Carnegie Mellon University
Dietrich College of Humanities and Social Sciences
School of Computer Science
Thesis

Submitted in Partial Fulfillment of the Requirements
For the Degree of Doctor of Philosophy

Title: Topics In Prediction — DRAFT

Presented by: Yotam Hechtlinger

Accepted by: Department of Statistics and Data Science & Machine Learning Department

Readers:

Larry Wasserman, Advisor

Alessandro Rinaldo, Advisor

Jing Lei

Ryan Tibshirani

Rebecca Nugent

Lucas Mentch, University of Pittsburgh

Approved by the Committee on Graduate Degrees:

Richard Scheines Date
Dietrich College, Dean

Martial Hebert Date
School of Computer Science, Dean

Carnegie Mellon University

Topics In Prediction — DRAFT

A Thesis Presented for

The Doctor of Philosophy

in

Statistics and Data Science & Machine Learning

by

Yotam Hechtlinger

Department of Statistics and Data Science & Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213

December 2019

Abstract

Modern science regularly requires to provide forecast of possible outcomes based on past experience. The field of statistics address this objective by developing methods that learn prediction functions from data. This thesis considers different aspects of such methods, mostly focused on the supervised learning setting in which the goal is to predict an output prediction of a variable Y from a set of input variables X's.

Level-set prediction is a non-parametric method for regression and classification. It utilizes $p(x \mid y)$ and $p(x, y)$ in contrast to dominant methods utilizing $p(y \mid x)$. The method provides conformal prediction sets that contains the right output with $1 - \alpha$ probability. The main advantage of this new approach is that it's cautious. It outputs the null set – meaning "I don't know" – when the input does not resemble previously seen examples.

Convolutional Neural Networks is used in many state-of-the-art prediction methods in the field of deep learning. Due to the operation of the convolution, the usage is limited to grid-structured input, such as $2D$ images and temporal sequences. *Graph Convolutional Neural Networks* generalizes the idea of convolution to graph structured input. It uses spatial methods to select the graph nearest neighbors in order to define an analogous convolution for graph.

There are situations in which the same data is used both for the selection and the inference. That can make the reported uncertainties deceptively optimistic: confidence intervals that ignore selection generally have less than their nominal coverage probability. *Confidence intervals for the selected parameters* construct valid confidence intervals post selection for the selection of the maximum k shift parameters out of m. These

intervals control the probability that one or more of the selected parameters do not cover—the "simultaneous over the selected" (SoS) error rate.

For a given model it is often desired to interpret which variables affect the prediction. We demonstrate the usefulness of the *input gradients* of the variables with respect to the prediction as a generic method to obtain model interpertability. The method conceptually generalizes the way linear regression studies the coefficients. We demonstrate its usefulness by interpreting complex neural networks on a sentiment analysis dataset.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Prediction is the forecasting of the future. Prediction has motivated humans since the early days of human existence. The ancient Greeks petitioned the Oracle of Delphi. Today we have science. Wikipedia defines science as a *systematic enterprise that builds and organizes knowledge in the form of testable explanations and predictions about the universe*. The scientific revolution aims to describe reality in a way that consistently provides sound predictions.

Statistics starts with data [Breiman, 2001]. It is a branch within science that bases reasoning on empirical data. The statistical discipline examines problems by collecting, analysing and interpreting data. The data implicitly serve as a means to attain information on unknown phenomena. Statistics, through careful usage of the data, is able to shed light on occurrences that we can't fully describe through logic and mathematics. It is also used to assess the validity of the models we use to describe the world.

This thesis focuses on statistical prediction. Statistical prediction uses empirical data to establish future predictions. In comparison to other prediction methods, the strength of statistical prediction follows from directly tying the occurrences of the past (the data) to draw influential statements about the future. There are many different questions within statistical prediction. In section 1.1 we give a broad overview of statistical prediction and in section 1.2 we discuss the thesis contribution.

1.1 Statistical Prediction Overview

From a broad mathematical perspective, prediction is a function. Even the Oracle of Delphi is a function. Given an input (*"Should I attack the Persians?"*) it provides an output (*"It will destroy a great empire"*). Statistical prediction uses empirical data and the mathematics of probability in order to get prediction functions. The prediction function approximates the information about the real world contained within the data.

Formally, statistical prediction learns a relation between an input (also called features or independent) random multivariate variable X defined over the space $\mathcal{X}$ to an output (also called responses or dependent) random variable Y defined over the space $\mathcal{Y}$. In the machine learning community this problem is called *supervised learning*. It is usually done using the empirical data which are samples from random pairs $(X_1, Y_1), \ldots, (X_n, Y_n)$. The pairs are often exchangeable repetitions from the joint distribution $\mathbb{P}$ of (X, Y). The unknown function is

$$f : \mathcal{X} \longmapsto \mathcal{Y}, \tag{1.1}$$

and the data are used to select an approximation function

$$\hat{f} : \mathcal{X} \longmapsto \mathcal{Y}. \tag{1.2}$$

Regression investigates the case when $\mathcal{Y}$ is continuous. It addresses problems in which the output is a value which belongs to an ordered set (*"What is this person's annual income?"*). The earliest example is linear regression and the least square method, which is also the origins of statistics. Linear regression assumes the relation between X and Y is linear. The least square method is the first algorithm for finding $\hat{f}$ in that case. It was invented and used by Legendre [1805] and Gauss (1809) for the prediction of planetary movement and was the dominant theme of nineteenth century mathematical statistics [Stigler, 1986]. By the 1830s it was a standard tool in astronomy and geodesy and was widespread in France, Italy, Prussia and England.

Since then countless breakthroughs have been made. Nevertheless, the regression prediction problem is still a central theme in twenty-first century statistics.

In contrast to regression, which predicts a continuous variable, classification investigates the case when $\mathcal{Y}$ is categorical. Classification addresses problems in which the output represents a selection of a particular category (*"Where does this person live?"*). Historically, classification emerged in the first half of the twentieth century [Fisher, 1936, 1938]. In the present day classification is widely used.

The applications of both regression and classification are broad. Countless research sciences and industries fundamentally base their decision making on statistical prediction models. These include, but are not limited to, the sciences of psychology, sociology, economics, biology, epidemiology, computer science (machine learning in particular) and the industries of technology, pharmaceuticals, genetics, finance and management. For example, computer vision uses classification to classify objects in the real world, which enables self-driving vehicles.

1.2 Topics In Prediction

This thesis covers several new methodologies related to statistical prediction. These methodologies offers richer prediction output which is more cautious towards outliers, applicable on graph structured datasets, address post-selection inference and study the problem of interpretability.

1.2.1 Set-Valued Prediction

Chapter 2 and 3 discuss set-valued prediction, which provides more informative output. Standard prediction methods yield a single output as in equation 1.2. Set-valued prediction estimates a prediction function for which the range is the power set of $\mathcal{Y}$, that is

$$\hat{f} : \mathcal{X} \longmapsto 2^{\mathcal{Y}}. \tag{1.3}$$

In regression, set-valued prediction usually outputs a set of intervals. In classification the output is a discrete and finite set. This type of prediction provides more complete information regarding the output. It is useful when a single output can't properly describe the output structure. But it comes with a price. A trivial predictor that outputs $\forall x \in \mathcal{X},\ \hat{f}(x) = \mathcal{Y}$ always achieves the right prediction. Therefore set-valued predictions requires more careful definitions of a "good" predictor.

The main tool we use to obtain set-valued predictors is conformal prediction. Conformal prediction is a method developed by Vladimir Vovk and his collaborators around the beginning of the century [Vovk et al., 2005, Shafer and Vovk, 2008]. In recent years the method has gained (and continues to gain) popularity.

The underlying concept behind conformal prediction is the empirical quantile. It assumes the past (the data collected) is a truthful representation of the future (the data to be predicted). Then — by carefully applying functions on the data — the output of the function of the future data is bounded within the empirical quantile of the function evaluated on the past data with a probabilistic degree of confidence. Conformal prediction enables the construction of many set-valued predictors. But the quality of the predictions considerably varies by the construction. Active research studies different ways to design set-valued predictors using conformal prediction. Rigorous and formal discussion of conformal prediction is covered in parts 2.3 and 3.2.

In chapter 2 (which is based on Hechtlinger et al. [2018]) we design a new predictor for classification that is established on the distribution of $p(x \mid y)$, in contrast to most other predictors that utilizes $p(y \mid x)$. The added value of this classifier is that it is cautious. It outputs the null set — meaning "I don't know" — when the input does not resemble previous examples in the data. Chapter 3 covers conformal predictors for the problem of regression. It conceptually discusses the advantages of $p(x, y)$ as a quantity that captures the complete information concealed in the data. We also provide simulation studies and comparisons for multiple common methods.

1.2.2 Graph Convolution

Chapter 4 (which is based on Hechtlinger et al. [2017]) generalizes the idea of Convolutional Neural Networks (CNN) to graph structured data. CNN are a leading tool used to address a large set of machine learning

problems (LeCun et al. [1998], LeCun et al. [2015]). They have successfully provided significant improvements in numerous fields, such as image processing, speech recognition, computer vision and pattern recognition, language processing and even the game of Go (Hinton et al. [2012], Le et al. [2011], Kim [2014], Silver et al. [2016] respectively).

CNN are part of the neural network framework. It is an intermediate layer which is built upon the convolution operator which requires grid-structured input. Recent years have seen a surge of research on the generalization of CNNs to other geometrical structures, in particular to graphs and manifolds. Many of the recent advances have been extensively reviewed by Bronstein et al. [2016].

The main contribution of this part is a novel generalization of CNNs to general graph-structured data, directed or undirected, called the Nearest Neighbors Convolutional Neural Network (NNCNN). It proposes spatial convolution that selects the top p closest neighbors of every node using a random walk. Then for every node, the convolution is computed as the inner product of the weights and the selected p closest neighbors, which are ordered according to their relative position from the given node. This allows the usage of the same set of weights (shared weights) for the convolution at every node and reflects the dependency between each node and its closest neighbors.

1.2.3 Confidence Intervals for Selected Parameters

Statistical inference is an central paradigm, widely used by many fields. Despite (or perhaps due to) the important role statistical inference has in many fields, in recent years its usage has been questioned. In many cases the reported significant findings are not replicable.

One main reason, among others, follows from the process of selection. The classical models of statistical inference first hypotheses how to model the world and then collect data to test the distributional assumptions. In practice, due to the emergence of computational technology, data is abundant. Many applications first collect the data and then select the most promising models. When the same data is used for the selection and the inference, the inversion of the process skew the distributions in a liberal direction. This yields over optimistic inferential claims. The problem is known as *post-selection inference*.

Chapter 5 (based on Benjamini et al. [2019]) studies this problem. In the chapter, we construct confidence intervals which are valid post-selection process. The intervals control the probability of "simultaneous over the selected" (SoS) error rate, which is the probability that one or more intervals for selected parameters do not cover. Intervals that control the SoS error rate can be constructed in ways that take advantage of knowledge of the selection rule. We show this process by constructing intervals for the selection of the top k out of m shift parameters that are estimated (by independent estimators) to be the largest. As shown in the chapter, these intervals offer significant improvement over simultaneous methods.

1.2.4 Input Gradients Interpretability

The *Interpretability* task is to interpret the prediction model $\hat{f}$. The goal is to understand the causes behind the decision the model output. There is not a clear mathematical definition of interpretability, but one way to approach it is to study which variables in X affect the prediction of $\hat{f}$ and in which way. Interpretability attracts interest in the machine learning community where models often are complex and effectively function as a "black box."

Chapter 6 (based on Hechtlinger [2016]) studies this problem . It suggests a simple method to interpret the behavior of any predictive model, both for regression and classification. Given a particular model, the information required to interpret it can be obtained by studying the partial derivatives of the model with respect to the input. This insight is exemplified by interpreting convolutional and multi-layer neural networks in the field of natural language processing.

Chapter 2

Level-set Classification

Chapter based on Hechtlinger et al. [2018].

We consider multiclass classification with a feature space $\mathcal{X}$ and labels $\mathcal{Y} = \{1, \ldots, k\}$. Given the training data $(X_1, Y_1), \ldots, (X_n, Y_n)$, the usual goal is to find a prediction function $\hat{F} : \mathcal{X} \longmapsto \mathcal{Y}$ with low classification error $P(Y \neq \hat{F}(X))$ where (X, Y) is a new observation of an input-output pair. This type of prediction produces a definite prediction even for cases that are hard to classify.

In this chapter we use conformal prediction [Vovk et al., 2005] where we estimate a set-valued function $C : \mathcal{X} \longmapsto 2^{\mathcal{Y}}$ with the guarantee that $P(Y \in C(X)) \geq 1 - \alpha$ for all distributions P. This is a distribution-free confidence guarantee. Here, $1 - \alpha$ is a user-specified confidence level. We note that the "classify with a reject option" [Herbei and Wegkamp, 2006] also allows set-valued predictions but does not give a confidence guarantee.

The function C can sometimes output the null set. That is, $C(x) = \emptyset$ for some values of x. This allows us to distinguish two types of uncertainty. When $C(x)$ is a large set, there are many possible labels consistent with x. But when x does not resemble the training data, we will get $C(x) = \emptyset$ alerting us that we have not seen examples like this so far.

There are many ways to construct conformal prediction sets. Our construction is based on finding an estimate $\hat{p}(x|y)$ of $p(x|y)$. We then find an appropriate scalar $\hat{t}_y$ and we set $C(x) = \{y : \ \hat{p}(x|y) > \hat{t}_y\}$. The

scalars are chosen so that $P(Y \in C(X)) \geq 1 - \alpha$. We shall see that this construction works well when there is a large number of classes as is often the case in deep learning classification problems. This guarantees that x's with low probability — that is regions where we have not seen training data — get classified as $\emptyset$.

An important property of this approach is that $\hat{p}(x|y)$ can be estimated independently for each class. Therefore, x is predicted to a given class in a standalone fashion which enables adding or removing classes without the need to retrain the whole classifier. In addition, we empirically demonstrate that the method we propose is applicable to large-scale high-dimensional data by applying it to the ImageNet ILSVRC dataset and the CelebA and IMDB-Wiki facial datasets using features obtained from state of the art convolutional neural networks.

Chapter Outline. In 2.1 we discuss the difference between $p(y|x)$, $p(x|y)$ and $p(x, y)$. In 2.2 we provide an example to enlighten our motivation. In 2.3 we present the general framework of conformal prediction and survey relevant works in the field. In 2.4 we formally present our method. In 2.6 we demonstrate the performance of the proposed classifier on the ImageNet challenge dataset using state of the convolutional neural networks. In 2.7 we consider the problem of gender classification from facial pictures and show that even when current classifiers fail to generalize from CelebA dataset to IMDB-Wiki dataset, the proposed classifier still provides sensible results. 2.8 contains our discussion and concluding remarks. The Appendix in the supplementary material contains some technical details.

Related Work. There is an enormous literature on set-valued prediction. Here we only mention some of the most relevant references. The idea of conformal prediction originates from Vovk et al. [2005]. There is a large followup literature due to Vovk and his colleagues which we highly recommend for the interested readers. Statistical theory for conformal methods was developed in [Lei, 2014, Lei et al., 2013, Lei and Wasserman, 2014], and the multiclass case was studied in [Sadinle et al., 2019] where the goal was to develop small prediction sets based on estimating $p(y|x)$. The authors of that paper, similarly to Vovk et al. [2003], tried to avoid outputting null sets. In this work, we use this as a feature. This approach was also adopted in Guan and Tibshirani [2019], which uses conformal prediction to construct BCOPS. BCOPS tries to optimize

the out-of-sample performance, aiming to include the correct class as often as possible, but also detecting outliers.

Finally, we mention a related but different technique called classification with the "reject option" [Herbei and Wegkamp, 2006]. This approach permits one to sometimes refrain from providing a classification but it does not aim to give confidence guarantees.

Recently, Lee et al. [2018] suggested a framework based on $p(x|y)$ to predict out of distribution and adversarial attacks.

2.1 $p\left(y \mid x\right)$ Versus $p\left(x \mid y\right)$ Versus $p\left(x, y\right)$

Most classifiers — including most conformal classifiers — are built by estimating $p(y|x)$. Typically one sets the predicted label of a new x to be $\hat{f}\left(x\right) = \arg\max_{y \in \mathcal{Y}} \left\{\hat{p}(y|x)\right\}$. Since $p(y|x) = p(x|y)p(y)/p(x)$ the prediction involves the balance between $p(y)$ and $p(x|y)$. The motivation follows from the assumption that is a single class has to be selected, it's preferable to take into account the distribution of $p(y)$. In the special case $p(y) = 1/k$ for all y, we have $\arg\max_{y \in \mathcal{Y}}\{p(y|x)\} = \arg\max_{y \in \mathcal{Y}}\{p(x|y)\}$.

For set-valued classification, $p(y|x)$ can be negatively affected by $p(y)$ and $p(x)$. The conformal methods utilizing $p(y|x)$ to predict a set of classes classify x to $C\left(x\right) = \left\{y \mid p\left(y \mid x\right) \geq t_\alpha\right\}$ for some threshold t_α. These methods face the inherent weakness within $p(y|x)$.

First, taking $p(y)$ into account ties the prediction of an observation x with the likelihood of observing that class. Since there is no restriction on the number of classes, ultimately an observation should be predicted to a class regardless of the class popularity.

Second, normalizing by $p(x)$ makes the classifier oblivious to the probability of actually observing x. When $p(x)$ is extremely low (an outlier), $p(y|x)$ still selects the most likely label out of all tail events. In practice this may result with most of the space classified with high probability to a handful of classes almost arbitrarily despite the fact that the classifier has been presented with virtually no information in those areas of the space. This approach might be necessary if a single class has to be selected $\forall x \in \mathcal{X}$. However, if this is not the case, then a reasonable prediction for an x with small $p(x)$ is the null set.

Third, $p(y|x)$ obtains low values at the decision boundary. Due to the nature of $p(y|x)$ the points which are most likely to be predicted as the null set are when $p(y = j|x) = \frac{1}{k}$, for all classes $j \in \mathcal{Y}$. But this is exactly the points in space for which any set valued prediction should predict all class as possible.

An alternative approach — which is the main idea behind density-level set classification — is to use $p(x|y)$. As will be formalized later, the goal is to classify x to $C(x) = \{y \mid p(x \mid y) \geq t_\alpha\}$ for some threshold t_α. The quantity $p(x|y)$ overcomes all of $p(y|x)$ disadvantages. First, regardless of the distribution of $p(y)$, an observation is predicted to a class if it has high probability to be from the class. Second, the outliers are predicted as null sets, since low values of $p(x)$ also has low values of $p(x|y)$. Third, the decision boundaries are a function of $p(y|x)$. Thus, $p(x|y)$ is invariant to the structure of the decision boundary and predicts multiple classes where multiple classes obtain high values of $p(x|y)$.

Another approach is to use $p(x, y)$. The relation between $p(x, y)$ and $p(x \mid y)$ is more delicate since $p(x, y) = p(x \mid y)\, p(y)$ and in this case $\mathcal{Y}$ is discrete. Therefore, if we find for every class a specific $t_{\alpha,y}$ then

$$\{y \mid p(x, y) \geq t_{\alpha,y}\} = \left\{y \mid p(x \mid y) \geq \tilde{t}_{\alpha,y}\right\}, \tag{2.1}$$

and the results coincide. If we select a global threshold t_α this approach is biased towards classes in which $p(y)$ is large, resulting with shortened prediction of classes in which $p(y)$ is small. However, since $p(x, y) \geq t_\alpha$ minimizes the joint space Lebesgue measure, this approach when facing outliers is likely to predicts the null set the most.

2.2 Motivating Example - Iris Dataset

The Iris flower data set is a benchmark dataset often used to demonstrate classification methods. It contains four features that were measured from three different Iris species. In this example, for visualization purposes, we only use two features: the sepal and petal lengths in cm.

Figure 2.1 shows the decision boundaries for this problem comparing the results of (a) K-nearest neighbors (KNN), (b) support vector machines with the RBF kernel (SVM) and (c) our conformal prediction method using an estimate $\hat{p}(x|y)$.

Both the KNN and the SVM methods provide sensible boundaries between the class where there are observations. In areas with low density $p(x)$ the decision boundaries are significantly different. The SVM classifies almost all of the space to a single class. The KNN creates an infinite strip bounded between two (almost affine) half spaces. In a hubristic manner, both methods provide very different predictions with probability near one without sound justification.

The third plot shows the conformal set $C(x) = \{y : \hat{p}(x|y) > \hat{t}_y\}$ where the $\hat{t}_y$ is chosen as described in Section 2.4. The result is a cautious prediction. If a new x falls into a region with little training data then we output $\emptyset$. In such cases our proposed method modestly avoids providing any claim.

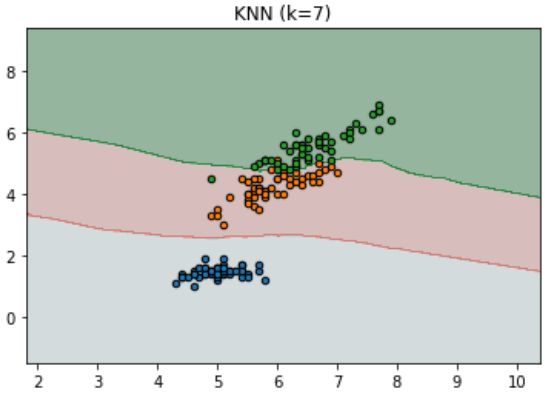
2.3 Conformal Prediction

Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ be n independent and identically distributed (iid) pairs of observations from a distribution P. In set-valued supervised prediction, the goal is to find a set-valued function $C(x)$ such that

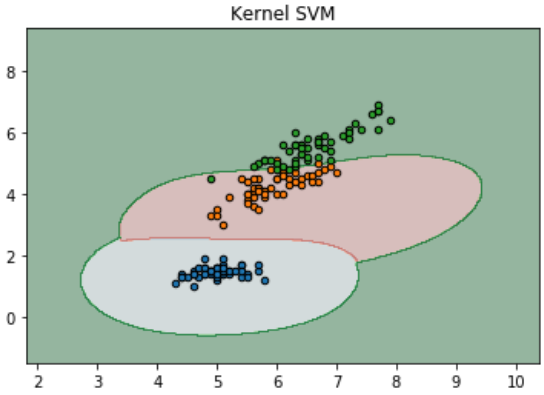
$$P(Y \in C(X)) \geq 1 - \alpha, \tag{2.2}$$

where (X, Y) denotes a new pair of observations.

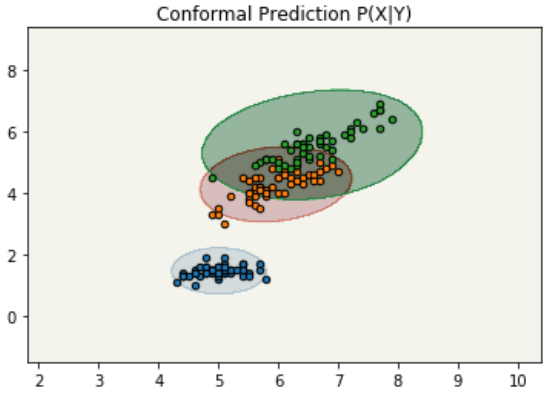
Conformal prediction — a method created by Vovk and collaborators [Vovk et al., 2005] — provides a general approach to construct prediction sets based on the observed data without any distributional assumptions. The main idea is to construct a conformal score, which is a real-valued, permutation-invariant function $\psi(z, \mathcal{D})$ where $z = (x, y)$ and $\mathcal{D}$ denotes the training data. Next we form an augmented dataset $\mathcal{D}' = \{(X_1, Y_1), \ldots, (X_n, Y_n), (X_{n+1}, Y_{n+1})\}$ where (X_{n+1}, Y_{n+1}) is set equal to arbitrary values (x, y). We then define $R_i = \psi((X_i, Y_i), \mathcal{D}')$ for $i = 1, \ldots, n+1$. We test the hypothesis $H_0 : Y = y$ that the new label Y is equal to y using the p-value $\pi(x, y) = 1/(n+1) \sum_{i=1}^{n+1} I(R_i \geq R_{n+1})$. Then we set $C(x) = \{y : \pi(x, y) \geq \alpha\}$.



(a) K-NN classification results



(b) Kernal SVM classification results



(c) Prediction interval based on $p(x \mid y)$

Figure 2.1: Classification boundaries for different methods for the Iris dataset. For the conformal prediction method (c) (with $\alpha = 0.05$) the overlapping areas are classified as multiple classes and white areas are classified as the null set. For the standard methods (a-b), the decision boundaries can change significantly with small changes in some of the data points and the prediction cannot be justified in most of the space. Online version in color.

[Vovk et al., 2005] proves that $P(Y \in C(X)) \geq 1 - \alpha$ for all distributions P. There is a great flexibility in the choice of conformity score and 2.3.1 discusses important examples.

As described above, it is computationally expensive to construct $C(x)$ since we must re-compute the entire set of conformal scores for each choice of (x, y). This is especially a problem in deep learning applications where training is usually expensive. One possibility for overcoming the computational burden is based on data splitting where $\hat{p}(x|y)$ is estimated from part of the data and the conformal scores are estimated from the remaining data; see [Vovk, 2015, Lei and Wasserman, 2014]. Another approach is to construct the scores from the original data without augmentation. In this case, we no longer have the finite sample guarantee $P(Y \in C(X)) \geq 1 - \alpha$ for all distributions P, but we do get the asymptotic guarantee $P(Y \in C(X)) \geq 1 - \alpha - o_P(1)$ as long as some conditions are satisfied*. See [Sadinle et al., 2019] for further discussion on this point.

2.3.1 Examples

Here are several known examples for conformal methods used on different problems.

Supervised Regression. Suppose we are interested in the supervised regression problem. Let $\hat{f} : \mathcal{X} \to \mathcal{Y}$ be any regression function learned from training data. Let ϵ_i denote the residual error of $\hat{f}$ on the observation i, that is, $\epsilon_i = |\hat{f}(X_i) - Y_i|$. Now we form the ordered residuals $\epsilon_{(1)} \leq \cdots \leq \epsilon_{(n)}$, and then define

$$C(x) = \left\{ y : \ |\hat{f}(x) - y| \leq \epsilon_{(\lceil (1-\alpha) \cdot n \rceil)} \right\}.$$

If $\hat{f}$ is a consistent estimator of $\mathbb{E}[Y|X = x]$ then $P(Y \in C(X)) = 1 - \alpha + o_P(1)$. See Lei and Wasserman [2014].

Unsupervised Prediction. Suppose we observe independent and identically distributed $Y_i, \ldots, Y_n \in \mathbb{R}^d$ from distribution P. The goal is to construct a prediction set C for new Y. Lei et al. [2013] use the level set $C = \{y : \ \hat{p}(y) > t\}$ where $\hat{p}$ is a kernel density estimator. They show that if t is chosen carefully then $P(Y \in C) \geq 1 - \alpha$ for all P.

*A sequence of random variables $X_1, X_2, \ldots,$ is $o_p(1)$ if $\forall \epsilon > 0$, $\lim_{n \to \infty} P(|X_n| \geq \epsilon) = 0$.

Multiclass Classification. There are two notable solutions also using conformal prediction for the multiclass classification problem which are directly relevant to this work.

Least Ambiguous Set-Valued Classifiers with Bounded Error Levels. Sadinle et al. [2019] extended the results of Lei [2014] and defined $R_i = \hat{p}\left(Y_i \mid X_i\right)$, where $\hat{p}$ is any consistent estimator of $p(y|x)$. They defined the *minimal ambiguity* as $\mathbb{A}\left(C\right) = \mathbb{E}\left(\left|C\left(X\right)\right|\right)$ which is the expected size of the prediction set. They proved that out of all the classifiers achieving the desired $1-\alpha$ coverage, this solution minimizes the ambiguity. In addition, the paper considers class specific coverage controlling for every class $P\left(Y \in C\left(X\right) \mid Y = y\right) \geq 1-\alpha_y$.

Universal Predictor. Vovk et al. [2003] introduce the concept of universal predictor and provide an explicit way to construct one. A universal predictor is the classifier that produces, asymptotically, no more multiple prediction than any other classifier achieving $1-\alpha$ level coverage. In addition, within the family of all $1-\alpha$ classifiers that produce the minimal number of multiple predictions it also asymptotically obtains at least as many null predictions.

2.4 The Method

2.4.1 The Classifier

Let $\hat{p}(x|y)$ be an estimate of the density $p(x|y)$ for class $Y = y$. Define $\hat{t}_y$ to be the empirical $1-\alpha$ quantile of the values $\{\hat{p}(X_i|y)\}$. That is,

$$\hat{t}_y = \sup\left\{ y : \ \frac{1}{n_y} \sum_i I(\hat{p}(X_i|y) \geq t) \geq 1-\alpha \right\} \tag{2.3}$$

where $n_y = \sum_i I(Y_i = y)$. Assuming that $n_y \to \infty$ and minimal conditions on $p(x|y)$ and $\hat{p}(x|y)$, it can be shown that $\hat{t}_y \xrightarrow{P} t_y$ where t_y is the largest t such that $\int_{y>t} p(x|y)dx \geq 1-\alpha$. See Cadre et al. [2009] and Lei et al. [2013]. We set $C(x) = \{y : \ \hat{p}(x|y) \geq \hat{t}_y\}$. We then have the following proposition which is proved in the appendix.

Assume the conditions in Cadre et al. [2009] stated also in the appendix. Let (X, Y) be a new observation. Then $|P(Y \in C(X)) - (1-\alpha)| \xrightarrow{P} 0$ as $\min_y n_y \to \infty$.

An exact, finite sample method can be obtained using data splitting. We split the training data into two parts. Construct $\hat{p}(x|y)$ from the first part of the data. Now evaluate $\{\hat{p}(X_i|y)\}$ on the second part of the data and define $\hat{t}_y$ using these values. We then set $C(x) = \{y : \hat{p}(x|y) \geq \hat{t}_y\}$. We then have:

Let (X, Y) be a new observation. Then, for every distribution and every sample size, $P(Y \in C(X)) \geq 1-\alpha$.

This follows from the theory in Lei and Wasserman [2014]. The advantage of the splitting approach is that there are no conditions on the distribution, and the confidence guarantee is finite sample. There is no large sample approximation. The disadvantage is that the data splitting can lead to larger prediction sets. Algorithm 3 describes the training, and Algorithm 4 describes the prediction.

Algorithm 1 Training Algorithm

Input: Training data $Z = (X, Y)$, Class list $\mathcal{Y}$, Confidence level α, Ratio p.
$\hat{p}_{list} = list; \hat{t}_{list} = list$ ▷ Initialize lists **for** y *in* $\mathcal{Y}$ **do**
end
Loop over all the classes independently
$X^y_{tr}, X^y_{val} \leftarrow SubsetData\left(Z, \mathcal{Y}, p\right)$ ▷ Split $X \mid y$ with ratio p
$\hat{p}_y \leftarrow LearnDensityEstimator\left(X^y_{tr}\right)$
$\hat{t}_y \leftarrow Quantile\left(\hat{p}_y\left(X^y_{val}\right), \alpha\right)$ ▷ The validation set α quantile
$\hat{p}_{list}.append\left(\hat{p}_y\right); \hat{t}_{list}.append\left(\hat{t}_y\right)$

return $\hat{p}_{list}$; $\hat{t}_{list}$

Algorithm 2 Prediction Algorithm

Input: Input to be predicted x, Trained $\hat{p}_{list}$; $\hat{t}_{list}$, Class list $\mathcal{Y}$.
$C = list$ ▷ Initialize $C\left(x\right)$ **for** y *in* $\mathcal{Y}$ **do**
end
Loop over all the classes independently **if** $\hat{p}_y\left(x\right) \geq \hat{t}_y$ **then**
end
$C.append\left(y\right)$

return C

2.4.2 Density Estimation

The density $p(x|y)$ has to be estimated from data. One possible way is to use the standard kernel density estimation method, which was shown to be optimal in the conformal setting under weak conditions in Lei, Robins, and Wasserman [2013]. This is useful for theoretical purposes due to the large literature on the topic. Empirically, it is faster to use the distance from the k nearest neighbors.

Density estimation in high dimensions is a difficult problem. Nonetheless, as we will show in the numerical experiments (Section 2.6), the proposed method works well in these tasks as well. An intuitive reason for this could be that the accuracy of the conformal prediction does not actually require $\hat{p}(x|y)$ to be close to $p(x|y)$ in L_2. Rather, all we need is that the ordering imposed by $\hat{p}(x|y)$ approximates the ordering defined by $p(x|y)$. Specifically, we only need that $\{(x, x') : \ p(x|y) > p(x'|y) + \Delta\}$ is approximated by $\{(x, x') : \ \hat{p}(x|y) > \hat{p}(x'|y) + \Delta\}$ for $\Delta > 0$. We call this "ordering consistency." This is much weaker than the usual requirement that $\int (\hat{p}(x|y) - p(x|y))^2 dx$ be small. This new definition and implications on the approximation of $p\left(x \mid y\right)$ will be further expanded in future work.

2.4.3 Class Adaptivity

As algorithms 3 and 4 demonstrate, the training and prediction of each class is independent from all other classes. This makes the method adaptive to addition and removal of classes ad-hoc. Intuitively speaking, if there is $1 - \alpha$ probability for the observation to be generated from the class it will be classified to the class regardless of any other information.

Another desirable property of the method is that it is possible to obtain different coverage levels per class if the task requires that. This is achieved by setting $\hat{t}_y$ to be the $1 - \alpha_y$ quantile of the values $\{\hat{p}(X_i|y)\}$.

2.4.4 Class Interaction

Defining $p(x|y)$ independently for each class has the desired property of class adaptivity, but also it discards relevant information regarding the relations of each of the classes. Figure 2.1 (c) demonstrate how the different classifiers decision boundaries are independent.

Figure 2.2: An example of more complex estimation function of $p(x|y)$ providing 0.95 coverage on the Iris dataset. $p(x|y)$ is higher for points which has high density within the class and low density in the other classes. The decision boundary closely resemble standard methods, while still providing cautious prediction and robustness to out of sample predictions.

More complex decision boundaries can be created using correlated estimators of $p(x|y)$. One example of such an estimator is $p'(x|y) \propto p(x|y) - \lambda \sum_{y' \neq y} p(x|y')$, for some $\lambda \in \mathbb{R}$. This estimator penalizes high density regions for the other classes. Figure 2.2 visualize the results of such estimator on the Iris dataset.

2.5 Outlier Analysis

In section 2.1 we intuitively discuss the advantages of $p(x \mid y)$ over $p(y \mid x)$ and $p(x, y)$ in the conformal setting. Since this is the main insight behind this line of research, here we analyze the behavior of outliers

under Guassian assumptions. Assume the supervised learning setting with joint distribution defined over $\mathcal{X} \times \mathcal{Y}$. For a given ϵ, we define the ϵ level outliers set as

$$\mathcal{I}_\epsilon = \{x \in \mathcal{X} \mid p(x) \leq \epsilon\}. \tag{2.4}$$

Denote $C_\mathcal{L}$ as the Lebesgue prediction conformal classifier, that is

$$C_\mathcal{L}(x) = \{y \in \mathcal{Y} \mid P(x \mid y) \geq t_{y,\alpha}\}, \tag{2.5}$$

where $t_{y,\alpha}$ is the $1-\alpha$ quantile of $p(x \mid y)$. Denote $C_\mathcal{B}$ as the conformal classifier described at Sadinle et al. [2019]. Specifically:

$$C_\mathcal{B}(x) = \{y \in \mathcal{Y} \mid p(y \mid x) \geq q_{y,\alpha}\}, \tag{2.6}$$

where $q_{y,\alpha}$ is the $1-\alpha$ quantile of $p(y \mid x)$ within the class y. Intuitively we've claimed Lebesgue prediction predict the null set for outliers. The following lemma makes it more concrete.

Lemma 2.0.1. *For any $\alpha \in (0,1)$, there exist $\widetilde{c}_\alpha > 0$ for which $\forall x \in \mathcal{I}_{\widetilde{c}_\alpha}$, $C_\mathcal{L}(x) = \emptyset$.*

Proof. Denote $\widetilde{T}_\alpha = \min_y \{t_{y,\alpha}\}$ and $\delta_y = \min_y \{p(y)\}$. Let $x \in \mathcal{I}_{\widetilde{T}_\alpha \cdot \delta_y}$. Then

$$\begin{aligned} \widetilde{T}_\alpha \cdot \delta_y &\geq p(x) \\ &= \sum_y p(x \mid y)\, p(y) \\ &\geq \sum_y p(x \mid y)\, \delta_y. \end{aligned} \tag{2.7}$$

Thus it follows that for all $x \in \mathcal{I}_{\widetilde{T}_\alpha \cdot \delta_y}$ it happens that for all $y \in \mathcal{Y}$

$$p(x \mid y) \leq \widetilde{T}_\alpha \leq t_{y,\alpha}. \tag{2.8}$$

In particular for all $x \in \mathcal{I}_{\widetilde{T}_\alpha \cdot \delta_y}$, $C_{\mathcal{L}}(x) = \emptyset$. $\square$

The analysis of $C_{\mathcal{B}}$ is less general and depends on the joint distribution. At this stage the intuition can be supported by results on multivariate Gaussian, as shown in the next Lemma.

Consider a mixture of p multivariate Gaussian variables such that $X_i \mid Y_i = k \sim \mathcal{N}(\mu_k, \Sigma_k)$, where $X_i \in \mathbb{R}^n$ and $P(Y_i = k) = \pi_k$. We assume that for all k, Σ_k is positive definite and for all $j \neq k$, $\Sigma_k \neq \Sigma_j$. We denote $\mathcal{S}^{n-1} = \{x \in \mathbb{R}^n : \|x\| = 1\}$, where $\|\cdot\|$ is the $\mathcal{L}_2$ norm.

Lemma 2.0.2. *For almost all $u \in \mathcal{S}^{n-1}$ and $v \in \mathbb{R}^n$,*

$$\lim_{a \to \infty} \{|C_{\mathcal{B}}(a \cdot u + v)|\} = 1. \tag{2.9}$$

Proof. Equation 2.9 is satisfied if for all pairs of different classes, j and k

$$\lim_{a \to \infty} \left\{ \frac{p(Y = k \mid a \cdot u + v)}{p(Y = j \mid a \cdot u + v)} \right\} \in \{0, \infty\}, \tag{2.10}$$

for almost all $u \in \mathcal{S}^{n-1}$ and $v \in \mathbb{R}^n$. Rewriting the ratio between the classes provide the following form

$$\frac{p(Y = k \mid a \cdot u + v)}{p(Y = j \mid a \cdot u + v)} = \frac{\pi_k}{\pi_j} \cdot \frac{|\Sigma_j|}{|\Sigma_k|} \exp\left\{h_u\left(a^2\right) + g_{u,v}(a) + c_v\right\}, \tag{2.11}$$

where we denote $\widetilde{\mu}_k \equiv v - \mu_k$, and

$$h_u\left(a^2\right) = -\frac{a^2 \left\langle \left(\Sigma_k^{-1} - \Sigma_j^{-1}\right) u, u \right\rangle}{2}, \tag{2.12}$$

$$g_{u,v}(a) = a \left\langle \Sigma_k^{-1} \widetilde{\mu}_k - \Sigma_j^{-1} \widetilde{\mu}_j, u \right\rangle, \tag{2.13}$$

$$c_v = -\frac{\left\langle \widetilde{\mu}_k, \Sigma_k^{-1} \widetilde{\mu}_k \right\rangle - \left\langle \widetilde{\mu}_j, \Sigma_j^{-1} \widetilde{\mu}_j \right\rangle}{2}. \tag{2.14}$$

The limit of Equation 2.9 is not $\{0, \infty\}$ only if $h_u\left(a^2\right) = 0$ and $g_{u,v}(a) = 0$. Since $\Sigma_k \neq \Sigma_j$, there are at most $n-1$ solutions for which $h_u\left(a^2\right) = 0$. These solutions need to also satisfy $g_{u,v}(a) = 0$. If $g_{u,v}(a) = 0$ due to orthogonality than for a given u it also has finite number of solutions in v. If $g_{u,v}(a) = 0$ because $\Sigma_k^{-1}\widetilde{\mu}_k = \Sigma_j^{-1}\widetilde{\mu}_j$, then the latter is a system with equal or greater number of equations than unknowns, which also has finite number of solutions. □

2.6 ImageNet Challenge Example

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Deng et al., 2009] is a large visual dataset of more than 1.2 million images labeled across $1,000$ different classes. It is considered a large scale complex visual dataset that reflects object recognition state-of-the-art through a yearly competition.

In this example we apply our conformal image classification method to the ImageNet dataset. We remove the last layer from the pretrained Xception convolutional neural network [Chollet, 2016] and use it as a feature extractor. Each image is represented as a $2,048$ dimensional feature in $\mathbb{R}^{2048}$. We learn for each of the $1,000$ classes a unique kernel density estimator trained only on images within the training set of the given class. When we evaluate results of standard methods we use the Inception-v4 model [Szegedy et al., 2017] to avoid correlation between the feature extractor and the prediction outcome as much as possible.

The Xception model obtains near state-of-the-art results of 0.79 (top-1) and 0.945 (top-5) accuracy on ImageNet validation set. As a sanity check to the performance of our method, selecting for each image the highest (and top 5) prediction of $\hat{p}(x \mid y)$ achieves 0.721 (top-1) and 0.863 (top-5) on ImageNet validation set. We were pleasantly surprised by this result. Each of the $\hat{p}(x \mid y)$'s were learned independently possibly discarding relevant information on the relation between the classes. The kernel density estimation is done in $\mathbb{R}^{2,048}$ and the default bandwidth levels were used to avoid overfitting the training set. Yet the naive performance is roughly on par with GoogLeNet [Szegedy et al., 2015] the winners of 2014 challenge (top-1: 0.687, top-5: 0.889).

For conformal methods the confidence level is predefined. The method calibrates the number of classes in the prediction sets to satisfy the desired accuracy level. The the main component affecting the results is

Our Method (α=.6, α=.8):	Inception-v4 Model:
• Bakery • Barber Shop • Shoe Shop • Toy Shop	• Barber Shop (0.994) • Barber Chair (0.002) • Shoe Shop (0.001)

(a)

Our Method (α=.6, α=.8):	Inception-v4 Model:
• Brain Coral • Coral Reef • Hair Slide • Tray • Water Bottle	• Brain Coral (0.927) • Coral Reef (0.003) • Brass (0.0006)

(b)

Figure 2.3: (a-b) are illustrative examples. When $\alpha = 0.6$ both black and red classes are predicted. When $\alpha = 0.8$ the red classes remain.

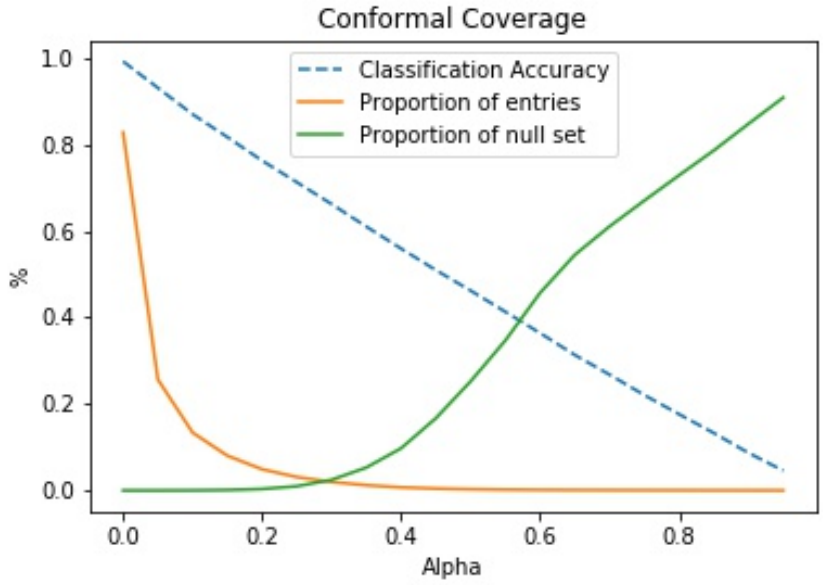


Figure 2.4: Performance plot for the conformal method. Accuracy is empirically linear as a function of α but affect the number of classes predicted per sample.

the hyperparameter α. For small values of α the accuracy will be high but so does the number of classes predicted for every observation. For large values of α more observations are predicted as the null set and less observations predicted per class. Figure 2.4 presents the trade-off between the α level and the number of classes and the proportion of null set predictions for this example. For example 0.5, accuracy would require on average 2.7 predictions per observation and 0.252 null set predictions. The actual selection of the proper α value is highly dependent on the task. As discussed earlier, a separate α_y for each class can also be used to obtain different accuracy per class.

Figures 2.3 (a) and (b) show illustrative results from the ImageNet validation set. (a) presents a picture of a "*Barber Shop*". When $\alpha = 0.6$ the method correctly suggests the right class in addition to several other relevant outcomes such as "*Bakery*". When $\alpha = 0.8$ only the "*Barber Shop*" remains. (b) show a "*Brain Coral*". For $\alpha = 0.6$ the method still suggests classes which are clearly wrong. As α increases the number of classes decrease and for $\alpha = 0.8$ only "*Brain Coral*" and "*Coral Reef*" remains, both which are relevant. At $\alpha = 0.9$ "*Coral Reef*" remains, which represents a misclassification following from the fact that the class threshold is lower than that of "*Brain Coral*". Eventually at $\alpha = 0.95$ the null set is predicted for this picture.

Figure 2.6 shows a collage of 20 images using $\alpha = 0.7$. To avoid selection bias we've selected the first 20 images in the ImageNet validation set.

2.6.1 Adversarial Robustness

Adversarial attacks attempt to fool machine learning models through malicious input. The suggested method is designed to be cautious and provide multiple predictions under uncertainty which results with a robust performance under different attacks. In this section we use the foolbox library [Rauber et al., 2017] to generate different attacks on ImageNet validation and test the performance of the method on the ResNet50 model [He et al., 2016]. We attack the first 200 images that are accurately classified by the model.

Table 2.1 shows the prediction results of two type of attacks, untargeted (using Deepfool [Rauber et al., 2017] and the FGSM attack [Kurakin et al., 2016]) and targeted (using L-BFGS-B [Tabacof and Valle, 2016] and Projected Gradient Descent (PGD) [Kurakin et al., 2016]). The untargeted attacks perturb the image the

Accuracy	$1-\alpha = 0.5$		$1-\alpha = 0.75$	
	True	Adversarial.	True	Adversarial
DeepFool (Untargeted)	0.510	0.385	0.770	0.680
FGSM (Untargeted)	0.495	0.375	0.750	0.720
L-BFGS-B (Targeted)	0.000	0.015	0.080	0.135
PGD (Targeted)	0.105	0.095	0.430	0.255

Table 2.1: Accuracy on 200 adversarial images. True denotes the accuracy over the original class and Adversarial denotes the accuracy on the adversarial class. Untargeted attacks obtain the expected accuracy on the true label. Targeted attacks have a greater effect on the image and are less likely to be predicted in any class.

least in order to find any misclassification. This yields predictions of both the true class and the adversarial class. While the attack reduces the performance of the model, the model is more robust than standard methods. Targeted attacks attempt to predict a specific class given apriori (randomly selected). This requires the attack to create larger modifications to the original image, and as a result the model mostly predict the null set both for the true label and the adversarial label.

2.6.2 Outliers

Figure 2.5 (a) shows the outcome when the input is random noise. We set the threshold $\alpha = 0.01$. This gives a less conservative classifier that should have the largest amount of false positives. Even with such a low threshold all 100 random noise images over $1,000$ categories are correctly flagged as the null set. Evaluating the same sample on the Inception-v4 model [Szegedy et al., 2017] results with a top prediction average of 0.0836 (with 0.028 standard error) to "Kite" and 0.0314 (0.009) to "Envelope". The top-5 classes together has mean probability of 0.196, much higher than the uniform distribution expected for prediction of random noise.

Figure 2.5 (b) show results on Jackson Pollock paintings - an abstract yet more structured dataset. Testing 11 different paintings with $\alpha = 0.5$ all result with the null set. When testing the Inception-v4 model output, 7/11 paintings are classified with probability greater than 0.5 to either "Coil", "Ant", "Poncho", "Spider Web" and "Rapeseed" depending on the image.

Figure 2.5 (c) is the famous picture of Muhammad Ali knocking out Sonny Liston during the first round of the 1965 rematch. "Boxing" is not included within in the ImageNet challenge. Our method correctly

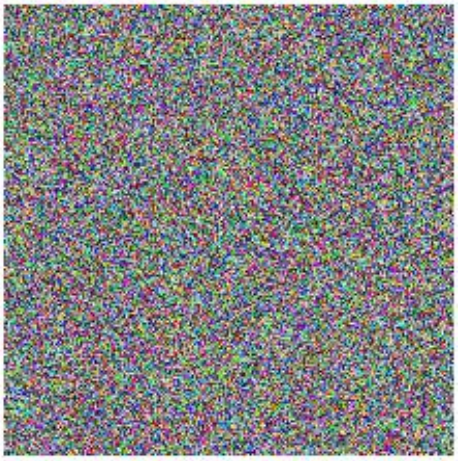
chooses the null set with α as low as 0.55. Standard method are forced to associate this image with one of the classes and choose "Volleyball" with 0.38 probability and the top-5 are all sport related predictions with 0.781 probability. This is good result given the constraint of selecting a single class, but demonstrate the impossibility of trying to create classes for all topics.

2.7 Gender Recognition Example

In the next example we study the problem of gender classification from facial pictures. CelebFaces Attributes Dataset (CelebA) [Liu et al., 2015] is a large-scale face attributes dataset with more than $200K$ celebrity images attributed, each with 40 attribute annotations including the gender (Male/Female). IMDB-Wiki dataset is a similar large scale ($500K+$ images) dataset [Rothe et al., 2016] with images taken from IMDB and Wikipedia.

We train a standard convolutional neural network (5 convolution and 2 dense layers with the corresponding pooling and activation layers) to perform gender classification on CelebA. It converges well obtaining 0.963 accuracy on a held out test set, but fails to generalize to the IMDB-Wiki dataset achieving 0.577 accuracy, slightly better than a random guess. The discrepancy between the two datasets follows from the fact that facial images are reliant on preprocessing to standardize the input. We have used the default preprocessing provided by the datasets, to reflect a scenarios in which the distribution of the samples changes between the training and the testing. Figure 2.7 (a) and (b) show mean pixel values for females pictures within CelebA vs pictures in the IMDB-Wiki dataset. As seen, the IMDB-Wiki is richer and offers larger variety of human postures.

Although the standard classification method fails in this scenario, the conformal method suggested in this work still offers valid and sensible results both on CelebA and IMDB-Wiki when using the features extracted from the network trained on CelebA. Figure 2.8 shows the performance of the method with respect to both datasets. CelebA results are good since they are based on features that perform well for this dataset. The level of accuracy is roughly $1-\alpha$ as expected by the design, while the proportion of null predictions is roughly α. Therefore for all α there are almost no false positives and all of the errors are the null set.

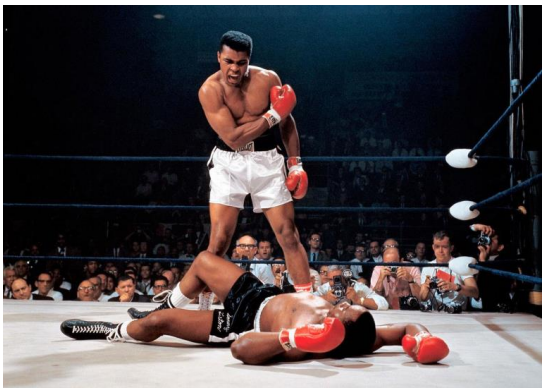


Our Method (α=.5):	Inception-v4 Model:
• Null Set	• Kite (0.137) • Bee Eater (0.033) • Missle (0.031)

(a)

Our Method (α=.5):	Inception-v4 Model:
• Null Set	• Coil (0.910) • Hay (0.008) • Maze (0.005)

(b)



Our Method (α=.55):	Inception-v4 Model:
• Null Set	• Volleyball (0.388) • Tennis Ball (0.160) • Racket (0.157)

(c)

Figure 2.5: Classification results for (a) random noise; (b) Jackson Pollock "Rabit Hole"; (c) Muhammad Ali towering over Sonny Liston (1965 rematch). These pictures are outliers for the Imagenet categories. The left labels of each picture are provided by our method and the right are the results of the Inception-v4 model.

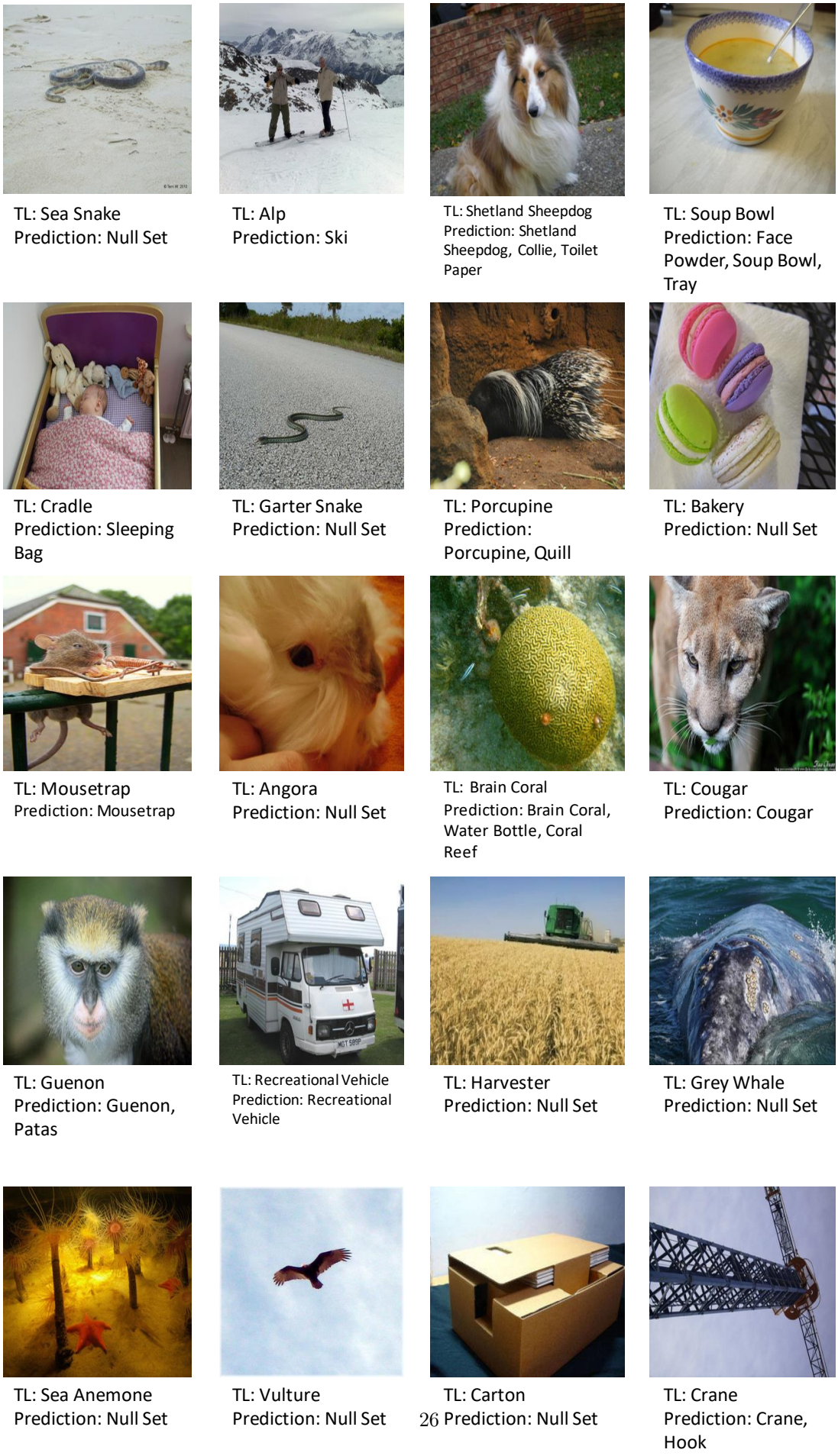
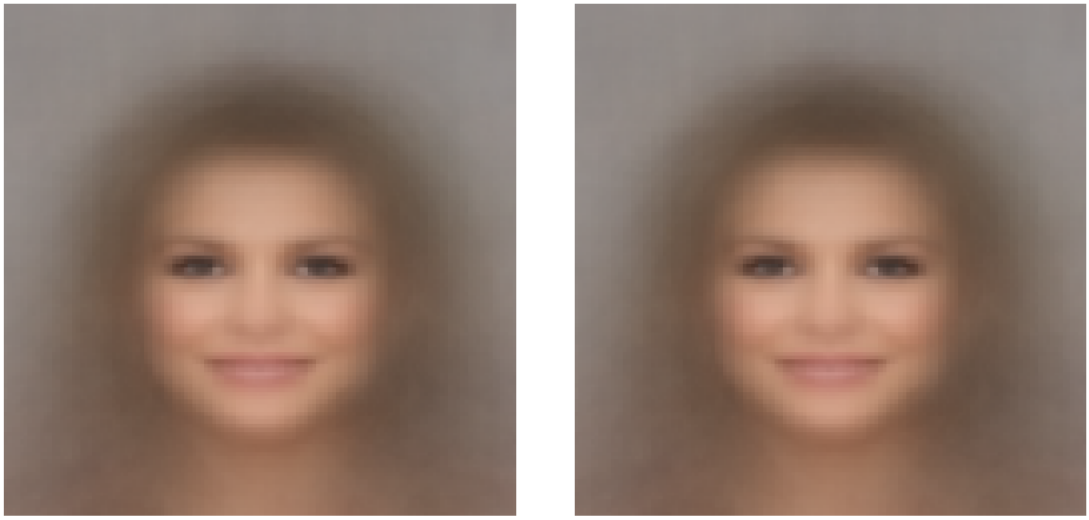


Figure 2.6: A collage of the first 20 images in the ImageNet validation set with $\alpha = 0.7$. *TL* denotes the image true label and *Prediction* is the method output. By design only 0.3 accuracy is expected, yet both the true and the false predictions are reasonable. Online version in color.



(a) (b)

Figure 2.7: (a) Females faces mean pixel values in (a) CelebA; (b) IMDB-Wiki. Within CelebA the pictures are aliened with fixed posture, explaining why it naively fails to generalize to IMDB-Wiki images.

The IMDB-Wiki results are not as good, but better than naively using a 0.577 accuracy classifier. Figure 2.8 show the classifier performance as a function of α. Both the accuracy and the number of false positives are tunable. For high values of $1 - \alpha$ the accuracy is much higher than 0.577, but would results in a large number of observations predicted as both genders. If cautious and conservative prediction is required small values of $1 - \alpha$ would guarantee smaller number of false predictions, but a large number of null predictions. The suggested conformal method provides a hyper-parameter controlling which type of errors are created according to the prediction needs, and works even in cases where standard methods fail.

2.8 Discussion

In this chapter we showed that conformal, set-valued predictors based on $\hat{p}(x|y)$ have very good properties. We obtain a cautious prediction associating an observation with a class only if the there is high probability of that observation is generated from the class. In most of the space the classifier predicts the null set. This stands in contrast to standard solutions which provide confident predictions for the entire space based on

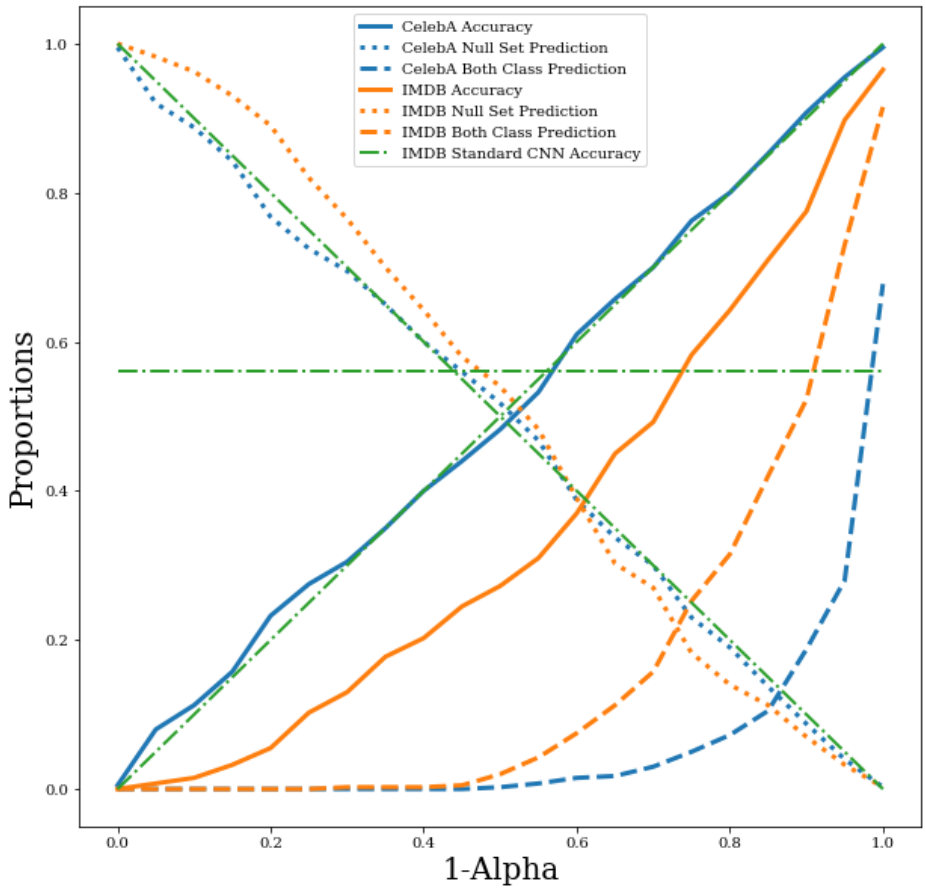


Figure 2.8: Performance plots for the conformal method on both CelebA and IMDB-Wiki.

data observed from a small area. This can be useful when a large number of outliers are expected or in which the distribution of the training data won't fully describe the distribution of the observations when deployed. Adversarial attacks are an important example of such scenarios. We also obtain a large set of labels in the set when the object is ambiguous and is consistent with many different classes. Thus, our method quantifies two types of uncertainty: ambiguity with respect to the given classes and outlyingness with respect to the given classes.

In addition, the conformal framework provides our method with its coverage guarantees and class adaptivity. It is straightforward to add and remove classes at any stage of the process while controlling either the overall or class specific coverage level of the method in a highly flexible manner, if desired by the application. This desired properties comes with a price. The distribution of $p(x|y)$ for each class is learned independently and the decision boundaries are indifferent to data not within the class. In case precise decision boundaries are more desired, complex estimation functions overcome this limitation and provide decision boundaries almost equivalent to standard methods.

During the deployment of the method, evaluation of a large number of kernel density estimators is required. This is relatively slow compared to current methods. This issue can be addressed in future research with more efficient ways to learn ordering-consistent approximations of $p(x|y)$ that can be deployed on GPU's.

Chapter 3

Level-set Regression

Chapter is partially based on Hechtlinger et al. [2019].

This chapter discusses conformal prediction methods for regression. The chapter starts with a conceptual discussion in section 3.1 that promotes the advantages of the joint distribution. Section 3.2 introduce the level-set regression, which is a cautious method for regression based on $p(x, y)$. Section 3.3 present Lebesgue regression, which extends chapter 2 to regression, and was the original motivation behind the chapter. Section 3.4 discusses other methods to obtain conformal prediction sets in regression settings. Section **??** conduct expriements and compare all methods.

3.1 Conceptual Discussion

Let (X, Y) be two random variables from a distribution P defined on $\mathcal{X} \times \mathcal{Y}$. In this section we conceptually discuss different possibilities to predict from an realization of X set of possible values of Y. Suppose we are given the joint distribution P by oracle. For a given $x \in \mathcal{X}$ and a predetermined coverage level $\alpha \in (0, 1)$ we construct prediction set in the form of

$$f_\psi(x) = \{y \mid \psi(x, y) > t_\alpha\}, \tag{3.1}$$

for some function ψ and the largest value of value t_α such that

$$p\left(Y \in f_\psi\left(X\right)\right) \geq 1 - \alpha. \tag{3.2}$$

This approach is more general than a point wise estimate, which can be obtained directly from ψ (for example) by $f_\psi\left(x\right) = \mathbb{E}\left[\psi\left(x, Y\right)\right]$ or $f_\psi\left(x\right) = \sup_y\left\{\psi\left(x, y\right)\right\}$.

In order to explore the behavior of different values of ψ we consider a simple scenario in which $\left(X, Y\right) \sim \mathcal{N}_2\left(0, \mathbb{I}_2\right)$. That is, both variables are independent standard normal. This is an extreme case used as a pathology to discuss regression modeling. Regression is designed to address the relation between X and Y. The independent case demonstrates possible conceptual challenges.

We analyze three concrete quantities for ψ: $p\left(y \mid x\right)$, $p\left(x \mid y\right)$ and $p\left(x, y\right)$. This main claim is that $p\left(x, y\right)$ offers significant advantages over the other quantities. This in contrast to the leading paradigm in modeling regression problems which utilizes $p\left(y \mid x\right)$.

First consider the scenario when $\psi\left(x, y\right) = p\left(y \mid x\right)$. Than the prediction interval from equation 3.1 is

$$f_\psi\left(x\right) = \left\{y \mid p\left(y \mid x\right) > t_\alpha\right\}. \tag{3.3}$$

This generalizes linear regression, which provides a point estimate value by approximating $\mathbb{E}\left[Y \mid x\right]$. Furthermore, many other methods to model regression problems learns an approximation of $\mathbb{E}\left[Y \mid g\left(X\right)\right]$, where $g\left(X\right)$ is model dependent transformation over the feature space. This abstraction includes generalized linear models, neural networks regression and support vector regression TO DO: REWRITE, VERIFY & CITE.

In the independent standard normal scenario, $p\left(y \mid x\right) = p\left(y\right)$. Therefore the prediction interval of equation 3.3 is a horizontal infinite strip such that $\forall x \in \mathcal{X}$

$$f_{p(y|x)}\left(x\right) = \left[\Phi^{-1}\left(\frac{\alpha}{2}\right), \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right], \tag{3.4}$$

where Φ^{-1} is the inverse of the standard normal CDF. Figure 3.1 (a) visualizes the prediction interval. The prediction is the same $\forall x \in \mathcal{X}$ regardless of the fact that X is centered at 0. The predictor is oblivious

to the distribution of X. At first, this might be seem inevitable since the variables are independent. But it is unnecessarily the case. As we show later, there is implicit information provided by the joint distribution ignored in models based on $p(y \mid x)$.

Additional problem is that this model is overconfident. The distribution P is dense around the origin. Nevertheless, the model extrapolates and provides an unsound prediction throughout the entire the space also in low density regions in which there is limited information. When facing an outlier, the prediction remains the same.

An alternative approach is to use $\psi(x, y) = p(x \mid y)$, as suggested in Hechtlinger et al. [2018] for classification purposes. In that case, under the independent standard normal scenario, $p(x \mid y) = p(x)$. The prediction interval of equation 3.3 is a vertical infinite strip

$$f_{p(x|y)}(x) = \begin{cases} \infty & x \in \left[\Phi^{-1}\left(\frac{\alpha}{2}\right), \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\right] \\ \emptyset & x \notin \left[\Phi^{-1}\left(\frac{\alpha}{2}\right), \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\right]. \end{cases}$$

(3.5)

Figure 3.1 (b) provides a visualization. This predictor is cautious. When facing an outlier, by design it avoids providing a prediction and output the empty set. But it has a major drawback. For values from the distribution P the prediction is infinitely long.

Defining $\psi(x, y) = p(x, y)$ results with the x-section of the $1-\alpha$ contour of the joint distribution, specifically

$$f_{p(x,y)}(x) = \{y \mid p(x, y) \geq t_\alpha\}. \tag{3.6}$$

Figure 3.1 (c) visualize this predictor. It's both accurate when predicting observations from the distribution P and cautious when predicting outliers. It fully utilize the joint relation between X and Y to provide a

prediction. The prediction interval is adaptive. The largest interval is when $x = 0$ and it's continuously shrinking until it predicts the null set for most of the space.

It is possible to construct different distributions of X and Y that has the exact results when $\psi(x, y)$ is either $p(y \mid x)$ or $p(x \mid y)$. For example for when both variables are independent uniforms on the square

$$(X, Y) \sim U\left[\Phi^{-1}\left(\frac{\alpha}{2}\right), \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right] \times U\left[\Phi^{-1}\left(\frac{\alpha}{2}\right), \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right]. \tag{3.7}$$

The uniqueness of the outcome is dependent on the joint distribution and changes when $\psi(x, y) = p(x, y)$. In that sense, using $p(x, y)$ is utilizing additional information provided by the distribution.

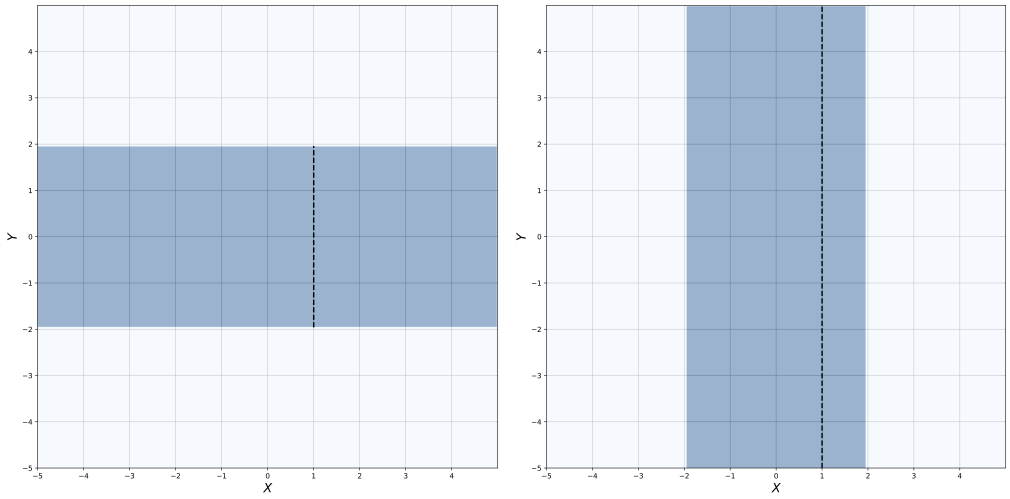
3.2 Level-set Regression

Level-set regression follows from the discussion in section 3.1 and was originally developed in Lei and Wasserman [2014]. It suggests using $p(x, y)$ and specifically equation 3.6 as an alternative way to model regression problems. This is done by finding

$$f_{p(x,y)}(x) = \{y \mid p(x, y) \geq t_\alpha\}. \tag{3.8}$$

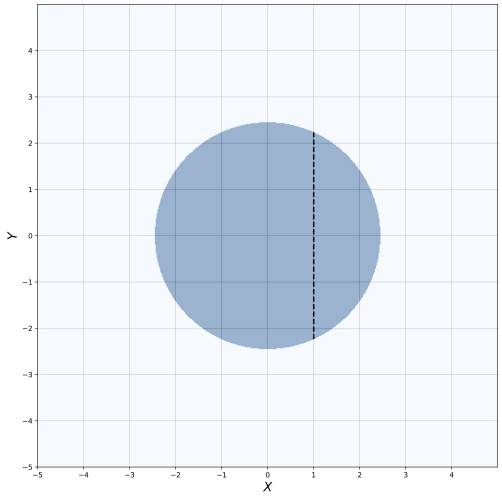
In practice $p(x, y)$ isn't known and has to be estimated. Level set estimation is an active research problem with different approaches. In this work we use the split conformal prediction approach suggested in Lei et al. [2013]. The method provides a distribution free, exact, finite sample coverage. Specifically for any density estimator $\hat{p}(x, y)$ of $p(x, y)$ it uses n independent data observations $(X_1, Y_1), \ldots, (X_n, Y_n)$ to estimates a prediction set $\hat{C}_n \in \mathcal{X} \times \mathcal{Y}$. This prediction set has exact coverage such that for a new observation (X_{n+1}, Y_{n+1}) it happens that

$$P\left((X_{n+1}, Y_{n+1}) \in \hat{C}_n\right) \geq 1 - \alpha. \tag{3.9}$$



(a) Prediction interval based on $p(y \mid x)$

(b) Prediction interval based on $p(x \mid y)$



(c) Prediction interval based on $p(x, y)$

Figure 3.1: Prediction intervals for $f_\psi(x) = \{y \mid \psi(x, y) > t_\alpha\}$ when both X and Y are independent standard normal. The dotted black line demonstrates the interval for $x = 1$ in each one of the cases. (a) shows $\psi(x, y) = p(y \mid x)$ which generalize the standard regression method. It provide a horizontal infinite interval that extrapolates throughout the space although the distribution is dense around the origin. (b) $\psi(x, y) = p(x \mid y)$. It cautious and predict the null set for outliers but provide a vertical infinite intervals for values near the origin. (c) shows $\psi(x, y) = p(x, y)$ which fully utilize the information provided by the joint density. It's cautious and predicts in most of the space the null set.

The results holds regardless of the data distribution or even if $\hat{p}$ is not a consistent estimator. The set $\hat{C}_n$ is calculated empirically from the n independent data points. Specifically, we denote $\hat{t}_\alpha$ as the $1-\alpha$ empirical quantile of the data evaluated on $n+1$ points, that is

$$\hat{t}_\alpha = \sup_{t \in \mathbb{R}^+} \left\{ \frac{\sum_{i=1}^{n} I\left\{\hat{p}\left(X_i, Y_i\right) \geq t\right\}}{n+1} \geq 1-\alpha \right\}. \tag{3.10}$$

Then

$$\hat{C}_n = \left\{(x,y) \mid \hat{p}\left(x,y\right) \geq \hat{t}_\alpha\right\}, \tag{3.11}$$

and for a new x the Level Set Regression provides a prediction for a new x in the form of

$$\hat{f}_{\hat{C}_n}\left(x\right) = \left\{y \mid (x,y) \in \hat{C}_n\right\} = \left\{y \in \mathcal{Y} \mid \hat{p}\left(x,y\right) \geq \hat{t}_\alpha\right\}. \tag{3.12}$$

Due to the properties of the empirical quantile and the conformal prediction method, the set obtains the coverage on equation 3.9. Algorithm 3 summarizes the training, Algorithm 4 summarizes the prediction. In 3.2.1 we discuss efficient ways to calculate $\hat{f}_{\hat{C}_n}$.

Algorithm 3 Training Algorithm

Input: Density estimator $\hat{p}\left(x,y\right)$, n independent observations $\{(X_i, Y_i)\}_{i=1}^{n}$, Confidence level α.
Evaluate:
$\forall i \in [1:n]$, $\psi_i \leftarrow \hat{p}\left(X_i, Y_i\right)$
$\hat{t}_\alpha \leftarrow \text{Quantile}\left(\left(\psi_1, \ldots, \psi_n\right), \alpha\right)$ ▷ The empirical quantile for $n+1$ observations
Return: $\hat{t}_\alpha$

Algorithm 4 Prediction Algorithm

Input: Input to be predicted $x \in \mathcal{X}$, density estimator $\hat{p}\left(x,y\right)$, learned $\hat{t}_\alpha$
Calculate: $\hat{f}_{\hat{C}_n}\left(x\right) = \left\{y \in \mathcal{Y} \mid \hat{p}\left(x,y\right) \geq \hat{t}_\alpha\right\}$
Return $\hat{f}_{\hat{C}_n}\left(x\right)$

3.2.1 Density Estimation and Computational Considerations

The method validity follows from the conformal approach and is valid with any density estimator $\hat{p}\left(x,y\right)$. It does not require this estimator to be consistent. This enables the usage less common density estimators.

The experiments are done with the Gaussian kernel density estimator (KDE) and the nearest neighbor estimator. The KDE is a consistent estimator, but it has limitations. First, it often requires careful tuning of hyper-parameters, which can be difficult in high dimensions. Second, the computation of the level-set in equation 3.12 require large number of evaluations over a grid. This can result with a slow algorithm when a finer prediction is required.

On the other hand, the nearest neighbor estimator is not consistent, but offers the following computational advantage. We use the distance from the $\mathcal{L}_2$ nearest neighbor,

$$\Psi(x,y) = \min_{i\in[1:n]} \left\{ \sqrt{\|x - x_i\|_2^2 + \|y - y_i\|_2^2} \right\}. \tag{3.13}$$

When $\mathcal{X} = \mathbb{R}^p$, $1/\Psi(x,y)$ is (proportional to) the value of the nearest-neighbor estimator of the density $p(x,y)$. This density estimator is not consistent, but as shown next it provides good computational advantages and works in practice on applied data. In that case equation 3.12 can be stated as

$$\hat{f}_{\hat{C}_n}(x) = \left\{ y \in \mathcal{Y} \mid \Psi(x,y) \leq \hat{t}_\alpha \right\}. \tag{3.14}$$

Further simplifying this expression results with

$$\begin{aligned} \hat{f}_{\hat{C}_n}(x) &= \left\{ y \in \mathcal{Y} \mid \exists i \in [1:n], \text{ s.t, } \|x - x_i\|_2^2 + \|y - y_i\|_2^2 \leq \hat{t}_\alpha^2 \right\}, \\ &= \left\{ y \in \mathcal{Y} \mid \exists i \in [1:n], \text{ s.t, } \|y - y_i\|_2^2 \leq \hat{t}_\alpha^2 - \|x - x_i\|_2^2 \right\} \end{aligned} \tag{3.15}$$

which follows from the fact that the minimum is not explicitly needed and suffices for any observation to be near enough. This provides a computational advantage since equation 3.15 is directly computed from the observations instead of evaluating it for all $y \in \mathcal{Y}$. In particular it results with the alternative definition of $\hat{f}_{\hat{C}_n}(x) = \bigcup_i A_i(x)$ when

$$A_i(x) = \begin{cases} \left[y_i - \sqrt{t^2 - \|x - x_i\|_2^2},\ y_i + \sqrt{t^2 - \|x - x_i\|_2^2} \right] & t^2 \geq \|x - x_i\|_2^2 \\ \emptyset & \text{Otherwise,} \end{cases} \tag{3.16}$$

which is straightforward to compute.

3.3 Lebesgue Regression

Lebesgue regression is a conformal extension to regression suggested in Hechtlinger et al. [2019]. It extends the level-set method for classification discussed in chapter 2. The level-set classification assign a prediction to a specific class iff $p(x \mid y) \geq t_{\alpha,y}$ for some threshold $t_{\alpha,y}$. Lebesgue regression achieve this property by discretization.

Let $b_1, \dots b_K$ be K values defining a partition of $\mathcal{Y}$ into $K+1$ different bins such that $\mathcal{Y} = \biguplus_{k=0}^{K} B_k$, $B_k = [b_k, b_{k+1})$ and $B = \{B_k\}_{k=0}^{K}$. For completeness, we denote $b_0 = -\infty$, $B_0 = (-\infty, b_1)$, $b_{K+1} = \infty$ and $B_K = (b_K, +\infty)$. Then each of the bins is treated as a class, in a similar way to chapter 2. A prediction is assigned to a specific class k iff $p(x \mid b_k) \geq t_{\alpha,b_k}$ for some threshold t_{α,b_k}. If the partition is fine enough, each of the bins converges to an infinitesimal approximation of $p(x \mid y)$, which is the motivation behind the name.

Lebesgue regression was developed prior to the conceptual discussion in section 3.1 and was the main motivation leading to it. As discussed there, it is preferable to use $p(x, y)$ over $p(x \mid y)$ and $p(y \mid x)$. Hence the transition from Lebesgue regression to level-set regression.

The advantage that the binning approach offers is mostly computational. It makes the prediction process an evaluation of $K+1$ bins per input. In some scenarios, and for different density estimators than the nearest-neighbor, this can be significant improvement. In that case the evaluation is equivalent to $\{b_k \mid p(x, b_k) \geq t_{\alpha,b_k}\}$ since $p(x, b_k) = p(x \mid b_k)\, p(b_k)$. This is just local version of level-set regression per bin. It is also possible to evaluate in a straightforward manner $\{b_k \mid p(x, b_k) \geq t_\alpha\}$, which is a discretized version of level-set regression.

3.4 Related Work

3.4.1 Residuals Based Intervals

The standard approach to generate prediction intervals to regression problems using the conformal prediction framework is based on the residuals [Vovk et al., 2005, Lei et al., 2018].

The split conformal approach constructs a prediction interval for a regression function $\hat{f} : \mathcal{X} \to \mathcal{Y}$ using n exchangeable residuals. Denote $\epsilon_i = \left| y_i - \hat{f}\left(x_i\right) \right|$ as the i'th residual. Denote $\hat{t}_\alpha$ as the $1-\alpha$ quantile rounded up. That is

$$\hat{t}_\alpha = \inf_{t \in \mathbb{R}^+} \left\{ \frac{\sum_{i=1}^{n} \mathbb{I}\left\{\epsilon_i \leq t\right\}}{n+1} \geq 1-\alpha \right\}. \tag{3.17}$$

Than for a new observation (X_{n+1}, Y_{n+1}) from the same distribution ϵ_{n+1} is exchangeable with all the other residuals. It means that

$$p\left(\epsilon_{n+1} \leq \hat{t}_\alpha\right) \geq 1-\alpha, \tag{3.18}$$

and it follows that

$$p\left(\left|Y_{n+1} - \hat{f}\left(X_{n+1}\right)\right| \leq \hat{t}_\alpha\right) = p\left(Y_{n+1} \in \left[\hat{f}\left(X_{n+1}\right) - \hat{t}_\alpha, \hat{f}\left(X_{n+1}\right) + \hat{t}_\alpha\right]\right) \geq 1-\alpha. \tag{3.19}$$

This approach is general, quick to use and applicable on all regression functions regardless of the data distribution. The main disadvantage of this approach is that it provides the same length compact interval for all $x \in \mathcal{X}$, which might be unnecessary long when heteroscedasticity is preset.

3.4.2 Locally Adaptive Residuals

The prediction intervals in 3.4.1 are of a fixed length. In Lei et al. [2018] the authors construct residual based intervals with varying length. This is done by redefining the residuals and estimating the residuals variance as a function of the input. In specific the authors define the residuals to be

$$\epsilon_i = \frac{\left| y_i - \hat{f}\left(x_i\right) \right|}{\hat{\rho}_y\left(x_i\right)}, \tag{3.20}$$

where $\hat{\rho}_y\left(x_i\right)$ is an estimator of the conditional mean absolute deviation (MAD) of $Y - \hat{f}\left(X\right) \mid X = x$, as a function of $x \in \mathcal{X}$. The rest of the conformal prediction methodology follows as before. The advantage of this approach is that it adjust the prediction interval length to heteroscedasticity when preset.

3.4.3 Conformalized Quantile Regression

Recently, Romano et al. [2019] suggested conformalized quantile regression (CQR). The method uses residuals generated by quantile regression estimators to create prediction intervals with varying length adjusting to heteroscedasticity when present.

Specifically, instead of using the residuals of any $\hat{f}$ regressor as in 3.4.1, the method estimates $\hat{q}_{\alpha_{lo}}\left(x\right)$ and $\hat{q}_{\alpha_{hi}}\left(x\right)$ which are the $\alpha/2$ and $1-\alpha/2$ conditional quantile functions. That is, $\hat{q}_\alpha$ estimate the conditional quantile function

$$q_\alpha\left(x\right) = \inf\left\{y \in \mathbb{R} \mid F\left(y \mid X = x\right) \geq \alpha\right\}, \tag{3.21}$$

where F is the conditional distribution CDF. The conformal prediction interval is constructed using the empirical $1-\alpha$ quantile of the quantile functions residuals

$$E_i = \max\{\hat{q}_{\alpha_{lo}}\left(X_i\right) - Y_i,\ Y_i - \hat{q}_{\alpha_{hi}}\left(X_i\right)\}. \tag{3.22}$$

The intuition is that positive values of E_i measure the magnitude of the error of either of the quantile functions, while negative values reflect observations in which the error was bounded between both quantiles estimation. Therefore, selecting the $1-\alpha$ empirical quantile of 3.22 control the errors of the observations around the quantiles prediction.

Since the difference between $\hat{q}_{\alpha_{lo}}$ and $\hat{q}_{\alpha_{hi}}$ changes as a function of x and the heteroscedasticity, the CQR provide prediction intervals with varying length according to the data. The disadvantage of the CQR method

is that it provide a single compact prediction interval. When the response variable is not unimodal it might result with long prediction intervals.

3.4.4 Multi-Modal Regression

Multi-Modal Regression [Chen et al., 2016] estimates the local modes of the response variable Y given the covariate $X = x$. This is in contrast to standard regression that usually estimate the conditional mean. The method is especially useful when the conditional density function has multiple local modes, such as when the $X - Y$ relation contains multiple patterns.

The modes in the Multi-Modal regression are obtained by estimating the local maximum of the conditional distribution of $p(y \mid x)$. Specifically, the method approximate

$$M(x) = \left\{ y \mid \frac{\partial}{\partial y} p(y \mid x) = 0,\ \frac{\partial^2}{\partial y^2} p(y \mid x) < 0 \right\}. \tag{3.23}$$

Since for a given x, $p(y \mid x) \propto p(x, y)$, often in practice the local maximum are found by estimating the joint distribution. Given a multi-modal regressor $\hat{M} : \mathcal{X} \to 2^{\mathcal{Y}}$ the split conformal method can be used to construct prediction sets. This is done by approximating the residuals quantile, similarly to 3.4.1, with the residuals now defined as

$$\epsilon_i = \min_{\hat{y} \in \hat{M}(x_i)} \{|y_i - \hat{y}|\}. \tag{3.24}$$

Since the multi-modal regression provides multiple predictions for a given x, the prediction intervals often include multiple disconnected intervals. The resulting intervals can be preferable when the data is better modeled with multi-modal regression than unimodal. Since the method is based on a fixed residual value for each mode, when there is heteroscedasticity between the different modes the prediction intervals might be longer in some cases.

Name	Method
Ridge	Residual based prediction intervals for ridge regression, regularized using cross validation.
Net	Residual based prediction intervals for fully connected neural network. Network has 3 layers, each layer 64 hidden variables.
RF	Random forest regression implemented using Python sklearn default parameters with maximum 1000 trees.
{Ridge, Net, RF}-L	Same methods as above with localized prediction intervals as explained in 3.4.2.
CQR {Net, RF}	Conformalized quantile regression, as in Romano et al. [2019]. Intervals are constructed using {Neural Networks, Random Forest} quantile estimation.
Level-set {1-NN, KDE}	Level-set prediction intervals based on $\hat{p}(x, y)$. The density estimator is learned using {Closest Nearest Neighbor, Kernel density estimation}.
Level-Set DNN Features {1-NN, KDE}	The level-set method, only trained on features first learned through a deep neural network, as described in 3.5.2.

Table 3.1: Details regarding the different methods implemented in the experiments section.

3.5 Simulations and Examples

This section presents experiments results comparing the different methods on multiple datasets. Table 3.1 provide details regarding the implementation of the different methods. For standardization purposes the input variables are normalized and the output variable is scaled by dividing it by its mean absolute value.

3.5.1 $D31$ Experiment

$D31$ is a synthetic dataset used to benchmark clustering algorithms. It contains 3100 observations generated from 31 different independent $2D$ Gaussian variables, 100 observations per variable. The dataset has a complicated spatial relation between the data points as demonstrated in Figure 3.2. When using the dataset for clustering the goal is to recover the different clusters. In this example we consider the problem as a simple regression problem predicting Y from X. Due to the unique structure of the data, any regression method that predicts a single output value may not be able to capture the structure of the data; the same consideration

applies to any method that outputs a convex prediction interval. A sensible solution for this data structure needs to contain multiple disjoint prediction sets, making it a useful dataset to exemplify our method.

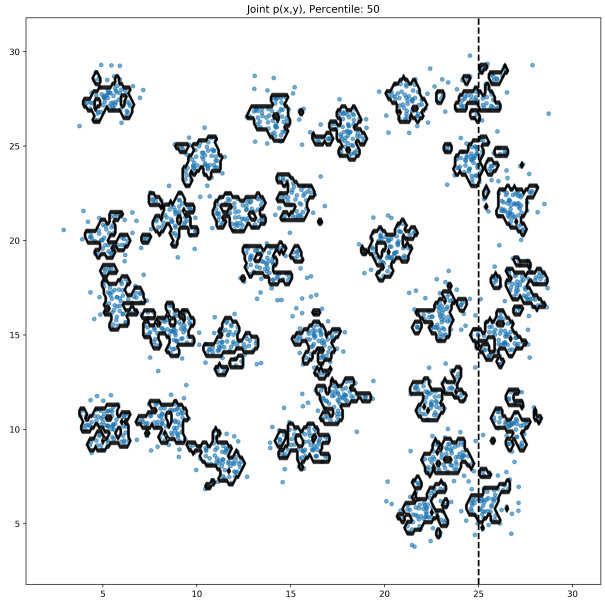
We assign 60% of the observation to the training of the density estimator. The remaining 40% are used to evaluate $\hat{t}_\alpha$ quantile. Since the problem is a univariate regression problem, we evaluate the prediction for different values over a grid.

Figure 3.2 (a) and (b) visualize the prediction results for coverage 0.5 and 0.95. For coverage of 0.5, the prediction intervals cover almost perfectly the area within the space that contains the relevant observations. For coverage of 0.95 the sets are larger but still sensible given the dataset. In both cases, values of x outside the scope of the dataset are predicted as the null set. For comparison, Figure 3.2 (c) shows the fit of an RBF kernel SVM to the dataset with 0.5 conformal prediction intervals suggested by Lei and Wasserman [2014]. Although those intervals for valid, they fail to capture the structure of the data. Figure 3.2 (d) shows the prediction intervals from multi-modal regression suggested by Chen et al. [2016]. These intervals are constructed by estimating the residuals to the closest predicted mode and calculating the appropriate conformal threshold. The intervals capture the structure of the data, but extrapolate throughout the space since the modal regression provides a prediction for all $x \in \mathcal{X}$.

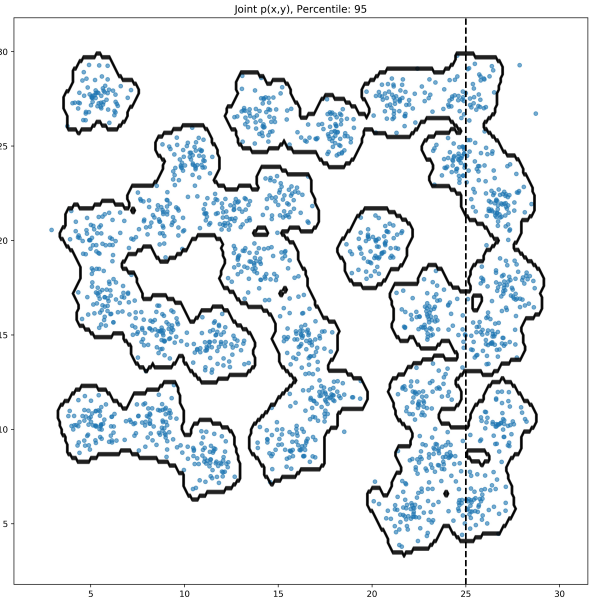
Figure 3.3 show the average prediction interval length for residual based methods and the level-set method over 20 random repetitions. Figure 3.4 validate all methods obtain the expected 0.9 coverage. The prediction length for the level-set method using the nearest neighbor metric is must shorter than the rest, demonstrating the method is able to address the data structure.

3.5.2 Merck Experiment

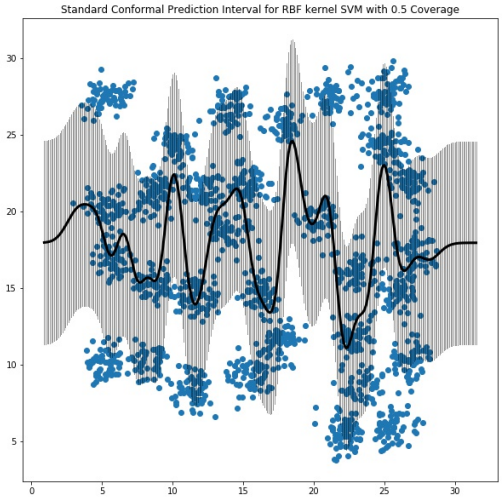
The Merck molecular activity dataset Ma et al. [2015] is a Kaggle challenge which is based on 15 molecular activity data sets. The goal is to predict activity levels for different molecules based on features extracted from the different atoms in the molecule. Each dataset is high dimensional and the features are highly correlated, making it difficult to obtain accurate predictions. It is regarded a challenging high dimensional regression problem. First, we demonstrate the method on the 'OX2' dataset which contains $14,875$ observations over



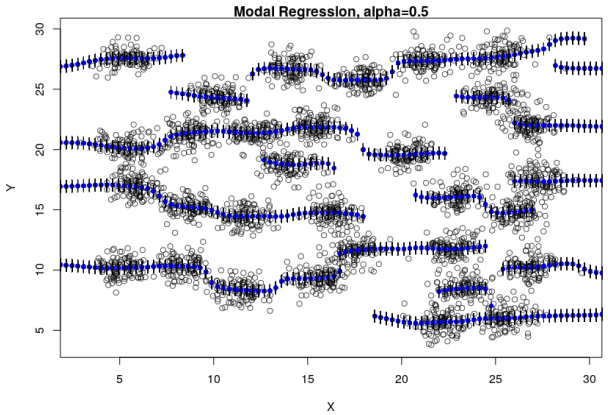
(a) Level Set Regression prediction interval $\alpha = 0.5$



(b) Level Set Regression prediction interval $\alpha = 0.95$



(c) Prediction interval based on SVM residuals, $\alpha = 0.5$



(d) Prediction interval based multi-modal Regression residuals, $\alpha = 0.5$

Figure 3.2: Level Set Regression prediction sets on the $D31$ dataset for $1 - \alpha$ coverage of (a) 0.5; (b) 0.95. For a given x, the prediction set of y is the set of intervals along the corresponding vertical strip. The dotted line intersection with the counturs demonstrates the prediction for $x = 25$. Values of x outside the range of the data give null set prediction sets. (c) Prediction intervals based on the residuals of a SVM model ($\alpha = 0.5$). These intervals are valid but miss the structure of the data. (d) Prediction intervals based on the residuals of a nonparametric multi-modal regression from Chen et al. [2016] ($\alpha = 0.5$). These intervals capture the structure but extrapolate throughout the space.

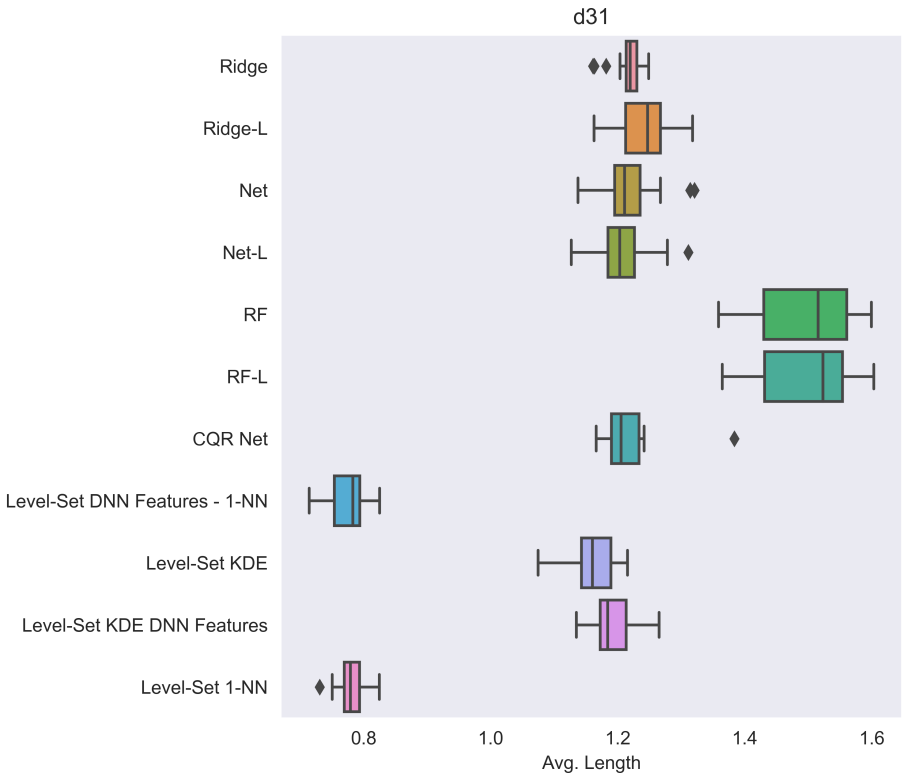


Figure 3.3: Average prediction set length on the D31 dataset. Due to the multi-modal structure of the data the level-set methods with the nearest neighbor density estimation provides the best results by a large margin. The box plots represent the distribution of 20 random repetitions.

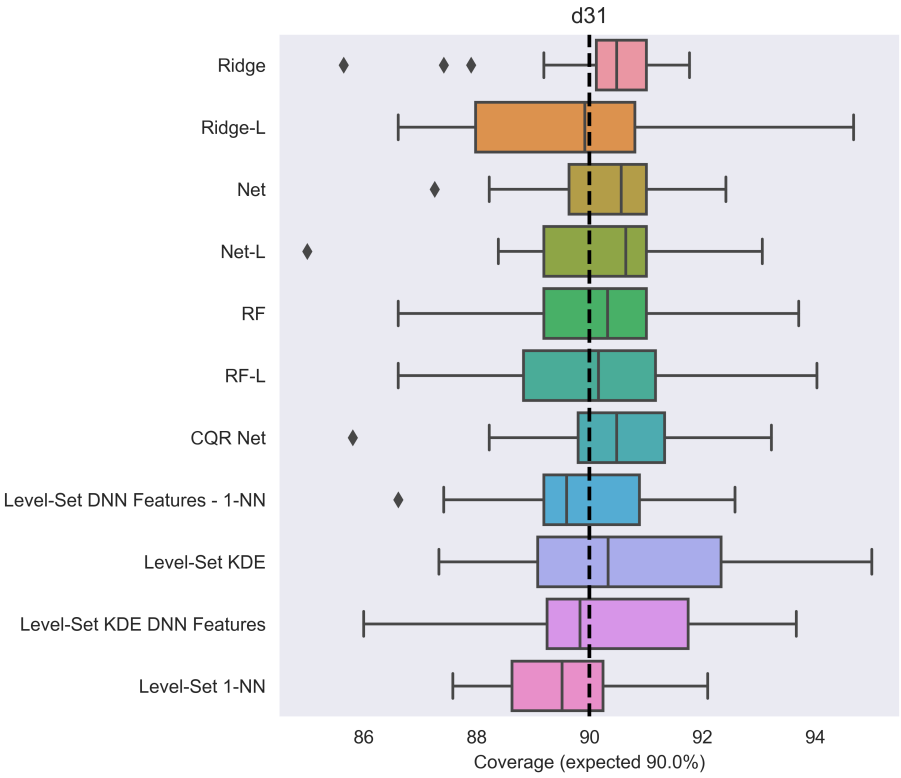


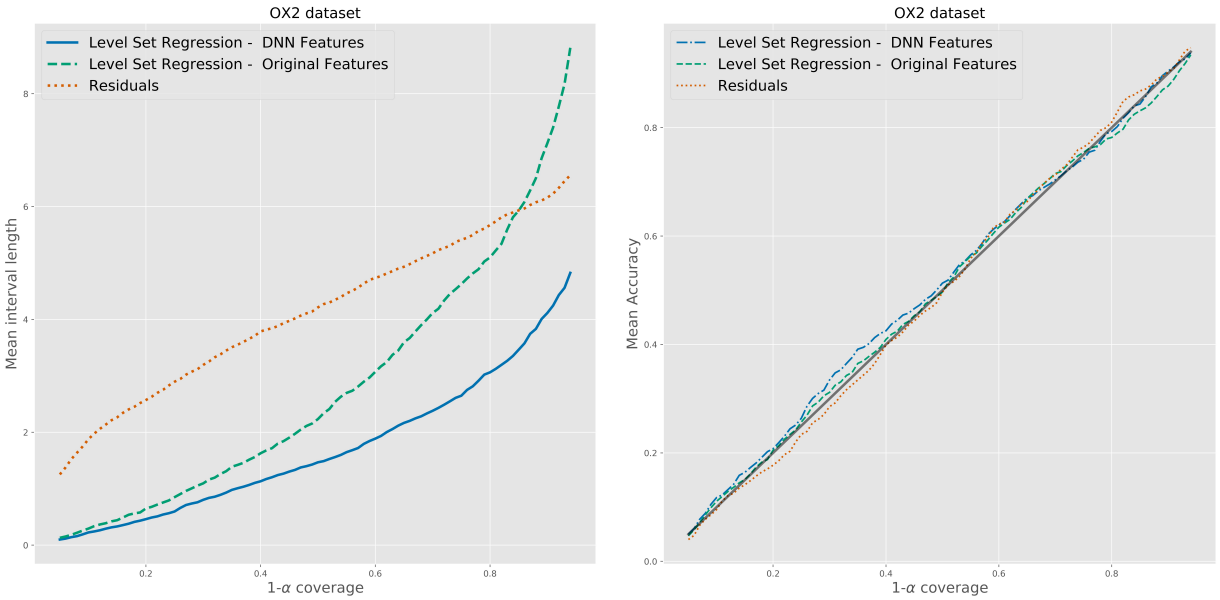
Figure 3.4: Empirical coverage on the D31 dataset. The box plots represent the distribution of 20 random repetitions. All conformal methods are valid.

$5,464$ features. 'OX2' was selected since it's a large, high-dimensional dataset that provided relatively good empirical results in comparison to the other datasets. Then we report results on other datasets.

The purpose of this experiment is two-fold. First we demonstrate that the method works well in high dimensional settings. Second we show that Level Set Regression can be built on top of current methods. For this we train a deep neural network (DNN) with 3 fully connected layers and corresponding dropout and activation layers on an independent partition of the dataset. We then use this network to extract features from the within and outwith sets and compare the performance of Lebesgue Regression when using the DNN features, the original features and standard residual based prediction intervals.

We partition the data into four sets: training, within, outwith and validation, where roughly half of the data is used for the training and the rest is split between the other sets. The training data is used to train the DNN and extract features. The Level Set Regression is trained using the within and outwith. As discussed in 3.2.1, the within set is used to learn the nearest neighbor joint density estimator and the outwith is used to find the proper thresholds. The performance is evaluated on the validation set.

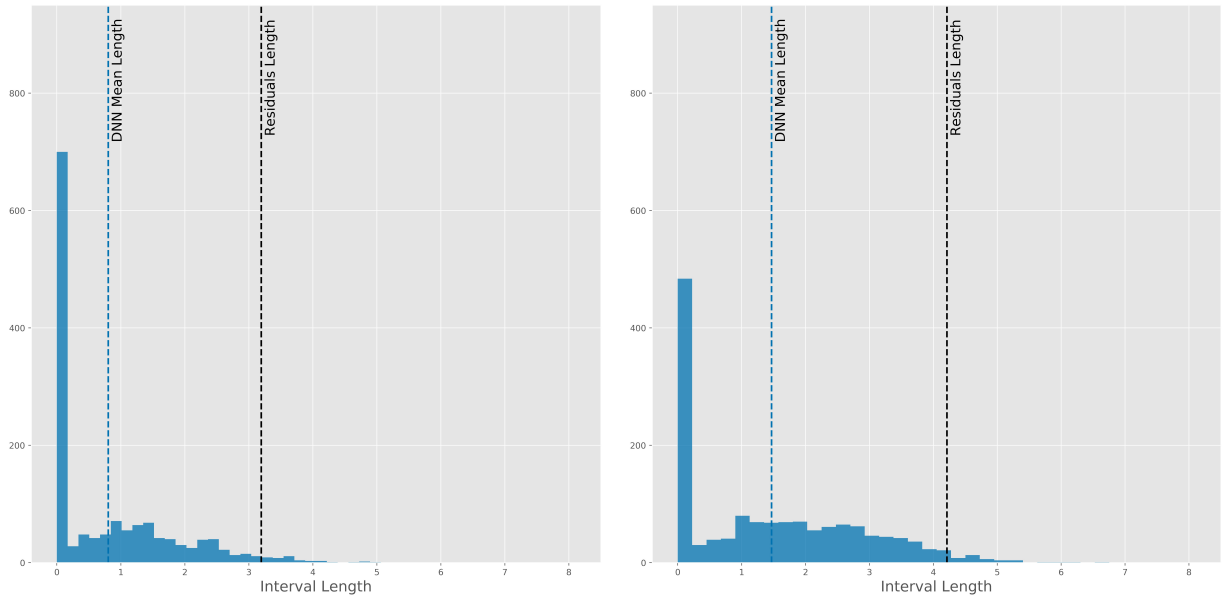
Figure 3.5 (a) show the mean interval length for each one of the methods. The Level Set Regression trained on the DNN features provides significantly shorter intervals. Figure 3.5 (b) confirms that the methods achieves the claimed $1 - \alpha$ coverage. Figure 3.6 provide several histograms showing the distribution of the intervals for different coverage levels.



(a) Mean Interval Length

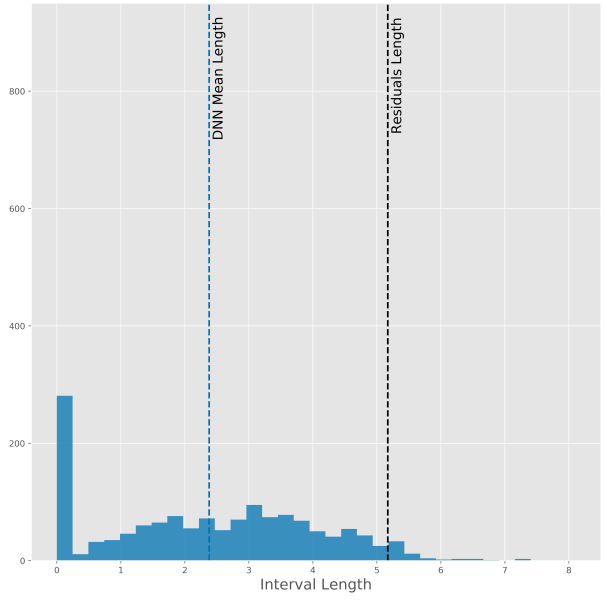
(b) Coverage

Figure 3.5: *Convergence plots for the Merck 'OX2' dataset comparing the method using the original features and features learned by a DNN with standard residual based prediction intervals. (a) shows the mean interval length as a function of* $1 - \alpha$*. (b) shows that the empirical coverage of all methods matches the expected theoretical coverage.*

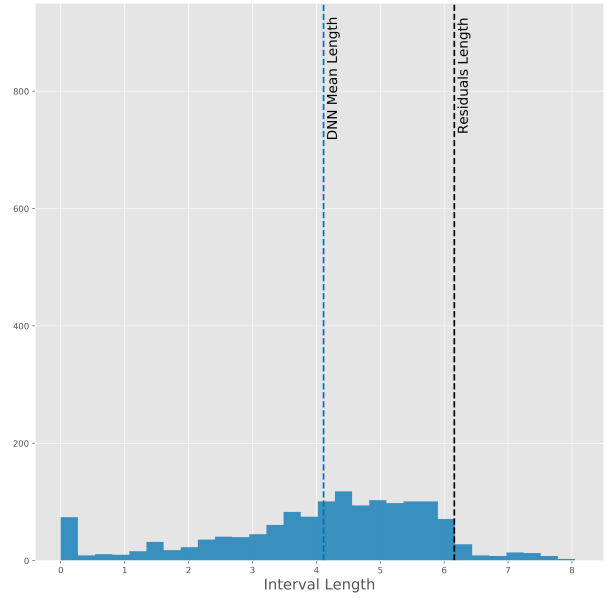


(a) $1 - \alpha = 0.3$

(b) $1 - \alpha = 0.5$



(c) $1 - \alpha = 0.7$



(d) $1 - \alpha = 0.9$

Figure 3.6: Histograms of the prediction interval length for the OX2 dataset using the joint distribution of the features learned by DNN. The vertical lines show the mean of the Level Set Regression method and the length of the residual prediction interval.

Chapter 4

Graph Convolution

Section based on Hechtlinger et al. [2017]

Convolutional Neural Networks (CNNs) are a leading tool used to address a large set of machine learning problems (LeCun et al. [1998], LeCun et al. [2015]). They have successfully provided significant improvements in numerous fields, such as image processing, speech recognition, computer vision and pattern recognition, language processing and even the game of Go boards (Krizhevsky et al. [2012], Hinton et al. [2012], Le et al. [2011], Kim [2014], Silver et al. [2016] respectively).

Since the standard convolution layer can only be applied to grid-structured input, recent years have seen a surge of research on the generalization of CNNs to other geometrical structures, in particular to graphs and manifolds. Many of the recent advances have been extensively reviewed by Bronstein et al. [2016],

The main contribution of this work is a novel generalization of CNNs to general graph-structured data, directed or undirected, called the Nearest Neighbors Convolutional Neural Network (NNCNN). We propose a novel spatial convolution that selects the top p closest neighbors of every node using a random walk as seen in Figure 4.1. Then for every node, the convolution is computed as the inner product of the weights and the selected p closest neighbors, which are ordered according to their relative position from the given node. This allows the usage of the same set of weights (shared weights) for the convolution at every node and reflects the dependency between each node and its closest neighbors.

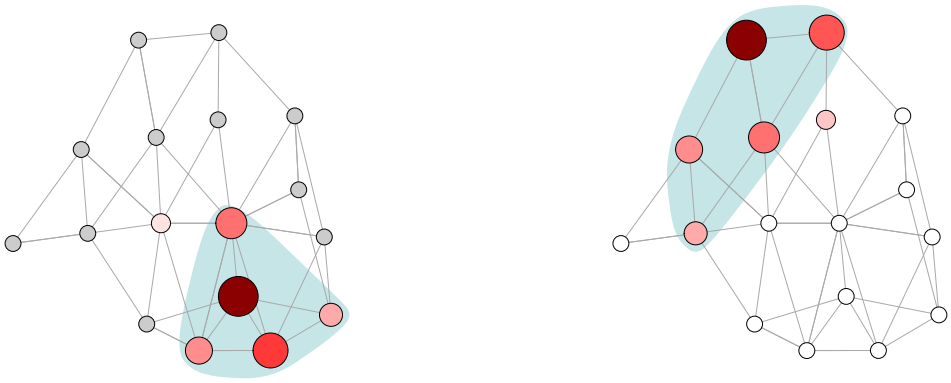


Figure 4.1: Visualization of the graph convolution size 5. For a given node, the convolution is applied on the node and its 4 closest neighbors selected by the random walk. As the right figure demonstrates, the random walk can expand further into the graph to higher degree neighbors. The convolution weights are shared according to the neighbors' closeness to the nodes and applied globally on all nodes.

For an image which can be considered as an undirected graph with edges between neighboring pixels, this convolution operation is the same as the standard convolution. The proposed convolution analogous to standard CNNs, offers a supervised algorithm that incorporates the structural information present in a graph. Furthermore, it can be applied to a wide range of standard regression and classification problems by first estimating the graph structure in the data.

4.1 Literature Review

Currently there are two main approaches generalizing CNNs to graph structured data, spectral and spatial approaches Bronstein et al. [2016]. The spectral approach generalizes the convolution operator using the eigenvectors derived from the spectral decomposition of the graph Laplacian. The motivation is to create a convolution operator that commutes with the graph Laplacian similar to the way the regular convolution operator commutes with the Laplacian operator. This was first studied by Bruna et al. [2013] and later extended by Henaff et al. [2015], Defferrard et al. [2016] and Kipf and Welling [2016].

A key drawback of the spectral approach is that it is not trivial to transfer it to another graph, as it learns filters that are functions of a particular graph Laplacian. This constrains the operation to be domain-dependent and restricts the transfer of knowledge between different domains.

The spatial domain methods generalize the convolution using the graph's spatial structure, capturing the essence of the convolution as an inner product of the parameters with spatially close neighbors. The main challenge with the spatial approach is that it is difficult to find a shift-invariance convolution for non-grid data. Duvenaud et al. [2015], Atwood and Towsley [2016], Atwood et al. [2017] and Monti et al. [2017] have made significant development on this front.

Spatial convolutions tend to be position dependent and generally lack meaningful global interpretations. The convolution proposed in this work is spatial and utilizes the relative distance between nodes to overcome this difficulty. In section 4.3 we review many of the current convolutional operators on graphs, from both spectral and spatial domains.

In addition to the research generalizing convolutions on graphs, there is active research on the application of different types of Neural Networks on graph-structured data. The earliest work in the field is the Graph Neural Network by Gori et al. [2005] and Scarselli et al. [2009]. The model connects each node in the graph with its first order neighbors and designs a recursive architecture inspired by recursive neural networks. Recently it has been extended by Li et al. [2015] to output sequences. Battaglia et al. [2016] introduce "interaction networks" studying spatial binary relations to learn objects and relations in physics. Gilmer et al. [2017] give a general model called Message Passing Neural Networks (MPNNs) that incorporates both convolutional and recurrent neural networks on graphs.

The problem of selecting nodes from a graph for a convolution is analogous to the problem of selecting local receptive fields in a general neural network. The work of Coates and Ng [2011] suggests selecting the local receptive fields in a feed-forward neural network, using the closest neighbors induced by the similarity matrix, with the weights not being shared among the different hidden units. Similarly, on a graph Niepert et al. [2016] select the receptive fields from a local neighborhood and then use it as the input of a regular 1D convolutional neural network, discarding the graph structure.

4.2 NNCNN Advantages

The nearest neighbors convolution suggested in this work possesses many desired advantages in comparison to other methods:

- **It is natural and intuitive.** The proposed CNN, similar to the standard CNN, convolves every node with its closest spatial neighbors, providing an intuitive generalization. For example, if we learn the graph structure using the correlation matrix, then selecting a node's p nearest neighbors is similar to selecting its p most correlated variables, and the weights correspond to the neighbors' relative position to the node (i.e. i^{th} weight globally corresponds to the i^{th} most correlated variable for every node).

- **It is transferable.** Since the criterion by which the p relevant variables are selected is their relative position to the node, the convolution is invariant to the spatial location of the node on the graph. This enables the application of the same filter globally across the data on all nodes on varying graph structures. It can even be transfered to different data domains, overcoming a known limitation of many other spatial generalizations of CNNs on graphs.

- **It is scalable.** Each forward call of the graph convolution requires $O(N \cdot p)$ flops, where N is the number of nodes in the graph or variables. This is also the amount of memory required for the convolution to run. Since $p \ll N$, it provides a scalable and fast operation that can efficiently be implemented on a GPU.

- **It is effective.** Experimental results demonstrate that by learning the graph structure for standard regression or classification problems, a simple application of NNCNN gives results that are comparable to state-of-the-art models.

4.3 Convolution Neural Networks on Graphs

In this section we explore the different proposed CNNs on an undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ with N nodes, $\mathcal{V} = (1, \ldots, N)$, a set of edges $\mathcal{E}$ and an adjacency matrix $\mathbf{W} = (W_{ij})$, where $W_{ij} = W_{ji}$, $W_{ij} > 0$ if

$(i,j) \in \mathcal{E}$ and $W_{ij} = 0$ if $(i,j) \notin \mathcal{E}$. Now in order to introduce the CNNs, we need to introduce some common notation.

The Graph Laplacian: The normalized graph Laplacian is an $N \times N$ symmetric, positive-semidefinite matrix given by $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is the degree matrix given by $\mathbf{D} = \sum_{j \neq i} W_{ij}$.

The eigendecomposition of the Laplacian can be given as $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_N)$ is a diagonal matrix of the eigen-values in decreasing order and $\mathbf{V} = (v_1, \ldots, v_N)$ is an orthogonal matrix with the corresponding eigen-vectors as its columns.

Graph Fourier Transform and Spectral Convolution: Given an input or a realization $\mathbf{f} = (f_1, \ldots, f_N)^T$ on the vertices of the graph $\mathcal{G}$, its graph Fourier transform is given by $\hat{\mathbf{f}} = \mathbf{V}^T\mathbf{f}$.

Given two inputs on the graph, $\mathbf{f}$ and $\mathbf{g}$, their spectral convolution is defined in terms of the dot-product of their Fourier transforms as

$$\mathbf{f} \star \mathbf{g} = \mathbf{V}\left(\hat{\mathbf{f}} \odot \hat{\mathbf{g}}\right) = \mathbf{V}\text{diag}(\hat{g}_1, \ldots, \hat{g}_N)\hat{\mathbf{f}}. \tag{4.1}$$

Transition Matrix for a Graph: Let us now consider a random walk on the graph. Let $\mathbf{P}$ denote the transition matrix, such that, P_{ij} is the probability to move from node i to j. Then the transition matrix, $\mathbf{P}$ can be given by (Lovász et al. [1996]),

$$\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}. \tag{4.2}$$

Spectral CNN: Bruna et al. [2013] uses a spectral convolutional layer that takes the spectral convolution given in (4.1), but only considers the eigen-vectors corresponding to the largest k eigen values of the unnormalized graph Laplacian (**DL**), $\mathbf{V}_k = (v_1, \ldots, v_k)$. Hence the convolution layer with input $\mathbf{f}$ and the weights $\mathbf{g}$ is given by:

$$\mathbf{f}^{out} = \xi\left(\mathbf{V}_k\text{diag}(\hat{g}_1, \ldots, \hat{g}_k)(\mathbf{V}_k^T\mathbf{f})\right), \tag{4.3}$$

where $\hat{\mathbf{g}} = (\hat{g}_1, \ldots, \hat{g}_k)^T = \mathbf{V}_k^T\mathbf{g}$ are the spectral multipliers representing the learnable filter and $\xi(\cdot)$ is a non-linear function (e.g. ReLU) applied on every vertex.

Smooth Spectral CNN: Henaff et al. [2015] observe that convolutional kernels are typically restricted spatially, enabling learning of parameters independent of N. So to achieve similar efficiency in the spectral domain, they propose using the spectral convolution given in (4.1) and restrict the class of spectral multipliers to be smooth by considering a smoothing kernel $\mathcal{K} \in \mathbb{R}^{N \times N_0}$, such as splines, and restricting

$$\hat{g}_i = \mathcal{K}\tilde{g}_i. \tag{4.4}$$

This reduces the number of parameters to a fixed N_0 instead of N.

Chebyshev Spectral CNN (ChebNet): To get a localized filter and make the convolutional layer computationally efficient, Defferrard et al. [2016] use Chebyshev polynomial filters for the spectral convolution. They parametrize the filter as:

$$g_\alpha(\mathbf{\Lambda}) = \sum_{j=0}^{r-1} \alpha_j T_j(\tilde{\mathbf{\Lambda}}), \tag{4.5}$$

where $\alpha \in \mathbb{R}^r$ is a vector of Chebyshev coefficients parameterizing the filter, $T_j(\tilde{\mathbf{\Lambda}})$ is the Chebyshev polynomial of order r, which can be computed by a recurrence relation, evaluated at $\tilde{\mathbf{\Lambda}} = 2\mathbf{\Lambda}/\lambda_1 - \mathbf{I}_N$, a diagonal matrix of scaled eigenvalues that lie in $[1, 1]$. This gives a localized filter that avoids the explicit computation of the eigenvectors of $\mathbf{L}$ and also reduces the computational complexity to $O(Nr)$ operations.

Graph Convolutional Network (GCN): Kipf and Welling [2016] extended ChebNet to a layer-wise linear formulation that allows building of deeper models. So they considered $r = 2$ and additionally assumed $\lambda_1 \approx 2$ and $\alpha = \alpha_0 = -\alpha_1$ to derive

$$g_\alpha(\mho) = \alpha \left(\mathbf{I} + \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}\right)\mathbf{f}. \tag{4.6}$$

Since they assume $\lambda_1 \approx 2$, for consecutive layers they propose to re-normalize the filter into $g_\alpha(\mho) = \alpha \left(\mathbf{I} + \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{W}}\tilde{\mathbf{D}}^{-1/2}\right)\mathbf{f}$, where $\tilde{\mathbf{W}} = \mathbf{I} + \mathbf{W}$ and $\tilde{\mathbf{D}} = \text{diag}\left(\sum_{j \neq i} \tilde{W}_{ij}\right)$. This allows multiple convolution layers.

Diffusion CNN (DCNN): Atwood and Towsley [2016] is a spatial method that performs a random walk (diffusion process) on the graph in order to select spatially close neighbors for the convolution. The proposed convolution layer uses the powers of the transition matrix $\mathbf{P}^0, \mathbf{P}, \ldots, \mathbf{P}^{r-1}$ to give an output

$$\mathbf{f}_{j,k}^{out} = \xi\left(g_{jk}\mathbf{P}^k\mathbf{f}_j^{in}\right), \tag{4.7}$$

for $j = 1, \ldots, p$ and $k = 1, \ldots, pr$, where $\mathbf{G} = (g_{jk})$ is the $p \times r$ matrix of weights. In practice, for dense graphs the number of nodes visited in i steps can be quite large, which might over-smooth the signal in dense graphs. This was addressed in Atwood et al. [2017] where they first make the graph sparse and then apply their method for scalable performance.

MoNet Monti et al. [2017] propose a generic spatial-domain framework for deep-learning on graphs and manifolds. Their main contribution is the introduction of pseudo-coordinates $\mathbf{u}(x, y)$, for a node x and its neighbor $y \in \mathcal{N}(x)$, where $\mathcal{N}(x)$ is a neighborhood of x and a weighting function $\mathbf{w}_{\mathbf{\Theta}}(\mathbf{u}) = (w_1(\mathbf{u}), \ldots, w_p(\mathbf{u}))$, where $\mathbf{\Theta}$ contains the learnable parameters. They then define a patch operator of the form $D_j(x)f = \sum y \in \mathcal{N}(x) w_j(\mathbf{u}(x, y))f(y)$, for $j = 1, \ldots, p$. The convolution is then given by,

$$f \star g = \sum_{j=1}^{p} g_j D_j(x) f. \tag{4.8}$$

They show that other deep learning methods (including the classical CNN on Euclidean domains, GCN and DCNN) are particular settings of their framework with appropriate $\mathbf{u}$ and $\mathbf{w}_{\mathbf{\Theta}}(\mathbf{u})$. They particularly recommend $w_j(\mathbf{u})$ to be the Gaussian kernel with mean vector, $\mathbf{u}_j$ and a diagonal covariance matrix, $\mathbf{\Sigma}_j$.

4.4 Nearest Neighbors Graph CNN

The main contribution of this work is a Nearest Neighbors convolution that uses the p - nearest neighbors of a node defined using any similarity matrix $\mathbf{S}$, along with the shared weights assumption to do the same.

In this section, we discuss the operation of the convolution on a single layer in a single graph. It is easy to extend the definition to larger tensors, as will be explicitly explained in Section 4.4.6.

4.4.1 Graph Nearest Neighbors

For $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, let $S(i,j)$ be any similarity function quantifying the relationship between the nodes i and j on the graph such that $S(i,j) > 0$. For example, we can just consider the entries in the adjacency matrix, $S(i,j) = W_{ij}$.

Then for all nodes $i \in \mathcal{V}$, the function $\mathbf{S}_i(j) \equiv S(i,j)$ induces a relative ordering on the nodes in the graph with respect to i. For simplicity, let $n_i(k)$ denote the the k^{th} nearest node to the node i. Then for every node i,

$$\mathbf{S}_i(n_i(1)) \geq \mathbf{S}_i(n_i(2)) \geq \ldots \geq \mathbf{S}_i(n_i(p)). \tag{4.9}$$

Therefore, the p closest neighbors of the node i will consist of $\mathcal{N}_p(i) = (n_i(1), \ldots, n_i(p))$, where $\mathcal{N}_p(i)$ is defined as a vector rather than a set to keep the ordering.

4.4.2 Nearest Neighbors Convolution on Graphs

The notion of ordered closeness between the nodes provides a global property for all graphs and all nodes, only requiring a similarity function over the nodes. We take advantage of it to define a convolution that has shared weights, a meaningful interpretation and transferable filters.

Intuitively, the p Nearest Neighbors convolution takes an inner product of the weights $g \in \mathbb{R}^p$ with the values at the nodes in $\mathcal{N}_p(i)$, such that g_l corresponds to the l^{th} nearest neighbor $(n_i(l))$ for all nodes i.

Given a signal $\mathbf{f} = (f_1, \ldots, f_N)^T$ on the vertices of the graph $\mathcal{G}$, we first define a patch operator as

$$D_l(i)\mathbf{f} = f_{n_i(l)}, \quad l = 1, \ldots, p, \tag{4.10}$$

where p is the number of neighbors and hence the size of the extracted patch. We can now define the convolution between two signals $\mathbf{f}$ and $\mathbf{g}$ as

$$(\mathbf{f} \star \mathbf{g})(i) = \sum_{l=1}^{p} g_l D_l(i)\mathbf{f}. \tag{4.11}$$

4.4.3 Selection of Nearest Neighbors

The suggested convolution utilizes the similarity function to set an order between the nodes. But in some cases, this function is not immediately evident and needs to be defined explicitly. In this work we use a random walk to construct a diffusion similarity function over the graph nodes.

Let $\mathbf{P}$ denote the transition matrix of the random walk on the graph. $\mathbf{P}$ can be either directly given, or calculated by normalizing the rows of the adjacency matrix as given in Section 4.3. $\mathbf{P}$ can itself be directly used to get the similarity function, but when $\mathbf{P}$ is sparse, it might not incorporate the full structure of the graph. Therefore, to incorporate neighbors further away from every node we use higher orders of $\mathbf{P}$ given by $\mathbf{P}^k$, that correspond to nodes that are k hops away from the given node.

We define $\mathbf{Q}^{(r)} := \sum_{k=0}^{r} \mathbf{P}^k$, where $[\mathbf{P}^k]_{ij}$ is the probability of transitioning from i to j in k hops. That is,

$$\mathbf{Q}^{(0)} = \mathbf{I}, \quad \mathbf{Q}^{(1)} = \mathbf{I} + \mathbf{P}, \mathbf{Q}^{(2)} = \mathbf{I} + \mathbf{P} + \mathbf{P}^2, \cdots \tag{4.12}$$

Note that $Q_{ij}^{(r)}$ is also the expected number of visits from node i to node j in r steps. Hence $\mathbf{Q}$ provides a measure of closeness between the nodes of the graph by considering a random walk on it. As r increases we incorporate neighbors further away from the node, while the summation gives appropriate weights to the node and its closest neighbors. Figure 4.2 provides a visualization of the matrix $\mathbf{Q}$ over the 2-D grid.

4.4.4 Unknown Graph Structure

If the graph structure is unknown, it can be learned using several unsupervised or supervised graph learning algorithms. Learning the data graph structure is an active research topic and is not in the scope of this work.

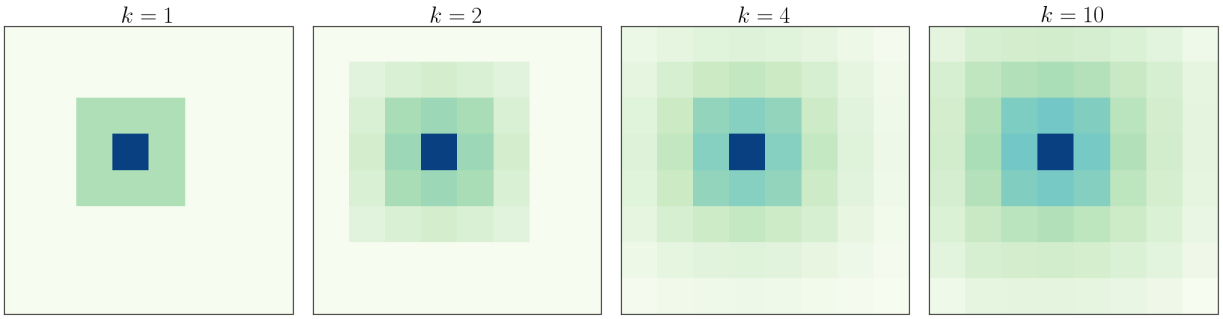


Figure 4.2: Visualization of a row of $Q^{(r)}$ on the graph generated over the 2-D grid at a node near the center. The graph was created by connecting each node to its 8 adjacent neighbors. For $r = 1$, most of the weight is on the node, with smaller weights on the first order neighbors. This corresponds to a standard 3×3 convolution. As r increases the number of active neighbors also increases, providing greater weight to neighbors farther away, while still keeping the local information.

The interested reader can start with Belkin and Niyogi [2001] and Henaff et al. [2015] discussing similarity matrix estimation.

4.4.5 Selection of the Power of Q

The selection of the value of r is data dependent, but there are two main components affecting its value. Firstly, it is necessary for r to be large enough to detect the top r neighbors of every node. If the transition matrix $\mathbf{P}$ is sparse, it might require higher values of r. Secondly, from properties of stochastic processes, we know that if we denote π as the Markov chain stationary distribution, then

$$\lim_{r \to \infty} \frac{Q_{ij}^{(r)}}{r} = \pi_j \quad \forall\ i, j. \tag{4.13}$$

This implies that for large values of r, local information will be smoothed out and the convolution will repeatedly be applied on the features with maximum connections. For this reason, we suggest keeping r relatively low (but high enough to capture sufficient features).

4.4.6 Implementation

The Convolution

An important feature of the suggested convolution is the complexity of the operation. For a graph with N nodes, a single p level convolution only requires $O(N \cdot p)$ flops and memory, where $p \ll N$.

For every graph convolution layer, we have as an input a $3D$ tensor of M observations with N nodes at depth d. We first extend the input with an additional dimension that includes the top p neighbors of each feature selected by $\mathbf{Q}^{(r)}$, transforming the input dimension from $3D$ to $4D$ tensor as

$$(M, N, d) \rightarrow (M, N, p, d)\,.$$

Now if we apply a graph convolution layer with d_{new} filters, the convolution weights will be a $3D$ tensor of size (p, d, d_{new}). Therefore application of a graph convolution which is a tensor dot product between the input and the weights along the (p, d) axes results in an output of size:

$$\Big((M, N)\,, (p, d)\Big) \bullet \Big((p, d)\,, (d_{new})\Big) = (M, N, d_{new})\,.$$

We have implemented the algorithm using Keras [Chollet, 2015]. The source code will be publicly available prior to the conference.

Computational Concerns

The major computational effort in this algorithm is the computation of Q, which is performed once per graph structure as a pre-processing step. As it is usually a one-time computation, it is not a significant constraint.

However, for very large graphs, if done naively, this might be challenging. An alternative can be achieved by recalling that Q is only needed in order to calculate the expected number of visits from a given node after k steps in a random walk. In most applications, when the graph is very large, it is also usually very sparse. This facilitates an efficient implementation of Breadth First Search algorithm (BFS). Hence, the selection of

the p neighbors can be parallelized and would only require $O(N \cdot p)$ memory for every unique graph structure, making the method scalable for very large graphs, when the number of different graphs is manageable.

Any problem that has many different large graphs is inherently computationally hard. The graph CNN reduces the memory required after the preprocessing from $O(N^2)$ to $O(N \cdot p)$ per graph. This is because the only information required from the graph is the p nearest neighbors of every node.

4.5 Experiments

In order to test the feasibility of the proposed CNN on graphs, we conducted experiments on well known data sets functioning as benchmarks: Merck molecular activity challenge, Cora citation graph and MNIST. These data sets are popular and well-studied challenges.

In our implementations, in order to enable better comparisons between the models and reduce the chance of over-fitting during the model selection process, we consider shallow and simple architectures instead of deep and complex ones. The hyper-parameters were chosen arbitrarily when possible rather than being tuned and optimized. Nevertheless, we still report state-of-the-art or competitive results on the data sets.

In this section, we denote a graph convolution layer with k feature maps by C_k and a fully connected layer with k hidden units by FC_k.

4.5.1 Merck molecular activity challenge

The Merck molecular activity is a Kaggle * challenge which is based on 15 molecular activity data sets. The target is predicting activity levels for different molecules based on the structure between the different atoms in the molecule. This helps in identifying molecules in medicines which hit the intended target and do not cause side effects.

Following Henaff et al. [2015], we apply our algorithm on the DPP4 dataset. DPP4 contains 6148 training and 2045 test molecules. Since we don't have an adjacency graph in this case, we use the absolute value of

*Challenge website is `https://www.kaggle.com/c/MerckActivity`

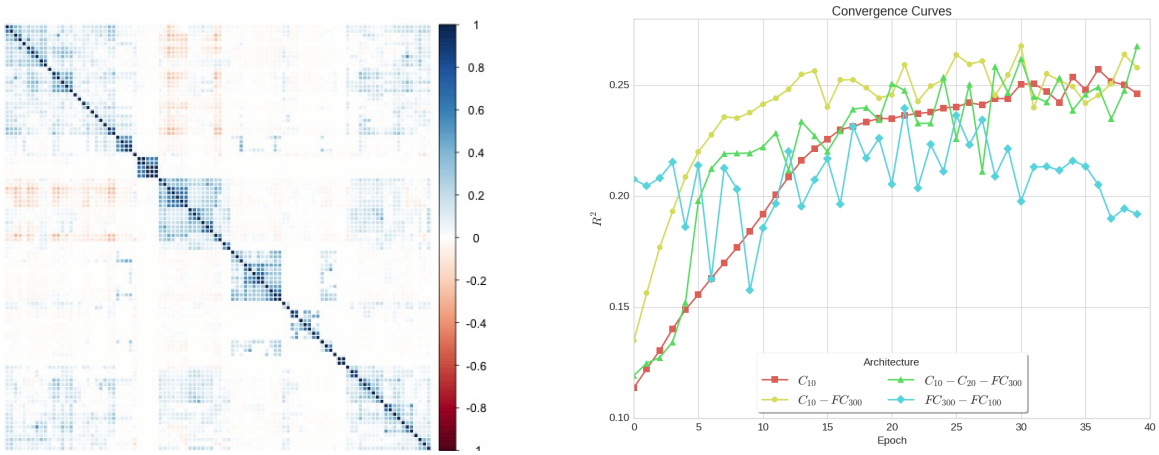


Figure 4.3: Left: Visualization of the correlation matrix between the first 100 molecular descriptors (features) in the DPP4 Merck molecular activity challenge training set. The proposed method utilizes the correlation structure between the features. **Right:** Convergence of R^2 for the different methods on the test set. The graph convolution converges more steadily as it uses fewer parameters.

the correlation matrix between the molecules as the adjacency matrix. That is if $\mathbf{R}$ is the correlation matrix, we consider $\mathbf{W} = |\mathbf{R}|$.

We notice that some of the features of the molecules are very sparse and are only active in a few molecules. For these features, the correlation estimation is not very accurate. Therefore, we use features that are active in at least 20 molecules (observations); resulting in 2153 features. As can be seen in Figure 4.3, there is significant correlation structure between different features. This implies strong connectivity among the features which is important for the application of the proposed method.

The training in the experiments was performed using Adam optimization procedure [Kingma and Ba, 2014] where the gradients are derived by the back-propagation algorithm, using the root mean-squared error loss (RMSE). We used learning rate $\alpha = 0.001$, fixed the number of epochs to 40 and implemented dropout regularization on every layer during the optimization procedure. The absolute values of the correlation matrix were used to learn the graph structure. We found that a small number of nearest neighbors (p) between 5 to 10 works the best, and used $p = 5$ in all models.

Following the standard set by the Kaggle challenge, results are reported in terms of the squared correlation (R^2), that is,

$$R^2 = \text{Corr}(Y, \hat{Y})^2,$$

where Y is the actual activity level and $\hat{Y}$ is the predicted one.

The convergence plot given in Figure 4.3 demonstrates convergence of the selected architectures. The contribution of the suggested convolution is explained in view of the alternatives:

- **Fully connected Neural Network:** Models first applying convolution followed by a fully connected hidden layer, converge better than more complex fully connected models. Furthermore, convergence in the former methods are more stable in comparison to the fully connected methods, due to the parameter reduction.

- **Linear Regression:** Optimizing over the set of convolutions is often considered as automation of the feature extraction process. From that perspective, a simple application of one layer of convolution, followed by linear regression, significantly outperforms the results of a standalone linear regression.

Table 4.1 provides more thorough R^2 results for the different architectures explored, and compares it to two of the winners of the Kaggle challenge, namely the Deep Neural Network and the random forest in Ma et al. [2015]. We perform better than both the winners of the Kaggle contest.

The models in Henaff et al. [2015] and Bruna et al. [2013] use a spectral approach and currently are the state-of-the-art. In comparison to them, we perform better than the Spectral Networks CNN on unsupervised graph structure, which is equivalent to what was done by using the correlation matrix as similarity matrix. The one using Spectral Networks on supervised graph structure holds the state-of-the-art by learning the graph structure. This is a direction we have not yet explored, as graph learning is beyond the scope of this work, although it will be straightforward to apply the proposed graph CNN in a similar way to any learned graph.

4.5.2 Cora Citation Graph

The Cora dataset is a popular benchmark for the problem of classification of graph vertices. The dataset itself is an undirected unweighted citation graph with 2708 vertices that represent scientific publications and 5429 edges that represent citations and the goal is to predict the class label for each publication, among

Method	Architecture	R^2
OLS Regression		0.135
Random Forest		0.232
Merck winner DNN		0.224
Smooth Spectral CNN	C_{64}-P_8-C_{64}-P_8-FC_{1000}	0.204
Smooth Spectral CNN (supervised graph)	C_{16}-P_4-C_{16}-P_4-FC_{1000}	0.277
Fully connected NN	FC_{300}-FC_{100}	0.195
NNCNN	C_{10}	0.246
NNCNN	C_{10}-FC_{100}	0.258
NNCNN	C_{10}-C_{20}-FC_{300}	0.264

Table 4.1: The squared correlation between the actual activity levels and predicted activity levels, R^2 for different methods on DPP4 data set from Merck molecular activity challenge.

Method	Cora
ManiReg Belkin et al. [2006]	59.5%
SemiEmb Weston et al. [2012]	59.0%
LP Zhu et al. [2003]	68.0%
DeepWalk Perozzi et al. [2014]	67.2%
Planetoid Yang et al. [2016]	75.7%
DCNN Atwood and Towsley [2016]	76.80%
GCN Kipf and Welling [2016]	81.59%
MoNet Monti et al. [2017]	81.69%
NNCNN	**81.00%**

Table 4.2: Vertex classification accuracy on the Cora dataset test set.

7 different classes. For each publication the features are a collection of 1433 binary vectors indicating the presence of a given word within a dictionary.

We use the experimental settings as presented in Kipf and Welling [2016], Monti et al. [2017] and Yang et al. [2016]. It can be seen in Table 4.2 that our results are comparable to current state of the art methods (as presented in Kipf and Welling [2016] and Monti et al. [2017]) addressing the same problem. We take the number of neighbors, $p = 10$ and the power of $\mathbf{Q}$, $r = 3$ with 15 filters and one graph convolution layer.

It is worth noting that in this example (in contrast to Section 4.5.1), the graph structure is over the observations (publications) themselves. The suggested spatial convolution effectively learns different weighted averages for words occurring in a given publication and its neighbors on the citation graph.

4.5.3 MNIST data

The MNIST data often functions as a benchmark data set to test new machine learning methods. We experimented with two different graph structures for the images. In the first experiment, we considered the images as observations from an undirected graph on the 2-D grid, where each pixel is connected to its 8 adjoining neighboring pixels. This experiment was done, to demonstrate how the graph convolution compares to standard CNN on data with grid structure.

We used the convolutions over the grid structure as presented in Figure 4.2 using $Q^{(3)}$ with $p = 25$ as the number of nearest neighbors. Due to the symmetry of the graph, in most regions of the image, multiple pixels are equidistant from the pixel being convolved. In order to solve this, if the ties were broken in a consistent manner, the convolution would be reduced to the regular convolution on a 5×5 window. The only exceptions to this would be the pixels close to the boundary. To make the example more compelling, we broke ties arbitrarily, making the training process harder compared to regular CNN. Imitating LeNet LeCun et al. [1998], we considered an architecture with C_{40}, $Pooling_{(2\times2)}$, C_{80}, $Pooling_{(2\times2)}$, FC_{100} followed by a linear classifier that resulted in a 0.87% error rate. This is comparable to a regular CNN with the same architecture that achieves an error rate of about 0.75%-0.8%. We outperform a fully connected neural network which achieves an error rate of around 1.4%, which is expected due to the differences in the complexities of the models.

In the second experiment, we used the correlation matrix to estimate the graph structure directly from the pixels. Since some of the MNIST pixels are constant (e.g the corners are always black), we restricted the data only to the active 717 pixels that are not constant. We used $Q^{(1)}$ with $p = 6$ as the number of neighbors. This was done in order to ensure that the spatial structure of the image no longer effected the results. With only 6 neighbors, and a partial subset of the pixels under consideration, the relative location of the top correlated pixels necessarily varies from pixel to pixel. As a result, regular CNNs are no longer applicable on the data whereas the convolution proposed in this work is. We compared the performance of our CNN to fully connected Neural Networks.

Method	Error (%)	# of Parameteres
Logistic Regression	7.49	$7,180$
C_{20}	1.94	$143,550$
$C_{20} - C_{20}$	1.59	$145,970$
$C_{20} - FC_{512}$	1.45	$7,347,862$
$FC_{512} - FC_{512}$	1.59	$635,402$

Table 4.3: Error rates of different methods on MNIST digit recognition task without the underlying grid structure.

During the training process, we used a dropout rate of 0.2 on all layers to prevent over-fitting. In all the architectures the final layer is a standard softmax logistic regression classifier.

Table 4.3 presents the experimental results. The NNCNN performs on par with the fully connected neural networks, with fewer parameters. A single layer of the nearest neighbors convolution followed by logistic regression greatly improves the performance of logistic regression, demonstrating the potential of the nearest neighbors convolution for feature extraction purposes. As with regular convolutions, $C_{20} - FC_{512}$ required over 7 million parameters as each convolution uses small amount of parameters to generate different maps of the input. This suggests that the convolution can be made even more effective with the development of an efficient spatial pooling method on graphs, which is a known but unsolved problem.

4.6 Conclusions

We propose a generalization of convolutional neural networks from grid-structured data to graph-structured data, a problem that is being actively researched by our community. Our novel contribution is a convolution over a graph that can handle different graph structures as its input. The proposed convolution contains many sought-after attributes; it has a natural and intuitive interpretation, it can be transferred within different domains of knowledge, it is computationally efficient and it is effective.

Furthermore, the convolution can be applied to any data with any structure given a relative ordering of closeness between the variables. For standard regression or classification problems, we can learn the graph structure from the data by using the correlation matrix or by using other methods that could give us a similarity function.

Compared to a fully connected layer, the suggested convolution has significantly fewer parameters while providing stable convergence and comparable performance. Our experimental results on the Merck Molecular Activity data set, Cora citation graph and MNIST data demonstrate the potential of this approach. In future research we hope to study the nearest neighbors convolution on other complex data structures equipped with a similarity function or a distance metric.

Convolutional Neural Networks have already revolutionized the fields of computer vision, speech recognition and language processing. We think an important step forward is to extend it to other problems which have an inherent graph structure.

Chapter 5

Confidence Intervals for Selected Parameters

Section based on Benjamini et al. [2019].

5.1 Introduction

Modern statistical applications, for instance, studies of high-throughput experiments or high-dimensional databases, rarely involve only one parameter. Rather, investigators generally rely on model selection or parameter selection of some kind, e.g., algorithmically, by trial and error, or merely by examining the data before deciding what analysis to perform. Practical or scientific reasons may limit interest or reporting to a subset of parameters, even when the analysis involved many more. These may be the only parameters reported in the analysis, or simply emphasized above the rest by inclusion in the abstract, discussion, tables, or figures. Inferences about such selected parameters often are based on the same data used to decide they are "important."

Because selection alters the distribution of p-values, estimators, and test statistics, selection complicates inference. The ASA board issued a warning about p-values for selected parameters [Wasserstein and Lazar, 2016], but did not suggest any remedy. Instead, they recommended using alternatives such as confidence intervals. Unfortunately, selection also alters the coverage probability of standard confidence intervals.

Here, we provide new methods for constructing simultaneous confidence intervals for parameters selected because their estimates are largest. The methods illustrate that using information about the specific way the parameters are selected can improve inferences.

The issue of inference after selecting hypotheses, parameters, or models for inference after observing the data—*selective inference*—was recognized 70 years ago in the problem of making all pairwise comparisons between independent groups. The field that copes with selective inference came to be known as 'multiple comparisons.' Inference about the difference selected as interesting because it was estimated to be largest led Tukey to introduce the studentized range [Tukey, 1953, Braun, 1994], and selection has remained a motivating theme, as exemplified by the introduction to the definitive work by Hochberg and Tamhane [1987]. That motivation notwithstanding, the book's solutions involved simultaneous coverage over *all* parameters of potential interest, thereby controlling the error probability for whatever subset was selected. Similarly, Berk et al. [2013], who develop methods for selective inference in the context of model selection in linear regression, rely on simultaneous protection.

Addressing selective inference by simultaneous coverage of all possible subsets of parameters is conservative in "modern" statistical problems with very many potential inferences, often more than observations ("$p \gg n$"). Benjamini and Yekutieli [2005] distinguished between *simultaneous inference* and *selective inference*. The latter requires that the inferential property (e.g., the Type I error rate or coverage probability) hold *on average over the selected parameters*. Selective inference is often less stringent than simultaneous inference. For instance, if a collection of intervals has simultaneous coverage probability $1 - \alpha$, its selective coverage is at least $1 - \alpha$, but the converse is not true. Methods for selective inference can be more powerful than methods for simultaneous inference, as illustrated by methods that control the False Coverage Rate (FCR) [Benjamini and Yekutieli, 2005].

FCR control is a property of the entire selected set of intervals. FCR is not controlled for subsets of *that* set. This is not a hypothetical problem. For instance, we might evaluate a large number of potential drug molecules for efficacy, then decide to look more closely at the most promising ten. One or two of the ten (not necessarily the largest) are then used in additional experiments. Regions on the genome may be identified to

be associated with some disease, but only one or two locations in each region used for replication. Identifying a few peaks of activity in fMRI studies is another example where initial screening is followed by study of a subset of items that pass the initial screening. In all these examples, controlling FCR in the initial screening guarantees nothing about subsets of the set that pass.

This problem is attracting attention, but proposed solutions require specifying how the sub-selection is performed. See, e.g., Katsevich and Ramdas [2018] and references therein. On the other hand, starting with simultaneous confidence intervals could be too conservative or exaggerate the potential of treatments by including larger effects than the data support.

In this work, we explore a middle way: Given a specific selection rule, a set of confidence intervals has *simultaneous coverage over the selected* if the probability of not covering any selected parameter is controlled at a desired level. Simultaneity is guaranteed only on the selected set, not all possible sets. This allows us to make sharper inferences than omnibus protection against all selection rules allows. For example, an interval for the parameter estimated to be the larger of two does not need to be longer than the standard univariate interval; see Section 5.3.

We demonstrate the advantage of this approach for a simple yet practical rule: "select the k parameters estimated to be largest." This rule is used—sometimes silently—in applications in which a single risk factor is of primary interest, but there is a collection of possible confounders. We also give a new result for the rule "select the parameter estimated to be largest in absolute value" in the bivariate normal case.

5.2 Simultaneous over selected parameters

We observe $Y = (Y_i)_{i=1}^m$, where $Y_i \sim F_{\theta_i}^i$, $i = 1, \ldots, m$, are real-valued random variables. A *selection rule* $S(\cdot)$ is a mapping from $\Re^m$ into $\mathcal{P}(\{1, \ldots, m\}) \equiv \mathcal{P}(m)$, the power set of $\{1, \ldots, m\}$, that is F_θ-measurable for all θ. The set $S(Y)$ consists of the indices of the *selected parameters*: If $Y = y$, we seek finite-length confidence intervals for $\{\theta_i\}_{i \in S(y)}$. We make no confidence statement about the other parameters.

The *Simultaneous over all Possible selections* (SoP) error rate is

$$\text{SoP} \equiv \max_{s \in \mathcal{P}(m)} \mathbb{E} \max_{i \in s} 1_{\theta_i \notin \mathcal{I}_i(Y)}, \tag{5.1}$$

The *False Coverage-statement Rate* (FCR) controls non-coverage on average over the selected parameters:

$$\text{FCR} \equiv \mathbb{E}_\theta \left[\frac{\sum_{i \in S(Y)} 1_{\theta_i \notin \mathcal{I}_i(Y)}}{|S(Y)|} \right], \tag{5.2}$$

where $\frac{0}{0} \equiv 0$ (if no interval is constructed, no interval fails to cover). Another error rate that explicitly involves S is the *Conditional over Selected* (CoS):

$$\text{CoS} \equiv \mathbb{E}_\theta \left[\max_{i \in S(Y)} 1_{\theta_i \notin \mathcal{I}_i(Y)} | i \in S(Y) \right]. \tag{5.3}$$

(For a recent review of this criterion, see Taylor and Tibshirani [2015].)

This paper focuses on the *Simultaneous over Selected parameters* (SoS) error rate:

$$\text{SoS} \equiv \mathbb{P} \left\{ \exists i \in S(Y) : \mathcal{I}_i(Y) \not\ni \theta_i \right\}. \tag{5.4}$$

SoS is the probability that any interval for a selected parameter fails to cover. Controlling SoS allows further selection from $S(y)$ without requiring the intervals to be modified.

Despite how often practitioners use the same data to select a set of parameters and then make inferences about the selected parameters, there are few results regarding controlling SoS. Venter [1988] constructed a confidence interval for the mean estimated to be the largest among m independent Gaussian estimators and Fuentes et al. [2018] recently constructed SoS intervals for the k means estimated to be largest among m independent normal estimators. "Multiple comparisons with sample best" [Hsu, 1981, 1996] also involves testing a set of hypotheses that depends on the data through the identity of the parameter estimated to be largest, to which all other parameter estimates are compared: this might be the first example of concern about SoS. But the general idea of simultaneous coverage over a selected subset does not seem to have been recognized.

Controlling SoP obviously controls both SoS and FCR. Controlling CoS assures conditional coverage for each selected parameter, so it controls coverage on average for the selected parameters. Therefore, CoS controls the conditional FCR, given that at least one parameter was selected. Since conditional FCR is larger than FCR, controlling CoS controls FCR.

For some selection rules, intervals that control CoS might not control SoP; for others, CoS yields longer intervals than SoP. Intervals with CoS $\leq 1 - \alpha/m$ control SoP at level $1 - \alpha$.

When $|S(Y)| = 1$ with probability 1, SoS and CoS are equivalent and CoS intervals control SoS and SoS intervals control CoS. If $|S(Y)| \geq 1$ with probability 1, $1 - \alpha/|S(Y)|$ CoS intervals control SoS at level $1 - \alpha$.

5.3 Controlling SoS by inverting non-equivariant unconditional tests

One way to construct intervals that control SoS is to make simultaneous intervals for all m parameters. If we construct the simultaneous confidence intervals by inverting suitable non-equivariant hypothesis tests, then projecting the joint confidence set onto the selected components, we can get "selective" behavior by designing the tests so that the confidence intervals for the parameters that are *not* selected are $(-\infty, \infty)$, in effect not making any inference about those parameters.

For instance, consider the acceptance region A_{μ,c_μ} for testing the hypothesis $\theta = \mu = (\mu_i)_{i=1}^m$:

$$A_{\mu,c_\mu} \equiv \{y \in \Re^m : |\mu_i - y_i| \leq c_\mu;\ i \in S(y)\}, \tag{5.5}$$

where

$$c_\mu \equiv \inf_{c \in \Re} \left\{c : \mathbb{P}_\mu\{Y \notin A_{\mu,c_\mu}\} \leq \alpha\right\}. \tag{5.6}$$

Inverting this family of tests for $Y = y$ yields

$$\mathcal{C}(y) = \{\mu \in \Re^m : y \in A_{\mu,c_\mu}\}, \tag{5.7}$$

which satisfies

$$\mathbb{P}_\theta\{\mathcal{C}(Y) \not\ni \theta\} \leq \alpha. \tag{5.8}$$

Whenever $\mathcal{C}(Y) \ni \theta$, θ_i is between $\inf_{\mu\in\mathcal{C}(y)} \mu_i$ and $\sup_{\mu\in\mathcal{C}(y)} \mu_i$, for all i. Therefore, the intervals

$$\mathcal{I}_i(y) \equiv \left[\inf_{\mu\in\mathcal{C}(y)} \mu_i, \sup_{\mu\in\mathcal{C}(y)} \mu_i\right], \quad i = 1, \ldots, m, \tag{5.9}$$

are simultaneous $1-\alpha$ confidence intervals for $\{\theta_i\}_{i=1}^m$. Because these intervals have simultaneous $1-\alpha$ coverage for *all* $\{\theta_i\}_{i=1}^m$, they have simultaneous coverage for $\{\theta_i\}_{i\in S(Y)}$.

This choice of A_{μ,c_μ} gives these confidence intervals a selective structure: Suppose $i \notin S(Y)$. Consider μ^γ with components

$$\mu_i^\gamma \equiv \begin{cases} Y_i, & i \in S(Y) \\ \gamma, & i \notin S(Y). \end{cases} \tag{5.10}$$

Then $A_{\mu^\gamma,c_{\mu^\gamma}} \ni Y$. Hence, $\mu^\gamma \in C(Y)$, and so $\gamma \in \mathcal{I}_i(Y)$ for all $i \notin S(Y)$. Since this holds for all $\gamma \in \Re$, $\mathcal{I}_i(Y) = (-\infty, \infty)$ for $i \notin S(Y)$: the procedure does not try to constrain non-selected parameters.

In general, c_μ depends on the selection rule $S(\cdot)$, but bounds on c_μ can make it easy to invert these families of tests conservatively. For instance, if $\forall\mu$, $c_\mu \leq c^+$, then $A_{\mu,c_\mu} \subset A_{\mu,c^+}$, and $\mathcal{I}_i(y) \subseteq [y_i - c^+, y_i + c^+]$ for $i \in S(Y)$. If one can find a simple function $f(\mu)$ such that $c_\mu \leq f(\mu)$ for all μ, then conservative confidence intervals can be found by inverting acceptance regions based on f rather than c.

5.3.1 The larger of two exchangeable, symmetric estimates

We construct SoS intervals for the parameter θ_i estimated to be the larger of two: $S(y) \equiv \{(1)\}$, where (1) denotes the index of the larger of Y_1 and Y_2. (Ties can be broken lexicographically.) We specialize to the case $\{Y_i - \theta_i\}_{i=1}^2$ are exchangeable and symmetric; they need not be independent or continuous.

Define the set transformation

$$A^{-T} \equiv \{(-y_2, -y_1) : (y_1, y_2) \in A\}. \tag{5.11}$$

Symmetry and exchangeability imply that if $A \subset \Re^2$ is measurable with respect to the joint distribution of $\{Y_i\}$, so is A^{-T}, and $\mathbb{P}_0\{A^{-T}\} = \mathbb{P}_0\{A\}$. By definition, $\mathbb{P}_\mu\{A\} = \mathbb{P}_0\{A - \mu\}$.

For this S, the acceptance region (5.5) is the union of two disjoint semi-infinite trapezoids:

$$A_{\mu,c} \equiv A_1 \cup A_2 \tag{5.12}$$

where

$$A_1 \equiv \{y \in \Re^2 : |y_1 - \mu_1| \le c \text{ and } y_1 \ge y_2\}$$

$$A_2 \equiv \{y \in \Re^2 : |y_2 - \mu_2| \le c \text{ and } y_2 > y_1\}.$$

Suppose $(Y_i - \theta_i)$, $i = 1, 2$, are exchangeable with a symmetric marginal distribution, and define $c_\alpha \equiv -\sup\{c : \mathbb{P}\{Y_i - \theta_i \le c\} \le \alpha\}$. For all $\mu \in \Re^2$,

$$\mathbb{P}_\mu\{Y \in A_{\mu,c_{\alpha/2}}\} \ge 1 - \alpha.$$

Proof. The proof hinges on the fact that

$$A_1 \cup \left\{(A_2 - \mu)^{-T} + \mu\right\} = \{y \in \Re^2 : |y_1 - \mu_1| \le c\}. \tag{5.13}$$

The LHS is a subset of the RHS: If $y \in A_1$, $|y_1 - \mu_1| \le c$. If $y \in (A_2 - \mu)^{-T} + \mu$, there exists x with $x_2 > x_1$ and $|x_2 - \mu_2| \le c$ for which $y = (x - \mu)^{-T} + \mu = (-x_2 + \mu_2 + \mu_1, -x_1 + \mu_1 + \mu_2)$. Thus $|y_1 - \mu_1| = |-x_2 + \mu_2| \le c$, and $y_1 - y_2 = x_1 - x_2 > 0$, so $y \in A_1$.

The RHS is a subset of the LHS: If $|y_1 - \mu_1| \le c$ and $y_1 \ge y_2$, $y \in A_1$. If $|y_1 - \mu_1| \le c$ and $y_1 < y_2$, $x = (y - \mu)^{-T} + \mu = (-y_2 + \mu_2 + \mu_1, -y_1 + \mu_1 + \mu_2)$. Then $|x_2 - \mu_2| = |-y_1 + \mu_1| \le c$, and $x_2 - x_1 = -y_1 + y_2 > 0$, so $x \in A_2$. Calculation verifies $(x - \mu)^{-T} + \mu = y$. Thus (5.13) holds.

Now

$$\begin{aligned}
\mathbb{P}_\mu A_{\mu,c} &= \mathbb{P}_0\{A_{\mu,c} - \mu\} \\
&= \mathbb{P}_0\{(A_1 - \mu) \cup (A_2 - \mu)\} \\
&= \mathbb{P}_0\{A_1 - \mu\} + \mathbb{P}_0\{A_2 - \mu\}
\end{aligned}$$

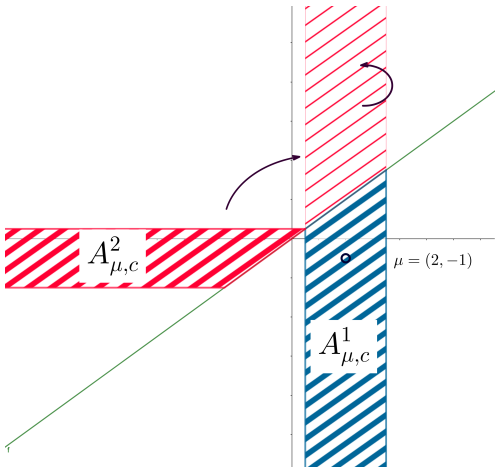


Figure 5.1: The semi-infinite trapezoids that comprise the acceptance region $A_{\mu,c}$. The blue hashed region is $A^1_{\mu_1,c}$ and the dark red hashed region is $A^2_{\mu_2,c}$. Exchangeability and symmetry of $\{Y_i - \theta_i\}$ imply that the probability of the lighter red region is equal to that of the dark red region if $\theta = \mu$, for every $\mu \in \Re^2$.

$$
\begin{aligned}
&= \mathbb{P}_0\{A_1 - \mu\} + \mathbb{P}_0\{(A_2 - \mu)^{-T}\} \\
&= \mathbb{P}_0\{(A_1 - \mu) \cup (A_2 - \mu)^{-T}\} \\
&= \mathbb{P}_\mu\left\{A_1 \cup \left\{(A_2 - \mu)^{-T} + \mu\right\}\right\} \\
&= \mathbb{P}_\mu\{Y \in \{y : |y_1 - \mu_1| \le c\}\} \\
&= \mathbb{P}_0\{|Y_1| \le c\}.
\end{aligned}
$$

The conclusion follows by substituting $c = c_{\alpha/2}$. □

Figure 5.1 (a) illustrates the key idea in the proof.

The proposition shows that $A_{\mu,c_{\alpha/2}}$ is the acceptance region for a level α test of the hypothesis $\theta = \mu$. As mentioned above, inverting tests of this form yields the confidence interval

$$\mathcal{I}(y) \equiv [y_{(1)} - c_{\alpha/2}, y_{(1)} + c_{\alpha/2}]. \tag{5.14}$$

This is the standard two-sided univariate symmetric confidence interval, based on the larger of the two observations. *The standard two-sided univariate confidence interval has the right coverage probability despite the selection and the dependence.*

5.3.2 Larger absolute value of two normal estimators

This section addresses constructing a confidence interval for the component selected by the rule $S(y) = \arg\max_i |y_i|$, where $Y \sim N(\theta, I)$ with $\theta \in \Re^2$. (Since the two estimators have normal distributions, the probability of a tie is zero.) As above, we find the intervals by inverting a family of hypothesis tests. The acceptance regions for the tests are again a union of two pieces, but they are more complicated than those in section 5.3.1.

Acceptance regions

The acceptance region $B_{\mu,c}$ for testing the hypothesis $\theta = \mu$ is

$$B_{\mu,c_\mu} \equiv B^1_{\mu_1,c_\mu} \cup B^2_{\mu_2,c_\mu}, \tag{5.15}$$

where

$$B^i_{\mu_i,c_\mu} \equiv \left\{ y \in \Re^2 : |\mu_i - y_i| \le c_\mu \text{ and } |y_i| > |y_j|, j \ne i \right\}, \tag{5.16}$$

and c_μ is chosen so that the test has level α. Figure 5.2 plots acceptance regions for six values of μ.

Finding c_μ

The constant c_μ is the smallest c for which $\mathbb{P}_\mu \{Y \in B_{\mu,c}\} \ge 1 - \alpha$. At $\mu = 0$, B_{0,c_0} is a square; so, c_0 is Šidák's constant $c_{\text{Šid}}$. We will bound c_μ for $\mu \ne 0$.

Let $a \equiv \max(|\mu_1|, |\mu_2|)$. For for all $c > 0$,

$$\mathbb{P}_\mu \{Y \in B_{\mu,c}\} \ge \mathbb{P}_{(a,0)} \left\{ Y \in B_{(a,0),c} \right\}. \tag{5.17}$$

The proof is in appendix 5.4. Define

$$c^+_\mu \equiv c_{(|\mu_1| \vee |\mu_2|,0)} = c_{(0,|\mu_1| \vee |\mu_2|)}. \tag{5.18}$$

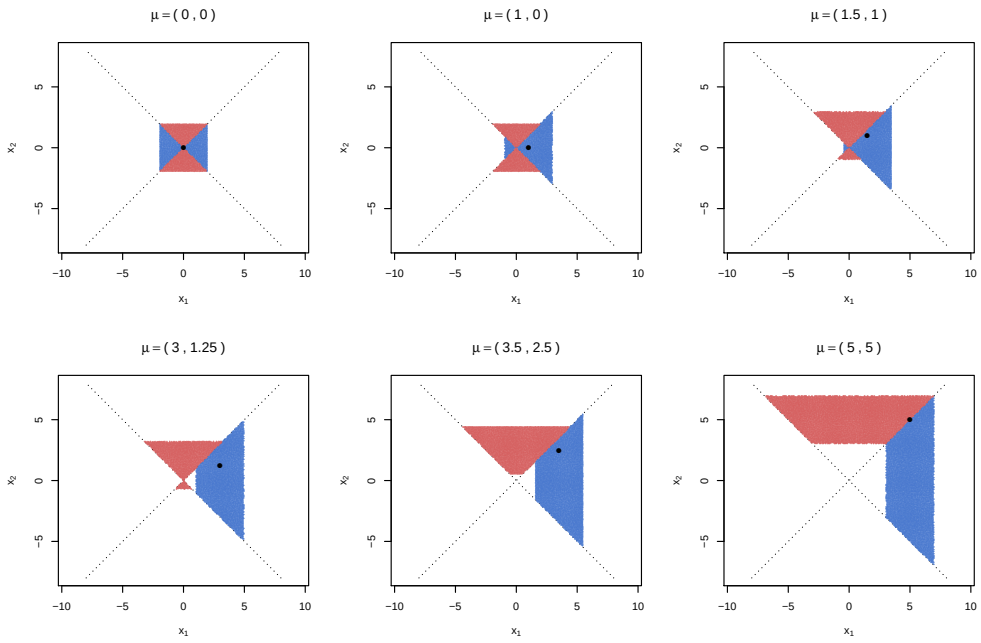


Figure 5.2: $B_{\mu,c}$ for various values of μ (the black dot in the figures). The blue area is $B^1_{\mu_1,c}$ and the cayenne area is $B^2_{\mu_2,c}$.

By Proposition 5.3.2, $c^+_\mu \geq c_\mu$. Explicit calculation gives

$$\mathbb{P}_\mu \left\{ Y \in B_{\mu,c^+_\mu} \right\} = \sum_{i \in \{1,2\}} \int_{-c}^{c} \phi(t) \left[\Phi(t + \mu_i - \mu_{i'}) - \Phi(-t - \mu_i - \mu_{i'}) \right] (-1)^{1_{t+\mu_i<0}} dt, \tag{5.19}$$

where $i' = 3 - i$. The value of c^+_μ is the smallest c for which (5.19) is at least $1 - \alpha$.

Hence, tests using c^+_μ instead of c_μ have level no larger than α, and inverting them will give confidence intervals with coverage probability at least $1 - \alpha$. Figure 5.3 plots c^+_μ as a function of $\max(|\mu_1|, |\mu_2|)$.

As $\max(|\mu_1|, |\mu_2|)$ grows, c^+_μ decreases monotonically; $c_{(3,0)} \approx z_{1-\frac{\alpha}{2}}$, the half-width of a standard univariate normal confidence interval.

The confidence interval

For $|y_1| > |y_2|$, the confidence interval for θ_1 is:

$$\mathcal{I}_1(y) = [\mu_-, \mu_+],$$

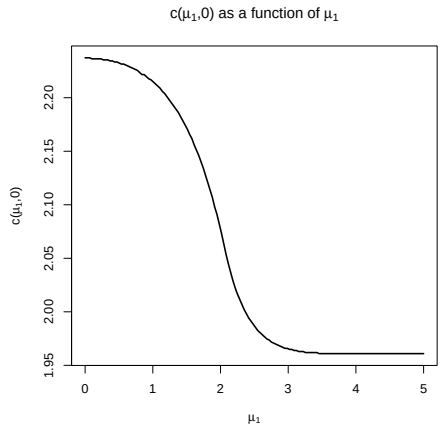


Figure 5.3: The upper bound c_μ^+ as a function of $\max(|\mu_1|, |\mu_2|)$.

where

$$\mu_- = \inf\{a \in \Re : a + c_{(a,0)} \geq y_1\}$$

$$\mu_+ = \sup\{a \in \Re : a - c_{(a,0)} \leq y_1\}.$$

Since $|c_{(\mu_1,0)}| \leq c_{(0,0)}$, both endpoints are between $y_1 - c_{(0,0)}$ and $y_1 + c_{(0,0)}$: the interval is shorter than Šidák's simultaneous interval. For $\alpha = 0.05$, the interval is widest when $y \approx (\pm 2.23, 0)$; there, the acceptance region for $\mu = 0$ just includes y. The maximum width is about 93.6% of the width of the Šidák intervals. As $|y_1| \to \infty$, the length converges to that of the standard unadjusted confidence interval, about 88% of the length of the Šidák interval.

Unlike the SoS interval for the larger of two, this SoS interval for the larger absolute value of two does not automatically work when $\{Y_i\}$ are dependent, because $\mathbb{P}_\mu\{Y \in B_{\mu,c}\}$ can be lower than it is when they are independent. Acceptance regions for parameters with dependent estimators could be calibrated by brute force computation, then inverted computationally to construct confidence intervals.

5.4 Proof of Proposition 5.3.2

Since $Y_i - \theta_i$ are IID standard normals, they are rotationally invariant and symmetric. Of course, $\mathbb{P}_\mu\{A\} = \mathbb{P}_0\{A - \mu\}$. These three properties let us find or bound $\mathbb{P}\{B_{\mu,c}\}$.

Assume without loss of generality that $mu_1 \geq mu_2 \geq 0$. The acceptance regions $B_{\mu,c}$ have three forms, depending on whether (1) $\mu_1 \geq \mu_2 \geq c$, (2) $\mu_1 \geq c \geq \mu_2$, or (3) $c \geq \mu_1 \geq \mu_2$.

Case (1) $\mu_1 \geq \mu_2 \geq c$. Because $\mu_1 \geq c$, $B^1_{(\mu_1,0),c}$ is a trapezoid and $B^2_{(\mu_1,0),c}$ consists of two congruent triangles. Rotating the triangles and reflecting them about $x = \mu_1$ yields the hexagon in Figure 5.4 and 5.5. Because $\mu_2 \geq c$, $B_{(\mu_1,\mu_2),c}$ consists of two trapezoids. Clockwise rotation and reflection of the trapezoid $B^2_{(\mu_1,\mu_2),c}$ yields a parallelogram. See Figure 5.6 and 5.7. The transformation $\varphi(x, y) = (x, y - \mu_2)$ translates (μ_1, μ_2) to $(\mu_1, 0)$, allowing us to compare their acceptance regions. See Figure 5.8 and 5.9.

The line $\overline{FB}$ in Figure 5.9 is $y = -x+2\mu_1+\mu_2$. It intersects $y = x$ at the point $C = (\mu_1 + \mu_2/2, \mu_1 + \mu_2/2)$. If $\mu_2/2 \geq c$, then $B_{(\mu_1,0),c} \subseteq B_{(\mu_1,\mu_2),c}$. Otherwise, $D = (\mu_1 + \mu_2 - c, \mu_1 + c)$ is above the point $E = (\mu_1 + \mu_2 - c, \mu_1 + \mu_2 - c)$, and to the right of $G = (\mu_1, \mu_1)$. Thus the triangles $\triangle CDE$ and $\triangle ABC$ are congruent. Since the reflection of $\triangle CDE$ about the horizontal line $x = \mu_1 + \mu_2/2$ is $\triangle ABC$, $\mathbb{P}\{\triangle CDE\} \geq \mathbb{P}\{\triangle ABC\}$, and so $\mathbb{P}_{(\mu_1,\mu_2)}\{A_{(\mu_1,\mu_2),c}\} \geq \mathbb{P}_{(\mu_1,0)}\{A_{(\mu_1,0),c}\}$.

Case (2) $\mu_1 \geq c \geq \mu_2$. The region $B_{(\mu_1,0),c}$ is as in case (1), but for $\mu_2 > 0$, $B^1_{(\mu_1,\mu_2),c}$ is a trapezoid and $B^2_{(\mu_1,\mu_2),c}$ consists of two triangles that meet at 0. Rotating the upper triangle $\pi/2$ clockwise, and the lower triangle $\frac{\pi}{2}$ counter clockwise, then reflecting both across $x = \mu_1$ and centering yields Figure **??**. As in case **(1)**, rotating the quadrilateral $DEFG$ about the line $x = \mu_1 + \mu_2/2$, and noticing that once again the top half is congruent to the bottom half (rotated by π) establishes the conclusion.

Case (3) $c \geq \mu_1 \geq \mu_2$. The only transformation required to compare the probabilities of the regions is to translate $B_{(\mu_1,\mu_2),c}$ to center it at $(\mu_1, 0)$. See Figure 5.12 and 5.13. The area of the quadrilateral $EFGH$ in

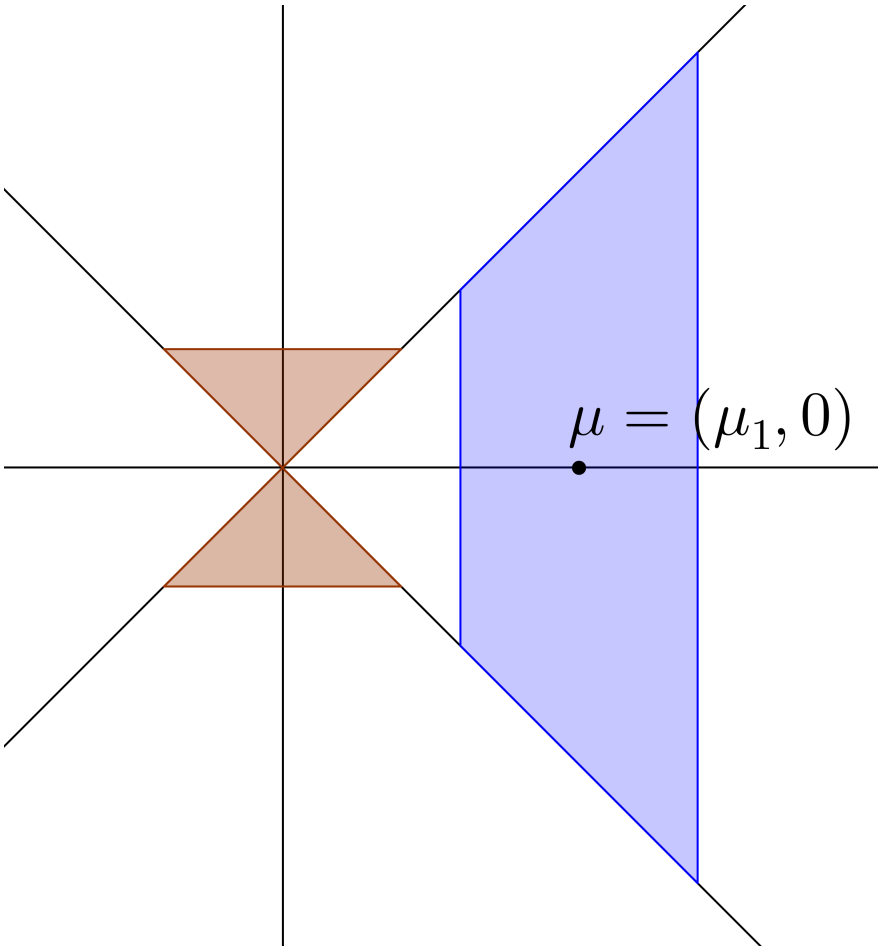


Figure 5.4: Acceptance region for $B_{(\mu_1,0),c}$.

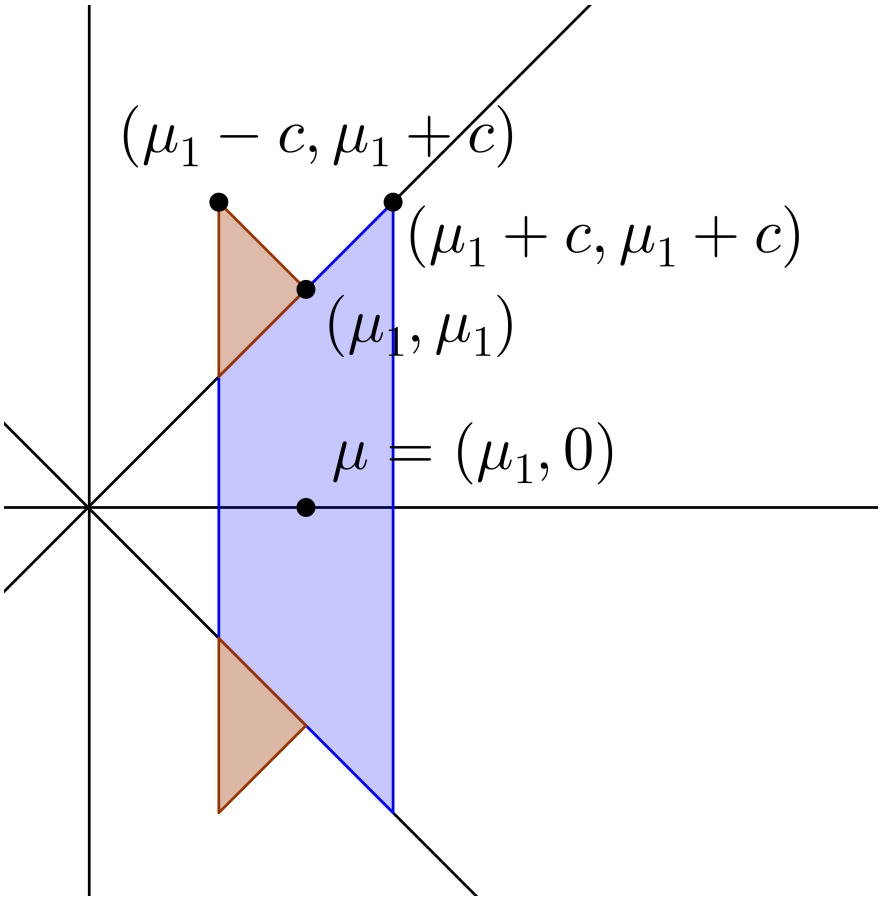


Figure 5.5: $B_{(\mu_1,0),c}$ after a $\mathbb{P}_\mu$-preserving transformation.

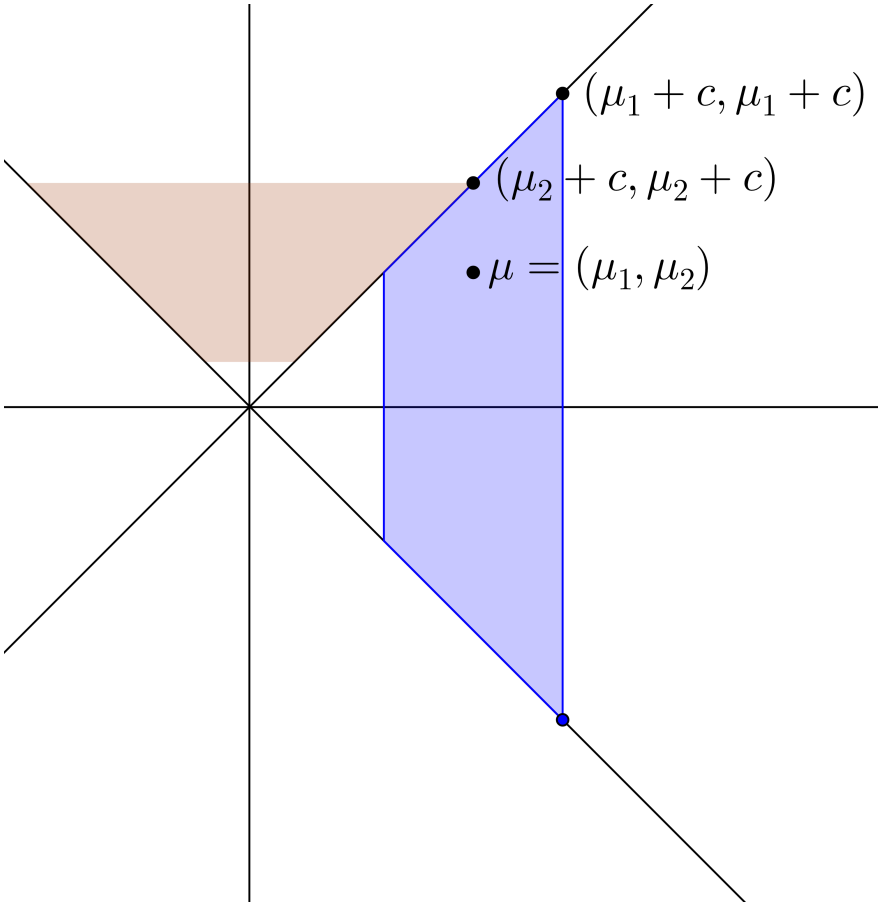


Figure 5.6: $B_{(\mu_1,\mu_2),c}$ for $\mu_1 > \mu_2 > c$.

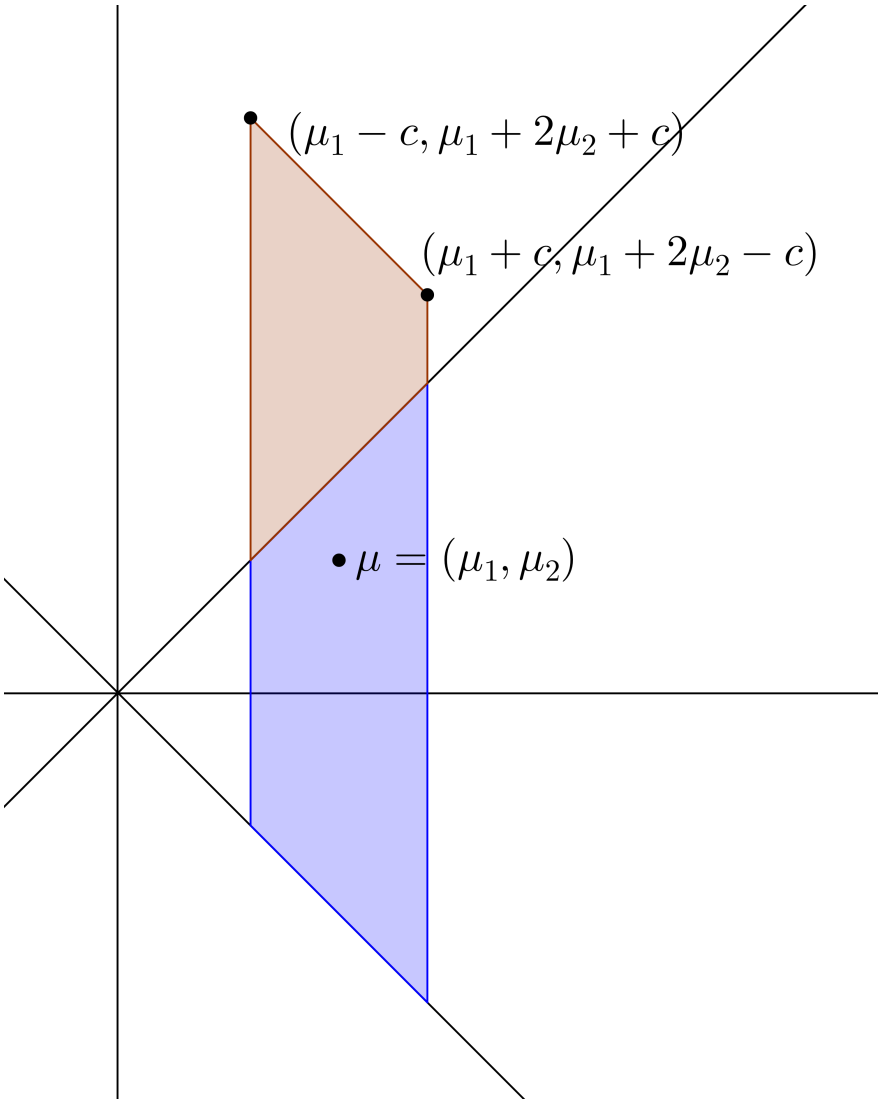


Figure 5.7: $B_{(\mu_1,\mu_2),c}$ after a $\mathbb{P}_\mu$-preserving transformation.

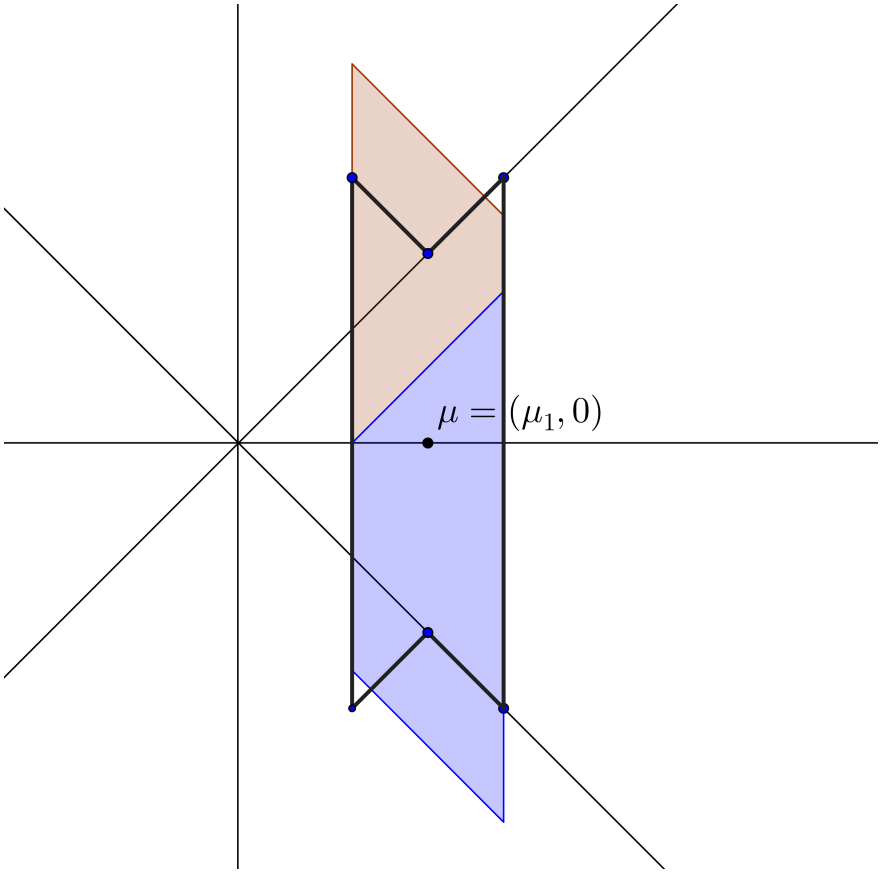


Figure 5.8: $B_{(\mu_1,\mu_2),c}$ and $B_{(\mu_1,0),c}$ superposed after $\mathbb{P}_\mu$-preserving transformations when $\mu_1 \geq \mu_2 \geq c$. The black hexagon is $B_{(\mu_1,0),c}$ and the shaded parallelogram is $B_{(\mu_1,\mu_2),c}$ after re-centered at $(\mu_1, 0)$.

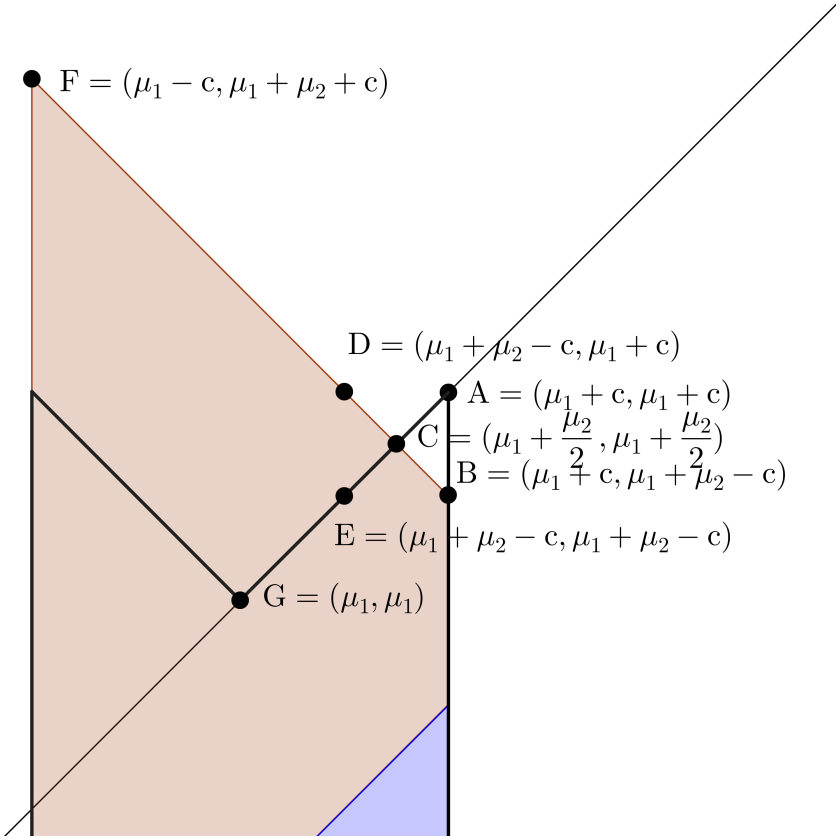


Figure 5.9: Detail of the top of panel 5.8 . The bottom half of the trapezoid is the top half rotated by π, which does change its probability.

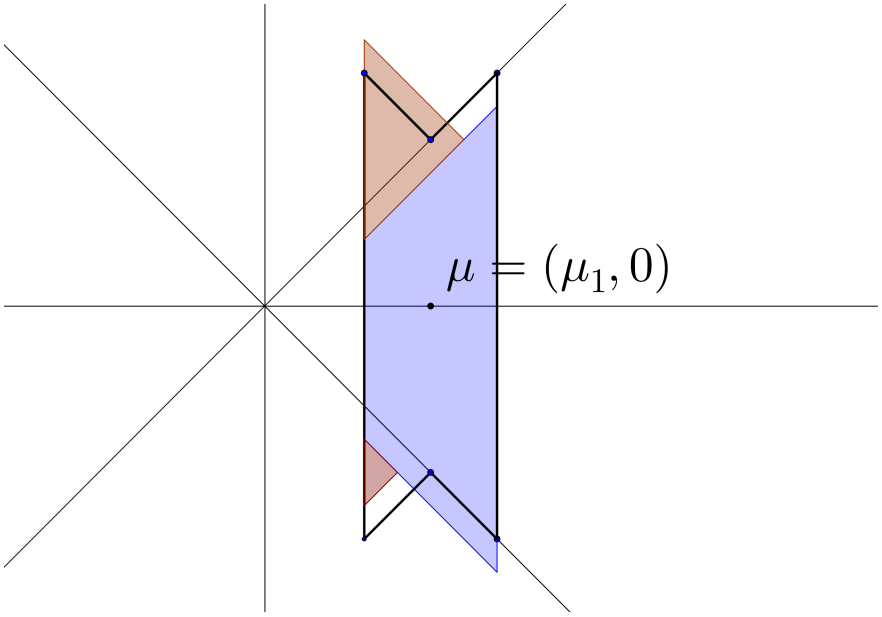


Figure 5.10: $B_{(\mu_1,\mu_2),c}$ for $\mu_1 > c > \mu_2$ superposed with $B_{(\mu_1,0),c}$, after a $\mathbb{P}_\mu$-preserving transformations.

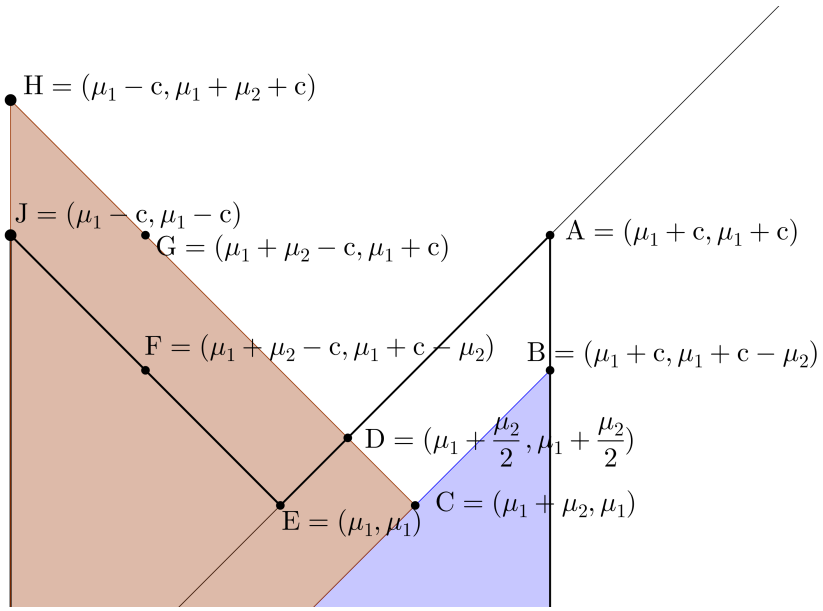


Figure 5.11: Detail of the top of panel 5.10

Figure 5.13 is greater than the area of the quadrilateral $ABCD$, and $EFGH$ is closer in norm to $(\mu_1, 0)$, so $\mathbb{P}_{(\mu_1,\mu_2)}\{A_{(\mu_1,\mu_2),c}\} \geq \mathbb{P}_{(\mu_1,0)}\{A_{(\mu_1,0),c}\}$.

5.5 Largest k of m

Rather than construct a confidence interval for the single parameter estimated to be the larger of 2, as in section 5.3.1, in this section we construct SoS-controlling confidence intervals corresponding for the k parameters estimated to be the largest of m parameters.

Consider m independent random variables $Y = (Y_1, \ldots, Y_m)$. Let $F^i_{\theta_i}$ be the CDF of Y_i, where $F^i_\zeta(y) = F^i_0(y - \zeta)$. Let $Y_{(1)} \geq Y_{(2)} \geq \cdots \geq Y_{(m)}$ be the order statistics of $\{Y_1, \ldots, Y_m\}$. Let $y = (y_1, \ldots, y_m) \in \Re^m$ and let $y_{(1)} \geq y_{(2)} \geq \cdots \geq y_{(m)}$ be the observed order statistics.

Consider the selection rule $S_k(\cdot)$ that keeps the components corresponding to the largest k components of Y, with ties broken lexicographically, and let $S^c_k(y) \equiv \{1, \ldots, m\} \setminus S_k(y)$. Let $|s|$ denote the cardinality of the set s, so $|S_k(y)| = k$. Let $\mathcal{S}_k \equiv \{s \subset \{1, \ldots, m\} : |s| = k\}$.

Since S contains the k components for which Y_i is largest, one might expect the conditional distribution Y_i given $i \in S$ to be stochastically larger than its unconditional distribution. It is, which allows us control

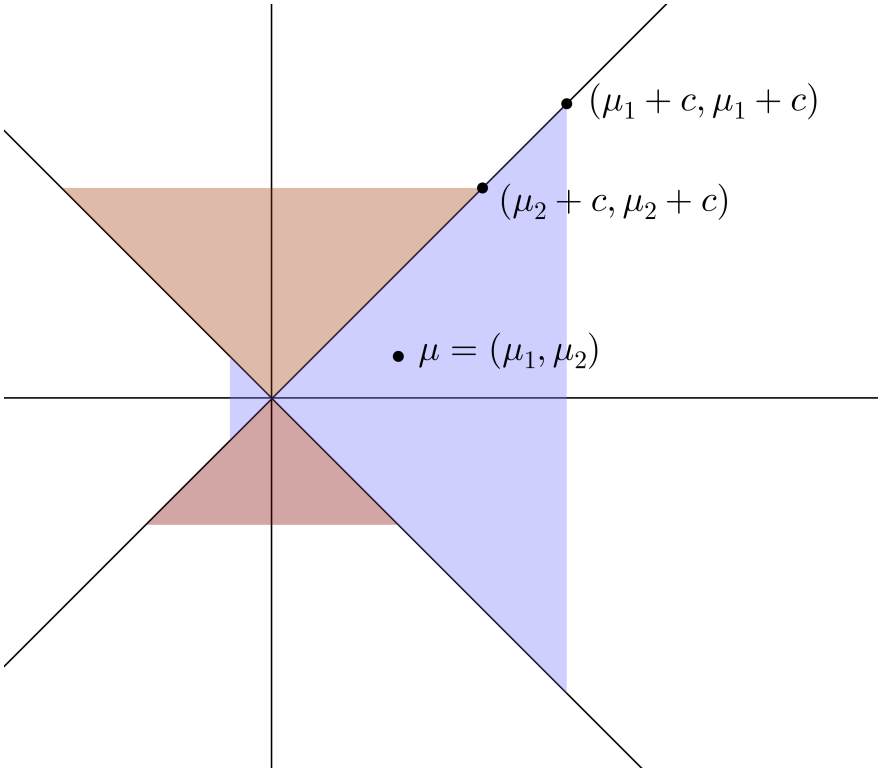


Figure 5.12: $B_{(\mu_1,\mu_2),c}$ and $B_{(\mu_1,0),c}$ when $c \geq \mu_1 \geq \mu_2$. $B_{(\mu_1,\mu_2),c}$.

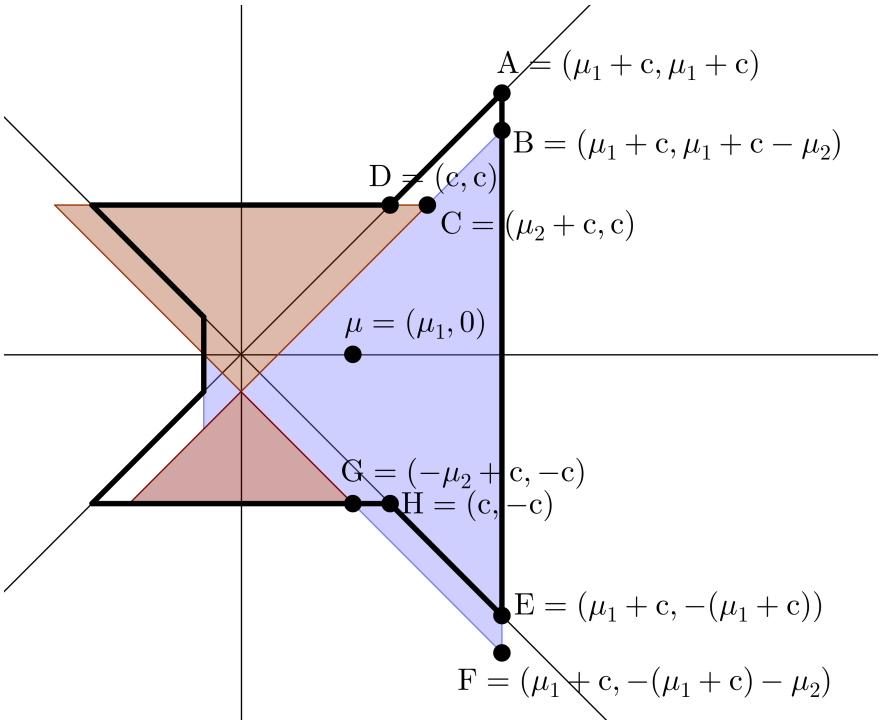


Figure 5.13: $B_{(\mu_1,\mu_2),c}$ and $B_{(\mu_1,0),c}$ when $c \geq \mu_1 \geq \mu_2$. $B_{(\mu_1,0),c}$ (the black polygon) and $B_{(\mu_1,\mu_2),c}$ (the shaded polygon) after the translation $h(x) = x - \mu_2$.

the chance that the upper endpoint of any interval is below its parameter using a Bonferroni adjustment with multiplicity k rather than m.

Consider the k intervals

$$\mathcal{I}_{(i)}(y) \equiv [y_{(i)} - \underline{c}_i, y_{(i)} + \bar{c}_i], \;\; 1 \le i \le k, \tag{5.20}$$

where $\underline{c}_i \equiv (F_0^i)^{-1}(1-\underline{\lambda})$ and $\bar{c}_i \equiv -(F_0^i)^{-1}(\bar{\lambda})$, $i = 1, \ldots, m$, with $\underline{\lambda} > 0$, $\bar{\lambda} > 0$, and $\underline{\lambda} + \bar{\lambda} < 1$

$$\mathbb{P}\left\{\exists i \in \{1, \ldots, m\} : \mathcal{I}_{(i)} \not\ni \theta_{(i)}\right\} \le m\underline{\lambda} + k\bar{\lambda}. \tag{5.21}$$

Corollary 5.0.1. *For all $\delta \in (0,1)$, if $\underline{\lambda} = \delta\alpha/m$ and $\bar{\lambda} = (1-\delta)\alpha/k$, then the intervals defined in equation 5.20 have $1-\alpha$ SoS coverage.*

Corollary 5.0.1 defines a family of intervals that control SoS at level α. The intervals are in general asymmetric, and the chance that the intervals miss the parameter from below is not in general equal to the chance that they miss from above. For $\delta = \frac{1}{2}$, neither the expected rate at which the upper endpoint is below its parameter nor the expected rate at which the lower endpoint is above its parameter exceeds $\alpha/2$. When $\{F_0^i\}$ are all symmetric, setting $\delta = m/(m+k)$ gives symmetric intervals corresponding to the two sided $m+k$ Bonferroni correction.

When $\{F_0^i\}$ are all equal, finding the δ that yields the shortest intervals is a 1-dimensional optimization problem. Figure 5.14 plots the length of the intervals for different values of k and m when $Y_i \sim N(\theta_i, 1)$. Because S selects large components, the shortest intervals extend below Y_i by more than they extend above Y_i.

Proof of Proposition 5.5. Define

$$\underline{V}_i = \begin{cases} 1, & y_i - \underline{c}_i > \theta_i \\ 0, & \text{otherwise} \end{cases}; \;\; \bar{V}_i = \begin{cases} 1, & y_i + \bar{c}_i < \theta_i \\ 0, & \text{otherwise} \end{cases} \tag{5.22}$$

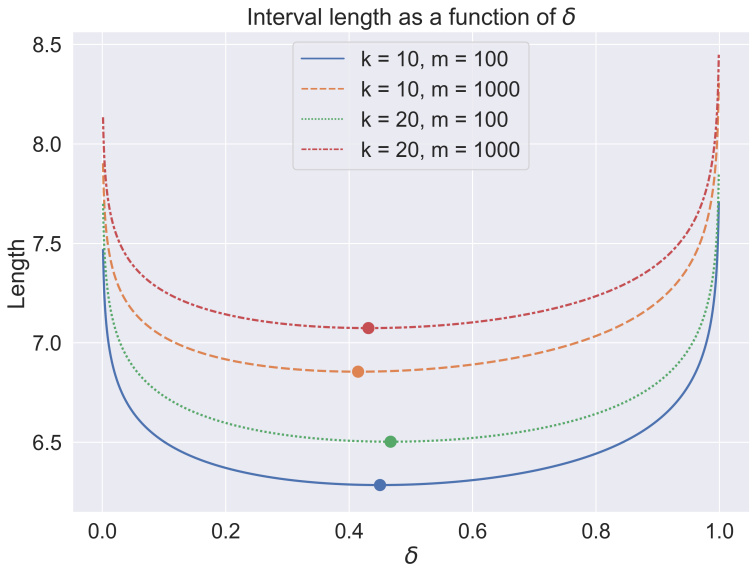


Figure 5.14: Lengths of the confidence intervals as a function of k, m, and δ, for $\alpha = 0.05$. The dots are plotted at the minimizing values of δ; lengths are nearly minimal when $\delta \approx 0.45$.

and

$$\underline{V}_{(i)} = \begin{cases} 1, & y_{(i)} - \underline{c}_i > \theta_{(i)} \\ 0, & \text{otherwise} \end{cases}; \quad \bar{V}_{(i)} = \begin{cases} 1, & y_{(i)} + \bar{c}_i < \theta_{(i)} \\ 0, & \text{otherwise.} \end{cases} \tag{5.23}$$

Now $\mathcal{I}_{(i)} \not\ni \theta_{(i)}$ if either $Y_{(i)} - \underline{c}_i > \theta_{(i)}$ (then the lower endpoint of the interval is greater than $\theta_{(i)}$, and $\underline{V}_{(i)} = 1$) or $Y_{(i)} + \bar{c}_i < \theta_{(i)}$ (then the upper endpoint of the interval is below $\theta_{(i)}$, and $\bar{V}_{(i)} = 1$).

Since $\underline{V}_{(i)} = 1$ and $\bar{V}_{(i)} = 1$ are mutually exclusive, the event $\mathcal{I}_{(i)} \not\ni \theta_{(i)}$ is the event $\underline{V}_{(i)} + \bar{V}_{(i)} = 1$. Let $\underline{V} \equiv \sum_{i=1}^{k} \underline{V}_{(i)}$ and $\bar{V} \equiv \sum_{i=1}^{k} \bar{V}_{(i)}$. The event that at least one interval does not cover its parameter is the event $\underline{V} + \bar{V} \geq 1$. Hence,

$$\mathbb{P}\{\exists i : \mathcal{I}_{(i)} \not\ni \theta_{(i)}\} = \mathbb{P}\left\{\underline{V} + \bar{V} \geq 1\right\} \leq \mathbb{P}\left\{\underline{V} \geq 1\right\} + \mathbb{P}\left\{\bar{V} \geq 1\right\}. \tag{5.24}$$

The first term on the right hand side is

$$\begin{aligned} \mathbb{P}\left\{\underline{V} \geq 1\right\} &\leq \sum_{i=1}^{k} \mathbb{P}\left\{\underline{V}_{(i)} = 1\right\} \leq \sum_{i=1}^{m} \mathbb{P}\left\{\underline{V}_{(i)} = 1\right\} \\ &= \sum_{i=1}^{m} \mathbb{P}\left\{\underline{V}_i = 1\right\} = \sum_{i=1}^{m} \mathbb{P}\left\{Y_i - \theta_i > \underline{c}_i\right\} \\ &= m\underline{\lambda}. \end{aligned} \tag{5.25}$$

The second term on the right of (5.24) is

$$\begin{aligned}
\mathbb{P}\left\{\bar{V} \geq 1\right\} &= \mathbb{P}\left\{\exists i \in S(Y) : \bar{V}_i = 1\right\} \\
&= \sum_{s \in \mathcal{S}_k} \mathbb{P}\left\{S(Y) = s \cap \exists i \in s : \bar{V}_i = 1\right\} \\
&\leq \sum_{s \in \mathcal{S}_k} \sum_{i \in s} \mathbb{P}\left\{S(Y) = s \cap \bar{V}_i = 1\right\} \\
&= \sum_{s \in \mathcal{S}_k} \sum_{i \in s} \mathbb{P}\left\{S(Y) = s \;\mid\; \bar{V}_i = 1\right\} \mathbb{P}\{\bar{V}_i = 1\} \\
&= \sum_{s \in \mathcal{S}_k} \sum_{i \in s} \mathbb{P}\left\{S(Y) = s \;\mid\; \bar{V}_i = 1\right\} \bar{\lambda} \\
&= \bar{\lambda} \sum_{s \in \mathcal{S}_k} \sum_{i \in s} \mathbb{P}\left\{\min_{j \in s} Y_j \geq \max_{k \in s^c} Y_k \;\mid\; Y_i < \theta_i - \bar{c}_i\right\}
\end{aligned}$$

Let $\tilde{F}_{\theta_i}(y) \equiv \mathbb{P}\{Y_i \leq y \mid Y_i < \theta_i - \bar{c}_i\}$. Then for all $y \in \Re$,

$$\begin{aligned}
\tilde{F}_{\theta_i}(y) &= \frac{\mathbb{P}\{Y_i \leq y \cap Y_i < \theta_i - \bar{c}_i\}}{\mathbb{P}\{Y_i < \theta_i - \bar{c}_i\}} \\
&= \begin{cases} F_{\theta_i}(y)/F_{\theta_i}(\theta_i - \bar{c}_i), & y < \theta_i - \bar{c}_i \\ 1, & y \geq \theta_i - \bar{c}_i \end{cases} \\
&\geq F_{\theta_i}(y).
\end{aligned}$$

By Theorem 4.12.3 of Grimmett and Stirzaker [2001], there exists a random variable $\tilde{Y}_i$ independent of $\{Y_j\}_{j \neq i}$ such that $\tilde{Y}_i \sim \tilde{F}_{\theta_i}$ and

$$\mathbb{P}\{\tilde{Y}_i(\omega) \leq Y_i(\omega)\} \;=\; 1.$$

Hence,

$$\begin{aligned}
\mathbb{P}\left\{\min_{j \in s} Y_j \geq \max_{k \in s^c} Y_k \;\mid\; Y_i < \theta_i - \bar{c}_i\right\} &= \mathbb{P}\left\{\min(Y_i, \min_{j \in s \setminus i} Y_j) \geq \max_{k \in s^c} Y_k \;\mid\; Y_i < \theta_i - \bar{c}_i\right\} \\
&= \mathbb{P}\left\{\min(\tilde{Y}_i, \min_{j \in s \setminus i} Y_j) \geq \max_{k \in s^c} Y_k\right\} \\
&\leq \mathbb{P}\left\{\min(Y_i, \min_{j \in s \setminus i} Y_j) \geq \max_{k \in s^c} Y_k\right\} \\
&= \mathbb{P}\left\{\min_{j \in s} Y_j \geq \max_{k \in s^c} Y_k\right\} \\
&= \mathbb{P}\{S(Y) = s\},
\end{aligned}$$

where the first step uses the independence of $\{Y_j\}$. Thus

$$\begin{aligned}\mathbb{P}\left\{\bar{V} \geq 1\right\} &\leq \bar{\lambda} \sum_{s \in \mathcal{S}_k} \sum_{i \in s} \mathbb{P}\left\{S(Y) = s\right\} \\ &= \bar{\lambda} \sum_{s \in \mathcal{S}_k} k\mathbb{P}\left\{S(Y) = s\right\} \\ &= k\bar{\lambda}. \end{aligned} \tag{5.26}$$

Combining (5.24), (5.25), and (5.26) yields the desired result. □

5.6 Comparisons

Bonferroni confidence intervals are of the form $[Y_j - c, Y_j + c]$, with

$$c = -\sup\{c : \mathbb{P}\{Y_i - \theta_i \leq c\} \leq \alpha/(2m)\}. \tag{5.27}$$

Bonferroni confidence intervals control SoP (and consequently SoS) even when $\{Y_i\}$ are dependent. When $\{Y_i\}$ are independent, Šidák confidence intervals, which are of the same form but with

$$c = -\sup\{c : \mathbb{P}\{Y_i - \theta_i \leq c\} \leq (1 - (1-\alpha)^{1/m})/2\}, \tag{5.28}$$

offer a small (but optimal) improvement on Bonferroni intervals.

Fuentes et al. [2018] construct SoS intervals for this k out of m problem with $\{Y_i - \theta_i\}$ are IID $N(0,1)$. Their intervals, which we call *FCW*, are of the form $[Y_j - c, Y_j + d]$, where c and d are constants such that

$$\left[\Phi\left(c\right) - \Phi\left(-d\right)\right]^{k-1} \times \left[\Phi^{m-k+1}\left(c\right) - \Phi^{m-k+1}\left(-d\right)\right] \geq 1 - \alpha. \tag{5.29}$$

Figure 5.15 compares the Bonferroni intervals, the Šidák intervals, FCW intervals (with $c = d$ and with c and d chosen to make the intervals as short as possible), and the SoS intervals (with δ chosen to make the intervals symmetric or as short as possible).

Because SoS intervals make inferences about only $k < m$ parameters, one would expect them to be narrower than SoP intervals—but not as narrow as intervals for a pre-specified set k parameters: there should

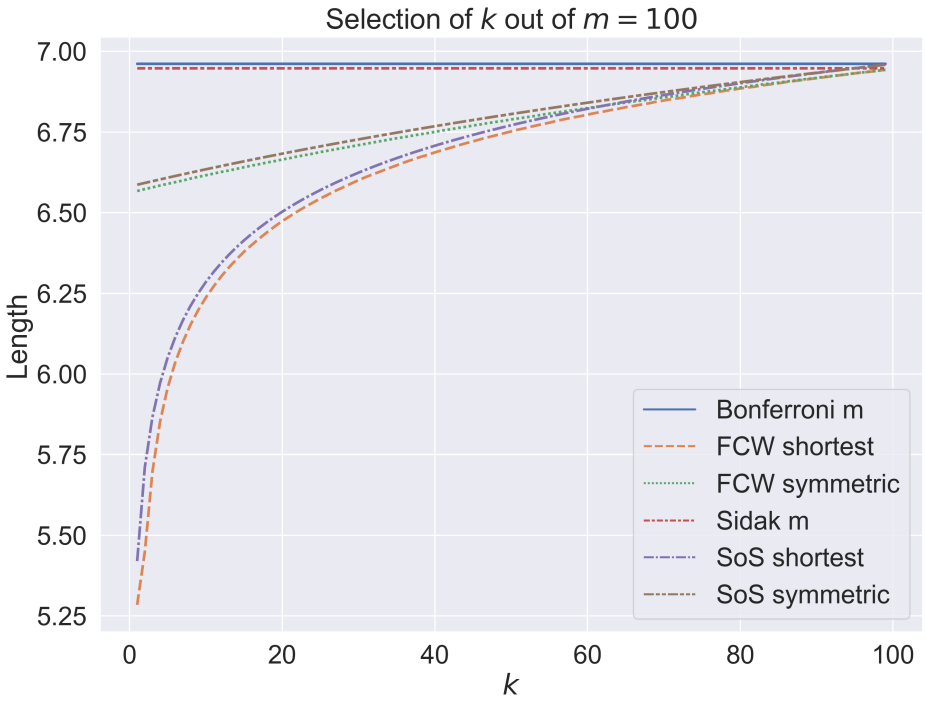


Figure 5.15: Lengths of confidence intervals with simultaneous coverage of the k of $m = 100$ normal means estimated to be largest, for $\alpha = 0.05$.

be some penalty for the selection. (The case $m = 2$, $k = 1$ is a remarkable exception.) Letting the intervals be asymmetric can reduce their length further. Suppose all the parameters are equal. Because S selects the largest k, S will preferentially select components for which $Y_i > \theta_i$. Using intervals that extend below Y_i more than they extend above Y_i allows the intervals to be shorter. Figure 5.15 shows that the advantage of allowing asymmetry is substantial when $k \ll m$. The SoS intervals are slightly wider than the FCW intervals, but the FCW intervals require $\{Y_i - \theta_i\}$ to have normal distributions, while the SoS intervals do not.

Weinstein et al. [2013], Fithian et al. [2014] and Reid et al. [2017] develop CoS intervals for the largest k of m. Controlling CoS controls FCR. These intervals also control SoS if each interval is constructed at level $1 - \alpha/|S|$, rather than $1 - \alpha$.

CoS intervals use conditional distributions, which depend on the conditioning event and the underlying parameters of the model. As a result, CoS intervals are sometimes useful and sometimes not. For instance, CoS intervals for the largest of m or largest absolute value of m can be too wide to be useful when the largest

	$\theta_1 \in (\text{Lower}, \text{Upper})$	Length
Marginal / SoS	$(0.940, 4.859)$	3.919
$\mathbb{P}_{\theta_1}(Y_1 \mid Y_1 > 2.5)$ - UMPU saturated	$(-9.090, 4.594)$	13.684
$\mathbb{P}_{(\theta_1,0)}(Y_1 \mid Y_1 > Y_2)$ - UMPU selected ($\theta_2 = 0$)	$(0.772, 4.855)$	4.082
$\mathbb{P}_{(\theta_1,2.5)}(Y_1 \mid Y_1 > Y_2)$ - UMPU selected ($\theta_2 = 2.5$)	$(-0.224, 4.558)$	4.782
$\mathbb{P}_{(\theta_1,\theta_2)}(Y_1 \mid Y_1 > Y_2)$ - UMPU selected ($\forall\theta_2 \in \Re$)	$(-\infty, 4.859)$	∞

Table 5.1: Intervals for $\alpha = 0.05$ for the bivariate normal after selecting the greater of the two estimators when $(y_1, y_2) = (2.9, 2.5)$. The depicted UMPU saturated and selected CoS intervals are from Fithian et al. [2014].

and second-largest estimators are close [Fithian et al., 2015]. Table 5.1 shows the *uniformly most powerful unbiased* (UMPU) CoS intervals from example 4 in Fithian et al. [2014] when (Y_1, Y_2) is bivariate uncorrelated normal centered at (θ_1, θ_2), for the observed value $(y_1, y_2) = (2.9, 2.5)$ and the greater $S(Y) = \{(1)\}$ is selected. In section 5.3, we showed that the unadjusted univariate interval has the right coverage. The UMPU saturated CoS intervals use $\mathbb{P}_{\theta_1}(Y_1 | Y_1 > 2.5)$. For $\theta_1 \ll 2.5$, the conditional distribution is a renormalized tail of the distribution, which yields long intervals. The UMPU selected CoS intervals use $\mathbb{P}_{(\theta_1,\theta_2)}(Y_1 \mid Y_1 > Y_2)$. Then, the underlying distribution is a function of both θ_1 and θ_2. Setting $\theta_2 = 0$ produces "nicer" intervals than $\theta_2 = 2.5$ because the selection event has less effect on the conditional distribution. But while setting the coefficient of an unselected variable to zero in a model might make sense, for location parameters, setting $\theta_2 = 0$ when it is estimated to be 2.5 does not. This shows the importance of the unselected parameters to the coverage of selected parameters. Furthermore, if a coverage were needed for all θ_2, the interval would have an infinite lower tail, since for all $\theta_1 \ll 2.5$ there exist $\theta_2 \gg 2.5$ such that $\mathbb{P}_{(\theta_1,\theta_2)}(Y_1 | Y_1 > Y_2)$ is centered at 2.9.

Because the lengths of CoS intervals depend on the observed value of Y (not only on k and m), we do not include CoS intervals in our figures or numerical comparisons.

5.7 Dependency Simulations

The conditional distribution of the variables selected is stochastically greater than the same variables independently of the selection. This gives rise to the intuition that the method should work under positive dependency. We test the intuition in simulations. Following Cai et al. [2014] we simulate each time $50,000$ observations and consider $m = 100$, $k = 10$. To demonstrate the methods is applicable when the variables

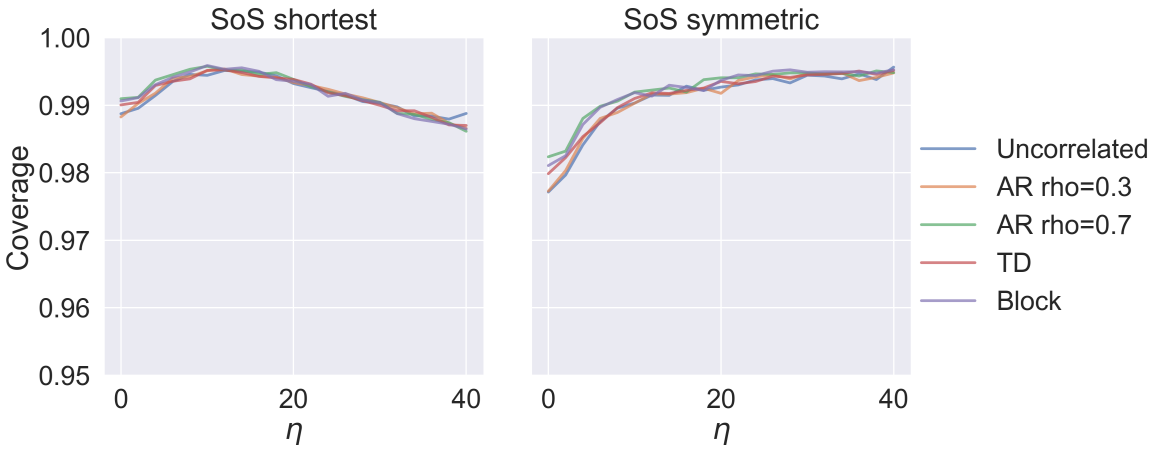


Figure 5.16: Intervals simultaneous coverage for different dependency structures. The selection of $k = 10$ of $m = 100$ estimated to be the largest out of 50 normal and 50 t_5 correlated variables for $\alpha = 0.05$.

have different distributions, we sample 50 variables of Y (denoted as $Y_{[1:50]}$) from multivariate normal $Y_{[1:50]} \sim \mathcal{N}\left(\theta_{[1:50]} \cdot \eta, \Sigma\right)$, and 50 variables (denoted as $Y_{[51:100]}$) from multivariate t with 5 degrees of freedom, $Y_{[51:100]} \sim t_5\left(\theta_{[51:100]} \cdot \eta, \Sigma\right)$. In each simulation we use the same Σ for both the normals and the t variables. For all variables, $\theta_i \sim \text{Uniform}\,(-1, 1)$ and η transition in the different simulations from 0 (identically distributed) to 40 (large differences). We consider three types of Σ covariance structures.

1. Auto Regression (AR) matrix such that $\Sigma = (\sigma_{i,j})$, and each entry $\sigma_{i,j} = \rho^{|i-j|}$ for $1 \leq i, j \leq m$ and $\rho \in \{0.3, 0.7\}$.

2. Time Decay model (TD) which taken from model 7 in Cai et al. [2014]. $\Sigma^\star = \left(\sigma^\star_{i,j}\right)$, $\sigma^\star_{ij} = |i-j|^{-5}/2$ for $j \neq i$. The covariance matrix is $\Sigma = D^{1/2}\Sigma^\star D^{1/2}$, where $D = (d_{i,j})$, such that $d_{i,j} \sim \text{Uniform}\,(1, 3)$.

3. Block Covariance matrix (Block) - 5 blocks sized 10 each, with $\rho \in \{0, 0.2, 0.5, 0.75, 0.9\}$.

Figure 5.16 shows the coverage of the SoS shortest and symmetrical intervals with different structures of dependency for $\alpha = 0.05$. All intervals obtain the desired coverage and often too conservative, demonstrating the bound might be improved in particular cases. In addition, affect of the different Σ is relatively minor and always more conservative. The major affect on the coverage is the selection distribution since the results vary significantly as a function of η.

5.8 Discussion

Katsevich and Ramdas [2018] address "simultaneous selective inference" in testing. They consider making selective inference statements on many selection rules $\{S_i(Y)\}$, guaranteeing this statements hold simultaneously with high probability. By restricting the possible selections to those determined by a particular algorithm, for instance, those obtained by varying the level at which FDR is controlled, their method can improve on full simultaneity.

We have shown that knowing the selection rule S can improve inferences, for instance by allowing one to control SoS or CoS without necessarily controlling SoP. More information is better. Lee et al. [2016] and Tibshirani et al. [2016] constructed confidence interval for parameters selected by the Lasso and by forward stepwise selection, respectively.

For SoP and FCR, it is not necessary to know S, but if S is known, improvements are possible. Berk et al. [2013] addressed the problem of inference when the model is selected because a pre-specified explanatory variable had the highest statistical significance, which restricts the family over which simultaneous coverage is required. The restriction transforms the problem into that of making a confidence interval for the coefficient whose estimate is most significantly nonzero among many correlated estimates. This amounts to selecting on the basis of the largest 1 of m correlated statistics, which can be solved computationally for problems of modest dimension. If the design matrix is orthogonal, their confidence interval amounts to the Bonferroni interval adjusted for m parameters—the dimension of the relevant family. Weinstein et al. [2013] constructed CoS intervals tailored to avoid containing zero, at the expense of being somewhat longer where it does not matter and Weinstein and Yekutieli [2014] designed FCR intervals that try to avoid covering 0. Barber and Candès [2015] took advantage of the fact that their knockoff method used forward stepwise search (their method does not yet offer confidence intervals).

To see that FCR intervals can also take advantage of knowledge of S, consider the rule S that selects the k parameters estimated to be the largest of m. This rule always selects k parameters. The proof of proposition 5.5 bounds the probability of making one or more errors via the expected number of upper endpoints that are smaller than their corresponding parameter plus the expected number of lower endpoints

that are larger than their corresponding parameters. These expectations are bounded by a multiple of α. FCR control requires the expected number of non-covering intervals, divided by k, to be at most α. Therefore, if S selects the k largest of m, FCR intervals can have lower endpoints based on the $1-(\alpha/2)\cdot(k/m)$ quantile (which works for any *simple* selection rule [Benjamini and Yekutieli, 2005]), and upper endpoints based on the $\alpha/2$ quantile instead of the $(\alpha/2)\cdot(k/m)$ quantile.

In conclusion, specification of a selection rule allows us to explore the inference ground between simultaneous and false confidence statement rate controlling confidence intervals, by defining the in-between goal of simultaneous coverage one the selected. We explored the implications of this error-rate by studying the ÃlargestÃ k-out-of-mÃ rule. This resulted in new confidence intervals. In their simplest version, the lower endpoints are Bonferroni adjusted for m, while the upper endpoints are Bonferroni adjusted only for k, offering uniform improvement over regular Bonferroni. Furthermore, the utilization of such a selection rule allows improvement over the general FCR intervals.

It seems clear that the confidence intervals retain their coverage for positive-regression-dependent estimators, because the conditional distribution of the variables selected is stochastically greater than the same variables without selection, but we do not have a formal proof. Appendix 5.7 gives numerical evidence that this is indeed the case. The intervals may also be valid when k is random, for instance when selection depends on crossing a fixed threshold or some other testing procedure.

We think the results for the k largest in absolute value of m extend to $m > 2$ and $k > 1$, but again, we offer no formal proof. We have decided to publish our current results without those generalizations with the hope that they will spark interest engage others in pursuing this approach.

Chapter 6

Interpretability Using the Input Gradient

Section based on Hechtlinger [2016]

The property of interpretability of a model can be defined in several ways. In this line of research, we are interested in the following property: given a prediction model $\hat{f} : X \to Y$, where X is the input space and Y is the output space, we would like interpret which variables in X affect the prediction of $\hat{f}$ and in which way. When this property is achieved we say the model $\hat{f}$ is interpretable.

In linear regression interpreting a given model relies on the model parameters. Since the model is of the form of $\hat{f}(x) = \hat{\beta}^T x$, there is a 1–1 correspondence between the parameters and the coordinate of the variables in the input space. Therefore, the effect of a given variable x_k on the model $\hat{f}$, can be evaluated through the value of $\hat{\beta}_k$. When $\hat{\beta}_k \gg 0$ we may say x_k has a significant positive effect on $\hat{f}(x)$. When $\hat{\beta}_k \approx 0$, we may say that x_k is irrelevant to the prediction of y, and it may get selected out of the model. Intuitively it is often also explained in terms of a slope: One unit increase in x_k would increase the prediction by the amount of $\hat{\beta}_k$.

This approach of inferring variables importance through the parameters, has been thoroughly studied by statisticians, since it is fundamental to the field of statistical inference. It has been expanded from linear regression to generalized linear models, and can also be applied to other algorithms which rely on the inner

product of the data and the variables, such as SVM. As models become more complex there is no longer a 1–1 relation between the parameters and the variables. For example, in neural networks, the number of parameters will be magnitude larger than the number of variables, and the relation between a variable and specific set of parameters is often intractable.

The key observation and main point of this work is that it is possible to interpret the model without using the parameters. Essentially it is possible to interpret any $\hat{f}$ by studying the partial derivatives of $\hat{f}$ with respect to the input variables $x \in X$. The intuition is straightforward. Suppose that $\forall x \in X$, it happens that $\frac{\partial \hat{f}}{\partial x_k}(x) = 0$ for the variable x_k , then effectively x_k doesn't influence $\hat{f}$ at all. This is also true the other way around. If $\forall x \in X$, $\left|\frac{\partial \hat{f}}{\partial x_k}(x)\right| \gg 0$, we (intuitively) learn that x_k is important to the model prediction.

For the particular case of linear regression, when $\hat{f}(x) = \hat{\beta}^T x$, the two approaches coincide since $\frac{\partial \hat{\beta}^T x}{\partial x_k} = \hat{\beta}_k$, and the parameters themselves are the partial derivatives. The major advantage of the suggested perspective is that it can be generalized to any given prediction model $\hat{f}$, both for regression and classification problems.

In what follows we show how it is possible to interpret complex models, specifically, convolutional and multi-layer neural networks using data from the field of natural language processing.

6.1 Method and Notation

Consider a prediction model $\hat{f} : X \to Y$, learned from observations of the input space X and the output space Y. This is a general framework used in supervised learning methods, including both regression, when Y is continuous and classification, when Y categorical. X is a high dimensional feature space, with p different features, $X_1, \ldots, X_p$, where X_k can also be discrete or categorical. We denote the training set as X_{train} and testing as X_{test}.

We propose to study the gradient values of $\hat{f}$ with respect to the features. Formally, an observation $x \in X$, is in the form of $x = (x_1, \ldots, x_p)$, where x_k is the value of the k-th variable (or feature). We will study the gradient $\nabla \hat{f}(x) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_p}\right)$. The k-th partial derivative will be denoted as $g_k(x) \equiv \frac{\partial \hat{f}}{\partial x_k}(x)$. We will also use the average of the partial derivatives in the test set, $\bar{g}_k = \frac{\sum_{x \in X_{test}} g_k(x)}{|X_{test}|}$, denoted in vector form as $\widetilde{g} = (\bar{g}_1, \ldots, \bar{g}_p)$.

The computation of the gradients in neural networks, and often in other models, can be derived explicitly using the backpropagation and the chain rule. This is what we have done in this paper. In other cases it may be possible to estimate it numerically. For a small h, $\frac{\partial \hat{f}}{\partial x_k} \approx \frac{\hat{f}(x_1,\ldots,x_k+h,\ldots,x_p)-\hat{f}(x_1,\ldots,x_k,\ldots,x_p)}{h}$. When x_k is continuous, this is quickly computable, and only requires two forward passes per coordinate, since $\hat{f}$ is given. In 6.2.1 and 6.2.2 we will discuss cases when x_k is categorical and $\hat{f}$ not differentiable.

In addition, the gradient, $\nabla \hat{f}$, is a function defined globally on the entire space. When it can not be analytically derived, as in most cases, it has to be evaluated at multiple points locally. In all of the examples we trained a model on the training set, and evaluated $\nabla \hat{f}$ on the test set. This is reasonable due to the standard assumption that the training and test sets are an $i.i.d$ sample from the population, but this course of action should be evaluated on a case-by-case basis.

6.2 Examples

In order to demonstrate the effectiveness of the gradients for interpretation purposes, we present 2 examples from the field of natural language processing in which complex models are being interpreted using the gradients. The examples are using the IMDB Large Movie Review Dataset Maas et al. [2011]. The dataset contains $50,000$ different movie reviews, equally partitioned to train and test sets, with Y being the labeled sentiment of the review - either positive and negative.

Implementation has been done using Keras Chollet [2015] and Theano Theano Development Team [2016]. Since Theano is doing symbolic differentiation, calculating the gradients efficiently was a simple short function call.

6.2.1 Example 1: Convolutional Neural Networks following Word Embedding

There are two major challenges with textual data. First the words are categorical and part of a dictionary. Second, significant information can be learned through the sequential order of the sentence. In recent years many state-of-the-art benchmarks in textual problems are being achieved using convolutional neural networks (CNN) models. CNN model address the first challenge by embedding the dictionary in a high dimensional

metric space, and the second by applying convolutions to capture the sequential component of the data. See **?** for model details and Kim [2014] for performance review.

Interpreting a convolutional neural networks (CNN) model with textual data is not a trivial task because the original sentence is hardly intractable after word embedding and the convolutions. Nevertheless, in this example we demonstrate how it is possible using the gradients to interpret the model.

We train a CNN on the IMDB dataset, with input sentence $x = (x_1, \dots x_{400})$, where x_k is the k-th word followed by 50 dimensional word embedding maps that transfer $x \mapsto z \in \mathbb{R}^{400\times 50}$. After the embedding, we apply 250 convolutions on z with length 3 (words), depth 50 (embedding dimension), followed by max pooling and a linear classifier. This model yields a 89.2% accuracy, not far from the state-of-the-art (topping 90% has been done a handful of times and is considered a significant improvement Johnson and Zhang [2016]).

To interpret this complex model, we turn to gradients. Since the words themselves are categorical, the prediction function is not differentiable with respect to the words. We therefore calculate for every word the partial derivative of the model with respect to the embedding vector, that is the k-th word gradient will be $\frac{\partial \hat{f}}{\partial z_k} \in \mathbb{R}^{50}$. This provides for each sentence 400 words gradients, which we rank according to their ℓ_2 norm in order to assess which ones are the most influential. The intuition toward taking the highest norm is due to the fact that the norm of the gradient is a proxy for the "magnitude" of the slope of the graph in the direction of the gradient vector, and the steepest slope will identify the word effecting the prediction the most.

Table 6.1 shows the results for several reviews in the test set. Since the convolution is being applied on length of 3 words we present the expressions activating the convolution. The highest ranked expressions are highly interpretable, and most examples are self explanatory, such as *"fantastic film total"* or *"lack of credibility"*. Some expressions were activated by the words following the selected one, such as *"ape was outstanding"*, in which the word *"ape"* had the largest gradient in the sentence, but probably explained as influential by the *"was outstanding"* expression.

Ex.1 - Positive Sentiment	Ex. 2 - Negative Sentiment	Ex. 3 - Negative Sentiment
fantastic film total	*unfortunately they suffer*	*bad as hell*
moving highly entertaining	*dull after five*	*badly stop motion*
unconvincing i can't	*painfully dull after*	*horrible you won't*
highly entertaining references	*becomes painfully dull*	*was bad as*

Table 6.1: The four most relevant expressions according to the norm of the gradients in the CNN model ordered from top to bottom. The first word in the sentence is the one selected according to the ordering of the gradient, while the following two are the rest of the expression being forwarded into a length 3 convolutions.

6.2.2 Example 2: Bag of Words Model

In the previous example we have used the gradients to interpret the local information about a particular sentence. In this example we will use the gradients globally in order to assess which words affect the classifier the most, and draw insight on the classifier functionality. To do so we represent the sentences with a simplified version of the Bag of Words (BoW) model.

A sentence $x \in X$ is represented as a binary vector in the length of the dictionary D, in which $x_d = 1$ if the d word occur in the sentence, and 0 otherwise. For this BoW representation We fit a fully connected neural network with 2 hidden layers resulting in a 85.4% model accuracy.

Next, for all $x \in X_{test}$ the partial derivatives are evaluated, and $\widetilde{g} = (\bar{g}_1, \ldots, \bar{g}_D)$, the vector of the averages of the partial derivatives is calculated. $\widetilde{g}$ provides a single global estimator of the gradient function across the input space, which can be used to rank the words according to their influence on the model. As can be seen in Table 6.2, the top positive and negative words according to the values of $\widetilde{g}$ are highly interpretable, and include words such as *"excellent"* and *"great"* for positive sentiment and *"worst"* and *"waste"* for negative sentiment.

Furthermore, $\widetilde{g}$ can also be used for approximating the prediction function $\hat{f}$, since the features are binary. By classifying x_{new} as 1 if $\langle \widetilde{g}, x_{new} \rangle > 0$ and 0 otherwise, we get a classifier that agrees with the prediction function $\hat{f}$ on $\frac{24907}{25000} \approx 99.6\%$ of the observations in the test set. This surprising result provide further insight on $\hat{f}$, specifically that (1) the decision boundary of $\hat{f}$ is closely defined by the hyper-planes created by $\widetilde{g}$, therefore the learned $\hat{f}$ is linear although the model is much more complex; (2) $\bar{g}_d$ can be used as a trustworthy estimation of influence of a given word.

Positive		Negative	
excellent	(0.096)	*worst*	(−0.160)
great	(0.091)	*waste*	(−0.117)
perfect	(0.085)	*boring*	(−0.105)
amazing	(0.082)	*awful*	(−0.103)
wonderful	(0.080)	*bad*	(−0.097)

Table 6.2: Most influential words for each sentiment to the prediction of the BoW model, ordered from top to bottom. In parenthesis we show the average of the partial derivatives over the test set.

There is a last nuance about this example that should be discussed. Although the function is not differentiable because $x_j \in \{0, 1\}$, the derivatives were obtained as if it were. Effectively it means that the derivative is being evaluated as if $x_j \in (-\epsilon, 1 + \epsilon)$. This approach can be extended to general categorical data with the use of indicators as features.

Chapter 7

Conclusion

This thesis developed new methodologies for *statistical prediction*, also called *supervised learning*. In prediction the goal is to select a "good" predictor. The straightforward approach towards "good" is accuracy. The minimization of the misclassification rate in classification or mean squared error (for example) in regression. While working on this thesis it became evident that accuracy by itself doesn't necessarily suffices as a notion of "good." The accuracy is evaluated at an advanced stage that follow many implicit decisions. For the final model to generalize well one needs to assume the training data is a truthful representation of the future data, that the selection of the model is independent from the data and that the relevant variables are used. This is not always the case. In this thesis we examined some of the issues that can occur with the standard framework.

Chapter 2 considers the classification framework. It points out that many of the existing classifiers might have a *hubristic bias*: an overconfident prediction for areas of the space which are not supported by data. This follows from the usage of $p(y \mid x)$ for prediction purposes. Since $p(y \mid x) = p(x \mid y)\, p(y) / p(x)$, most classification methods normalize by $p(x)$. This makes output a relative statement. It suggest the class that is most likely out of all possible classes. When $p(x)$ is very small, such as when predicting an outlier, the model might still provide a confident prediction.

To address this issue we suggested an approach based on $p(x \mid y)$ constructed using conformal prediction. This method is cautious and output "I don't know" in most of the space. It approaches the decision differently.

Instead of predicting the most likely class, it predicts any class which is likely based on the training data. If no class is likely it output by default "I don't know." This is useful when the distribution of future data is expected to be different from the training data — for example when facing large number of outliers.

Chapter 3 considers the regression framework. It presents the added value $p(x, y)$ has over $p(y \mid x)$ and $p(x \mid y)$ in that scenario. Specifically, $p(x, y)$ contains all the information evident within the data, while regression solutions based on $p(y \mid x)$ loses information: given a regression predictor the data distribution that generated the predictor is usually intractable. For this reason we explore level-set regression which is a conformal prediction method based on $p(x, y)$. We also compare the performance of the method with other conformal based predictions.

Chapter 4 expands an existing method to new a data structure. It suggests a extension of neural networks — a dominant prediction methodology — to graph structured input. In particular, it propose a spatial convolution on graph which generalizes the successful convolutional neural network layer. This enables the application of convolutions, which are usually applied on a grid, on a graph. The convolution operates as an inner product on the node and the k ordered nearest neighbor as learned from the graph. As explained in the chapter, this results with a graph convolution layer that perform better prediction and uses less parameters.

In chapter 5 we consider the problem of *post-selection inference*. The problem originates when the same data is used for the selection and the inference. This skew the distributions and might lead to deceptively optimistic results. In the bigger context of prediction, it can result with prediction models and statistical insights that fail to generalize as expected. In the work we construct confidence intervals that control a new error rate called "simultaneous over the selected" (SoS). The motivation is to exemplify that the knowledge of the selection rule — without the particular outcome of the selection — can be utilized to obtain shorter intervals. This is done by bounding the distribution of the statistics following the selection.

Chapter 6 studies the problem of *interpretability*. The motivation is to interpret and reason the mechanism behind the model decision. Interpretability rises from applications for which an accurate prediction doesn't suffice. In some cases there is a need to know why the prediction was made. It relates to other concepts such are fairness, privacy, reliability, causality and trust [Molnar, 2019]. The chapter introduced a simple, yet

general, method to measure which of the input variables affect the output. It used the input gradient with respect to the output, which generalizes the linear regression approach of studying the variable coefficent. This approach is applicable on complex and intractable models and is straightforward to use.

The methods developed throughout this thesis are applicable to many situations. The experiments used datasets from computer vision, natural language processing, drug discovery, phytology and genetics. Potential applications are wider. The topics selected lie in the mainstream of active statistical research. The research build upon recently published papers. The main ambition behind the works incorporated in the thesis was to try and offer straightforward conceptual contributions to problems which are general and broad.

Bibliography

James Atwood and Don Towsley. Diffusion-convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1993–2001, 2016. 53, 57, 65

James Atwood, Siddharth Pal, Don Towsley, and Ananthram Swami. Sparse diffusion-convolutional neural networks. *arXiv preprint arXiv:1710.09813*, 2017. 53, 57

Rina Foygel Barber and Emmanuel J Candès. Controlling the false discovery rate via knockoffs. *The Annals of Statistics*, 43(5):2055–2085, 2015. 96

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, pages 4502–4510, 2016. 53

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001. 60

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006. 65

Y. Benjamini and Y. Yekutieli. False discovery rate controlling confidence intervals for selected parameters. 100(469):71–80, 2005. 70, 97

Yoav Benjamini, Yotam Hechtlinger, and Philip B. Stark. Confidence intervals for selected parameters. *arXiv preprint arXiv:1905.04396*, 2019. 6, 69

R. Berk, L. Brown, A. Buja, K. Zhang, and L. Zhao. Valid post-selection inference. *The Annals of Statistics*, 41(2):802–837, 2013. 70, 96

Henry I Braun. *The collected works of John W. Tukey: Multiple comparisons, 1948-1983*. Chapman & Hall, 1994. 70

Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001. 1

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *arXiv preprint arXiv:1611.08097*, 2016. 5, 51, 52

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013. 52, 55, 64

Benoît Cadre, Bruno Pelletier, and Pierre Pudlo. Clustering by estimation of density level sets at a fixed probability. 2009. 14, 15

T Tony Cai, Weidong Liu, and Yin Xia. Two-sample test of high dimensional means under dependence. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):349–372, 2014. 94, 95

Yen-Chi Chen, Christopher R Genovese, Ryan J Tibshirani, Larry Wasserman, et al. Nonparametric modal regression. *The Annals of Statistics*, 44(2):489–514, 2016. xii, 41, 43, 44

François Chollet. Keras. `https://github.com/fchollet/keras`, 2015. 61, 101

François Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, 2016. 20

Adam Coates and Andrew Y Ng. Selecting receptive fields in deep networks. pages 2528–2536, 2011. 53

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3837–3845, 2016. 52, 56

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 20

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, pages 2215–2223, 2015. 53

Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936. 3

Ronald A Fisher. The statistical utilization of multiple measurements. *Annals of eugenics*, 8(4):376–386, 1938. 3

William Fithian, Dennis Sun, and Jonathan Taylor. Optimal inference after model selection. *arXiv preprint arXiv:1410.2597*, 2014. ix, 93, 94

William Fithian, Jonathan Taylor, Robert Tibshirani, and Ryan Tibshirani. Selective sequential model selection. *arXiv preprint arXiv:1512.02565*, 2015. 94

Claudio Fuentes, George Casella, and Martin T. Wells. Confidence intervals for the means of the selected populations. *Electronic Journal of Statistics*, 12(1):58–79, 2018. 72, 92

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*, 2017. 53

Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 729–734. IEEE, 2005. 53

Geoffrey R. Grimmett and David R. Stirzaker. *Probability and Random Processes*. Oxford University Press, August 2001. ISBN 0198572220. 91

Leying Guan and Rob Tibshirani. Prediction and outlier detection: a distribution-free prediction set with a balanced objective. *arXiv preprint arXiv:1905.04396*, 2019. 8

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 22

Yotam Hechtlinger. Interpretation of prediction models using the input gradient. *arXiv preprint arXiv:1611.07634*, 2016. 6, 99

Yotam Hechtlinger, Purvasha Chakravarti, and Jining Qin. A generalization of convolutional neural networks to graph-structured data. *arXiv preprint arXiv:1704.08165*, 2017. 4, 51

Yotam Hechtlinger, Barnabás Póczos, and Larry Wasserman. Cautious deep learning. *arXiv preprint arXiv:1805.09460*, 2018. 4, 7, 33

Yotam Hechtlinger, Niccolo Dalmasso, Alessandro Rinaldo, and Larry Wasserman. Lebesgue regression. 2019. 31, 38

Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015. 52, 56, 60, 62, 64

Radu Herbei and Marten H Wegkamp. Classification with reject option. *Canadian Journal of Statistics*, 34 (4):709–721, 2006. 7, 9

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012. 5, 51

Yosef Hochberg and Ajit C Tamhane. *Multiple comparison procedures*. Wiley, New York., 1987. 70

J. Hsu. *Multiple Comparisons: Theory and Methods*. Chapman and Hall, London, 1996. 72

J.C. Hsu. Simultaneous confidence intervals for all distances from the best. 9:1026Ã1034, 1981. 72

Rie Johnson and Tong Zhang. Supervised and semi-supervised text categorization using one-hot lstm for region embeddings. *arXiv preprint arXiv:1602.02373*, 2016. 102

E. Katsevich and A. Ramdas. Towards" simultaneous selective inference": post-hoc bounds on the false discovery proportion. *arXiv preprint arXiv:1803.06790*, 2018. 71, 96

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014. 5, 51, 102

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 63

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 52, 56, 65

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 51

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016. 22

Quoc V Le, Will Y Zou, Serena Y Yeung, and Andrew Y Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3361–3368. IEEE, 2011. 5, 51

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5, 51, 66

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. 5, 51

Jason D Lee, Dennis L Sun, Yuekai Sun, and Jonathan E Taylor. Exact post-selection inference, with application to the lasso. *The Annals of Statistics*, 44(3):907–927, 2016. 96

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177, 2018. 9

Adrien Marie Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805. 2

Jing Lei. Classification with confidence. *Biometrika*, 101(4):755–769, 2014. 8, 14

Jing Lei and Larry Wasserman. Distribution-free prediction bands for non-parametric regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):71–96, 2014. 8, 13, 15, 34, 43

Jing Lei, James Robins, and Larry Wasserman. Distribution-free prediction sets. *Journal of the American Statistical Association*, 108(501):278–287, 2013. 8, 13, 14, 16, 34

Jing Lei, Max GâSell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018. 39

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015. 53

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015. 24

László Lovász et al. Random walks on graphs: A survey. *Combinatorics, Paul Erdos is Eighty*, 2:353–398, 1996. 55

Junshui Ma, Robert P Sheridan, Andy Liaw, George E Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure–activity relationships. *Journal of chemical information and modeling*, 55 (2):263–274, 2015. 43, 64

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association*

for Computational Linguistics: Human Language Technologies-Volume 1, pages 142–150. Association for Computational Linguistics, 2011. 101

Christoph Molnar. *Interpretable Machine Learning*. 2019. `https://christophm.github.io/interpretable-ml-book/`. 106

Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proc. CVPR*, volume 1, page 3, 2017. 53, 57, 65

Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023, 2016. 53

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014. 65

Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox v0. 8.0: A python toolbox to benchmark the robustness of machine learning models. *arXiv preprint arXiv:1707.04131*, 2017. 22

Stephen Reid, Jonathan Taylor, and Robert Tibshirani. Post-selection point and interval estimation of signal sizes in gaussian samples. *Canadian Journal of Statistics*, 45(2):128–148, 2017. 93

Yaniv Romano, Evan Patterson, and Emmanuel J Candès. Conformalized quantile regression. *arXiv preprint arXiv:1905.03222*, 2019. 40, 42

Rasmus Rothe, Radu Timofte, and Luc Van Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision (IJCV)*, July 2016. 24

Mauricio Sadinle, Jing Lei, and Larry Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234, 2019. 8, 13, 14, 18

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. 53

Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(Mar):371–421, 2008. 4

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 5, 51

Stephen M Stigler. *The history of statistics: The measurement of uncertainty before 1900*. Harvard University Press, 1986. 2

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015. 20

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017. 20, 23

Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 426–433. IEEE, 2016. 22

J. Taylor and R.J. Tibshirani. Statistical learning and selective inference. *Proceedings of the National Academy of Sciences*, 112(25):7629–7634, 2015. 72

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. 101

Ryan J Tibshirani, Jonathan Taylor, Richard Lockhart, and Robert Tibshirani. Exact post-selection inference for sequential regression procedures. *Journal of the American Statistical Association*, 111(514):600–620, 2016. 96

John W Tukey. *The problem of multiple comparisons.* Unpublished manuscript. In The Collected Works of John W. Tukey VIII. Multiple Comparisons: 1948â1983, 1â300. Chapman and Hall, New York., 1953. 70

JH Venter. Confidence bounds based on the largest treatment mean. *South African Journal of Science*, 84(5): 340, 1988. 72

Vladimir Vovk. Cross-conformal predictors. *Annals of Mathematics and Artificial Intelligence*, 74(1-2):9–28, 2015. 13

Vladimir Vovk, David Lindsay, Ilia Nouretdinov, and Alex Gammerman. Mondrian confidence machine. *Technical Report*, 2003. 8, 14

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005. 4, 7, 8, 11, 13, 39

Ronald L. Wasserstein and Nicole A. Lazar. The asa's statement on p-values: Context, process, and purpose. *The American Statistician*, 70(2):129–133, 2016. 69

A. Weinstein, W. Fithian, and Y. Benjamini. Selection adjusted confidence intervals with more power to determine the sign. *Journal of the American Statistical Association*, 108(501):165–176, 2013. 93, 96

Asaf Weinstein and Daniel Yekutieli. Selective sign-determining multiple confidence intervals with fcr control. *arXiv preprint arXiv:1404.7403*, 2014. 96

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012. 65

Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016. 65

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003. 65