] A. McCallum, D. Freitag, and F. Pereira, “Maximum entropy markov models for information extraction and segmentation,” in *Proceedings of the 17th International Conference on Machine Learning*, 2000.
- [68] B. Taskar, P. Abbeel, and D. Koller, “Discriminative probabilistic models for relational data,” in *Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI02)*, 2002.
- [69] M. Richardson and P. Domingos, “Markov logic networks,” *Machine Learning*, 2005.
- [70] S. F. Chen and R. Rosenfeld, “A gaussian prior for smoothing maximum entropy models,” tech. rep., CMU-CS-99-108, Carnegie Mellon University,, 1999.

- [71] K. Gimpel and N. A. Smith, “Softmax-margin crfs: Training log-linear models with cost functions,” in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL*, 2010.
- [72] P. Pletscher, C. S. Ong, and J. M. Buhmann, “Entropy and margin maximization for structured output learning,” in *Proceedings of European Conference on Machine Learning*, 2010.
- [73] T. Hazan and R. Urtasun, “A primal-dual message-passing algorithm for approximated large scale structured prediction,” in *Advances in Neural Information Processing Systems*, 2010.
- [74] S. Kakade, Y. W. Teh, and S. Roweis, “An alternate objective function for markovian fields,” in *Proceedings of the International Conference on Machine Learning*, 2002.
- [75] C. Sutton and A. McCallum, “Piecewise pseudolikelihood for efficient training of conditional random fields,” in *Proceedings of the International Conference on Machine Learning*, 2007.
- [76] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 1999.
- [77] C. Huang and A. Darwiche, “Inference in belief networks: A procedural guide,” *International Journal of Approximate Reasoning*, vol. 11, pp. 1–158, 1994.
- [78] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” tech. rep., Gatsby Computational Neuroscience Unit, 2000.
- [79] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Constructing free energy approximations and generalized belief propagation algorithms,” tech. rep., Mitsubishi Electric Research Laboratories, 2004.
- [80] M. J. Wainwright and M. I. Jordan, “Graphical models, exponential families and variational inference,” *Foundations and Trends in Machine Learning*, vol. 1, pp. 1–305, 2008.

REFERENCES

- [81] Y. Altun, I. Tsochantaridis, and T. Hofmann, “Hidden markov support vector machines,” in *ICML*, 2003.
- [82] D. Bertsekas, *Nonlinear programming*. Athena Scientific, 1999.
- [83] M. Collins, “Discriminative training models for hidden markov models: Theory and experiments with perceptron algorithms,” in *EMNLP*, 2002.
- [84] S. Shalev-Shwartz, Y. Singer, and N. Srebro, “Pegasos: Primal estimated subgradient solver for svm,” in *Proceedings of the 24th International Conference on Machine Learning*, 2007.
- [85] C. H. Teo, S. V. N. Vishwanathan, A. Smola, and Q. Le, “Bundle methods for regularized risk minimization,” *Journal of Machine Learning Research*, pp. 311–365, 2010.
- [86] A. Smola, S. Vishwanathan, and Q. Le, “Bundle methods for machine learning,” in *Advances in Neural Information Processing Systems*, 2008.
- [87] P. Bartlett, M. Collins, B. Taskar, and D. McAllester, “Exponentiated gradient algorithms for large-margin structured classification,” in *Advances in Neural Information Processing Systems*, 2005.
- [88] M. Collins, T. C. X. Globerson, A. Koo, and P. Bartlett, “Exponentiated gradient algorithms for conditional random fields and max-margin markov networks,” *Journal of Machine Learning Research*, vol. 9, pp. 1775–1822, 2008.
- [89] B. Taskar, V. Chatalbashev, and D. Koller, “Learning associative markov networks,” in *Proceedings of the twenty-first International Conference on Machine Learning*, 2004.
- [90] B. Taskar, S. Lacoste-Julien, and M. I. Jordan, “Structured prediction, dual extra-gradient and bregman projections,” *Journal of Machine Learning Research*, vol. 7, pp. 1672–1653, 2006.
- [91] R. McDonald, F. Pereira, K. Ribarov, and J. Hajić, “Non-projective dependency parsing using spanning tree algorithm,” in *HLT/EMNLP*, 2005.

- [92] T. Finley and T. Joachims, “Supervised clustering with support vector machines,” in *ICML*, 2005.
- [93] L. Yao, S. Riedel, and A. McCallum, “Collective cross-document relation extraction without labeled data,” in *EMNLP*, 2010.
- [94] T. Finley and T. Joachims, “Training structural SVMs when exact inference is intractable,” in *ICML*, 2008.
- [95] K. Raghunathan, H. Lee, S. Rangarajan, N. Chambers, M. Surdeanu, D. Jurafsky, and C. Manning, “A multi-pass sieve for coreference resolution,” in *EMNLP*, 2010.
- [96] D. L. Vail, J. Lafferty, and M. M. Veloso, “Feature selection in conditional random fields for activity recognition,” in *IEEE/RSJ IROS*, 2007.
- [97] F. R. Bach, “Consistency of the group lasso and multiple kernel learning,” *Journal of Machine Learning Research*, pp. 1179–1225, 2008.
- [98] A. Rakotomamonjy, R. Flamary, G. Gasso, and S. Canu, “ ℓ_p - ℓ_q penalty for sparse linear and sparse multiple kernel multitask learning,” *IEEE Transaction on Neural Networks*, vol. 22, pp. 1307–1320, 2011.
- [99] R. Tomioka and T. Suzuki, “SpicyMKL,” *Arxiv preprint arXiv:0909.5026*, 2009.
- [100] M. Hu, Y. Chen, and J. T. Kwok, “Building sparse multiple-kernel svm classifiers,” *IEEE Transaction on Neural Networks*, vol. 20, pp. 827–839, 2009.
- [101] H. Yang, Z. Xu, I. King, and M. Lyu, “Efficient sparse generalized multiple kernel learning,” *IEEE Transaction on Neural Networks*, vol. 22, pp. 433–446, 2011.
- [102] J. López and J. R. Dorronsoro, “Simple proof of convergence of the SMO algorithm for different SVM variants,” *IEEE Transaction on Neural Networks and Learning Systems*, vol. 25, pp. 1142–1147, 2012.
- [103] F. Cai and V. Cherkassky, “Generalized SMO algorithm for svm-based multitask learning,” *IEEE Transaction on Neural Networks and Learning Systems*, vol. 23, pp. 997–1003, 2012.

REFERENCES

- [104] O. Dekel and Y. Singer, “Data-driven online to batch conversions,” in *NIPS*, 2005.
- [105] O. Dekel, “From online to batch learning cutoff-averaging,” in *NIPS*, 2008.
- [106] O. Shamir and T. Zhang, “Stochastic gradient descent for non-smooth optimization: convergence results and optimal averaging schemes.” CoRR, abs/1212.1824, 2012.
- [107] A. Rakhlin, O. Shamir, and K. Sridharan, “Making gradient descent optimal for strongly convex stochastic optimization.” CoRR, abs/1109.5647, 2011.
- [108] X. Zhang, A. Saha, and S. Vishwanathan, “Smoothing multivariate performance measures,” in *UAI*, 2011.
- [109] D. R. Musicant, V. Kumar, and A. Ozgur, “Optimizing f-measure with support vector machines,” in *Proceedings of the 16th International Florida Artificial Intelligence Research Society Conference*, 2003.
- [110] H. Valizadegan, R. Jin, R. Zhang, and J. Mao, “Learning to rank by optimizing ndcg measure,” in *NIPS*, 2009.
- [111] T. Joachims, “Training linear SVMs in linear time,” in *SIGKDD*, 2006.
- [112] Q. V. Le and A. Smola, “Direct optimization of ranking measures,” *JMLR*, vol. 1, pp. 1–48, 2007.
- [113] G. Stamou, J. Van, J. Z. Pan, G. Schreiber, and J. R. Smith, “Multimedia annotations on the semantic web,” *IEEE MultiMedia*, 2006.
- [114] C. Jin and C. Yang, “Integrating hierarchical feature selection and classifier training for multi-label image annotation,” in *SIGIR*, 2011.
- [115] J. Jeon, V. Lavrenko, and R. Manmatha, “Automatic image annotation and retrieval using cross-media relevance models,” in *SIGIR*, 2003.
- [116] D. Wang, S. Hoi, and Y. He, “Mining weakly labeled web facial images for search-based face annotation,” in *SIGIR*, 2011.

REFERENCES

- [117] C. Wang, L. Zhang, and H.-J. Zhang, “Learning to reduce the semantic gap in web image retrieval and annotation,” in *SIGIR*, 2008.
- [118] A. Makadia, V. Pavlvoic, and S. Kumar, “A new baseline for image annotation,” in *ECCV*, 2008.
- [119] C. Wang, D. Blei, and L. Fei-Fei, “Simultaneous image classification and annotation,” in *CVPR*, 2009.
- [120] Y. Wang and G. Mori, “Max-margin latent dirichlet allocation for image classification and annotation,” in *BMVC*, 2011.
- [121] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid, “Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation,” in *ICCV*, 2009.
- [122] A. Hanbury, “Analysis of keywords used in image understanding tasks,” in *Proceedings of the OntoImage Workshop*, 2006.
- [123] L. Chen, D. Xu, I. W. Tsang, and J. Luo, “Tag-based web photo retrieval improved by batch mode re-tagging,” in *CVPR*, 2010.
- [124] H. Wang, H. Huang, and C. Ding, “Image annotation using bi-relational graph of images and semantic labels,” in *CVPR*, 2011.
- [125] Y. Xiang, X. Zhou, Z. liu, T.-S. Chua, and C.-W. Ngo, “Semantic context modeling with maximal margin conditional random fields for automatic image annotation,” in *CVPR*, 2010.
- [126] A. Sun, S. S. Bhowmick, and J.-A. Chong, “Social image tag recommendation by concept matching,” in *ACM MM*, 2011.
- [127] Y. Song, Z. Zhuang, H. Li, Q. Zhao, J. Li, W.-C. Lee, and C. L. Giles, “Real-time automatic tag recommendation,” in *SIGIR*, 2008.
- [128] M.-L. Zhang and Z.-H. Zhou, “Ml-knn: A lazy learning approach to multi-label learning,” *Pattern Recognition*, vol. 40(7), pp. 2038–2048, 2007.

REFERENCES

- [129] S. H. Yang, J. Bian, and H. Zha, “Hybrid generative/discriminative learning for automatic image annotation,” in *UAI*, 2010.
- [130] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan, “Matching words and pictures,” *JMLR*, vol. 3, pp. 1107–1135, 2003.
- [131] S. Feng, R. Manmatha, and L. V, “Multiple bernoulli relevance models for image and video annotation,” in *CVPR*, 2004.
- [132] G. Carneiro, A. Chan, P. Moreno, and N. Vasconcelos, “Supervised learning of semantic classes for image annotation and retrieval,” *PAMI*, vol. 29(3), pp. 394–410, 2007.
- [133] C. Cusano, G. Ciocca, and R. Schettini, “Image annotation using svm,” in *SPIE*, 2004.
- [134] D. Grangier and S. Bengio, “A discriminative kernel-based model to rank images from text queries,” *PAMI*, vol. 30(8), pp. 1371–1384, 2008.
- [135] X. Liu, B. Cheng, S. Yan, J. Tang, T.-S. Chua, and H. Jin, “Label to region by bi-layer sparsity priors,” in *ACM MM*, 2009.
- [136] F. Kang, R. Jin, and R. Sukthankar, “Correlated label propagation with application to multi-label learning,” in *CVPR*, 2006.
- [137] R. E. Schapire and Y. Singer, “Boosttexter: a boosting-based system for text categorization,” *Machine Learning*, vol. 39, pp. 135–168, 2000.
- [138] Y. Yang, “An evaluation of statistical approaches to text categorization,” *JIR*, vol. 1, pp. 67–88, 1999.
- [139] A. Elisseeff and J. Weston, “A kernel method for multi-labelled classification,” in *NIPS*, 2001.
- [140] D. Parikh and K. Grauman, “Relative attributes,” in *ICCV*, 2011.
- [141] X. Zhang, T. Graepel, and R. Herbrich, “Bayesian online learning for multi-label and multi-variate performance measures,” in *AISTATS*, 2010.

- [142] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, pp. 41–75, 1997.
- [143] E. D. Andersen and A. D. Andersen, *The MOSEK interior point optimizer for linear programming: An implementation of the homogeneous algorithm*. Kluwer Academic Publishers, 2000.
- [144] C. H. Teo, Q. Le, A. Smola, and S. V. N. Vishwanathan, “A scalable modular convex solver for regularized risk minimization,” in *ACM SIGKDD*, 2007.
- [145] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, “Efficient projections onto the ℓ_1 -ball for learning in high dimensions,” in *ICML*, 2008.
- [146] N. Xue, “Chinese word segmentation as character tagging,” *Computational Linguistics and Chinese Language Processing*, vol. 8, pp. 29–48, 2003.
- [147] A. J. Viterbi, “Error bounds for convolutional codes and an asymptotically optimal decoding algorithm,” *IEEE Transaction on Information Theory*, pp. 260–269, 1967.
- [148] D. Wipf and S. Nagarajan, “Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions,” *Journal of Selected Topics in Signal Processing (Special Issue on Compressive Sensing)*, vol. 1, 2010.
- [149] J. Eisner, “Three new probabilistic models for dependency parsing: an exploration,” in *COLING*, 1996.
- [150] Y. J. Chu and T. H. Liu, “On the shortest arborescence of a directed graph,” *Science Sinica*, vol. 14, pp. 1396–1400, 1965.
- [151] T. Emerson, “The second international Chinese word segmentation bakeoff,” in *the Fourth SIGHAN workshop on Chinese Language Processing*, 2005.
- [152] T. Joachims, “A support vector method for multivariate performance measures,” in *ICML*, 2005.
- [153] A. Singhal, “Modern information retrieval: a brief overview,” in *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2001.

REFERENCES

- [154] Y. Yue, T. Finley, F. Radlinsko, and T. Joachims, “A support vector method for optimizing averaging precision,” in *SIGIR*, 2007.
- [155] L. Adamic and B. Huberman, “Power-law distribution of the world wide web,” *Science*, vol. 287, 2000.
- [156] C.-C. Chang and C.-J. Lin, “Libsvm : a library for support vector machines,” *ACM TIST*, vol. 2, pp. 1–27, 2011.
- [157] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas, “Multilabel classification of music into emotions,” in *Proceedings of International Conference on Music Information Retrieval (ISMIR 2008)*, 2008.
- [158] L.-J. Li and L. Fei-Fei, “what, where and who? classifying events by scene and object recognition,” in *ICCV*, 2007.
- [159] B. C. Russell, A. B. Torralba, K. P. Murphy, and W. T. Freeman, “Labelme: a database and web-based tool for image annotation,” *IJCV*, vol. 77(1-3), pp. 157–173, 2008.
- [160] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng, “NUS-WIDE: a real world web image database from national university of singapore,” in *CIVR*, 2009.
- [161] P. Duygulu, K. Barnard, J. de Freitas, and D. Forsyth, “Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary,” in *ECCV*, 2002.
- [162] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *CVPR*, 2006.
- [163] Y. Chen, J. Bi, and J. Z. Wang, “MILES: Multiple-instance learning via embedded instance selection,” *TPAMI*, vol. 28, pp. 1931–1947, 2006.
- [164] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, “Solving the multiple instance problem with axis-parallel rectangles,” *Artificial Intelligence*, vol. 89, pp. 31–71, 1997.

- [165] Q. Zhang, S. Goldman, W. Yu, and J. Fritts, “Content-based image retrieval using multiple-instance learning,” in *ICML*, 2002.
- [166] O. Maron and A. L. Ratan, “Multiple-instance learning for natural scene classification,” in *ICML*, 1998.
- [167] S. Andrews, I. Tsochantaridis, and T. Hofmann, “Support vector machines for multiple-instance learning,” in *NIPS*, 2003.
- [168] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, “Optimization with sparsity-inducing penalties,” *Foundations and Trends in Machine Learning*, vol. 4, pp. 1–106, 2012.
- [169] J. M. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization*. Springer, 2000.
- [170] A. Mutapcic and S. Boyd, “Cutting-set methods for robust convex optimization with pessimizing oracles,” *Optimization Methods & Software*, vol. 24, no. 3, p. 381406, 2009.
- [171] Z. Xu, R. Jin, I. King, and M. R. Lyu, “An extended level method for efficient multiple kernel learning,” in *NIPS*, 2008.
- [172] J. B. Hiriart-Urruty and C. Lemarechal, *Convex Analysis and Minimization Algorithms*. Springer-Verlag, 1993.
- [173] Y. Yue, T. Finley, F. Radlinski, and T. Joachims, “A support vector method for optimizing average precision,” in *SIGIR*, 2007.
- [174] A. Zien, U. Brefeld, and T. Scheffer, “Transductive support vector machines for structured variables,” in *ICML*, 2007.
- [175] O. Maron and A. L. Ratan, “Multiple-instance learning for natural scene classification,” in *ICML*, 1998.
- [176] Z.-H. Zhou and M.-L. Zhang, “Multi-instance multi-label learning with application to scene classification,” in *NIPS*, 2007.

REFERENCES

- [177] Q. Zhang and S. Goldman, “Em-dd: An improved multiple-instance learning technique,” in *NIPS*, 2001.
- [178] J. Wang and J.-D. Zucker, “Solving the multiple-instance problem: a lazy learning approach,” in *ICML*, 2000.
- [179] S. Andrews, I. Tsochantaridis, and T. Hofmann, “Support vector machines for multiple-instance learning,” in *NIPS*, 2007.
- [180] Z.-H. Zhou and M.-L. Zhang, “Ensembles of multi-instance learners,” in *ECML*, 2003.
- [181] Y.-Y. S. Z.-H. Zhou and Y.-F. Li, “Multi-instance learning by treating instances as non-i.i.d. samples,” in *ICML*, 2009.
- [182] L. Bo and C. Sminchisescu, “Structured output-associative regression,” in *CVPR*, 2009.
- [183] K. Jia, L. Wang, and N. Liu, “Efficient structured support vector regression,” in *ACCV*, 2010.
- [184] K. Daniilidis, P. Maragos, and N. Paragios, “Structured output ordinal regression for dynamic facial emotion intensity prediction,” in *ECCV*, 2010.
- [185] Y. Altun, D. McAllester, and M. Belkin, “Maximum margin semi-supervised learning for structured variables,” in *NIPS*, 2005.
- [186] C. Yu, “Transductive learning of structural svms via prior knowledge constraints,” in *AISTATS*, 2012.
- [187] U. Brefeld and T. Scheffer, “Semi-supervised learning for structured output variables,” in *ICML*, 2006.
- [188] A. Herschtal and B. Raskutti, “Optimising area under the roc curve using gradient descent,” in *ICML*, 2004.
- [189] J. Xu, T.-Y. Liu, M. Lu, H. Li, and W.-Y. Ma, “Directly optimizing evaluation measures in learning to rank,” in *ACM SIGIR*, 2008.

REFERENCES

- [190] M. Xu, Y.-F. Li, and Z.-H. Zhou, “Multi-label learning with PRO loss,” in *AAAI*, 2013.