Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/44/20230249

Distributionally Robust Optimization methods on robust medical diagnosis systems

Yumeng Tang[1,6,11], Ziming Cui[2,7], Xishi Wang[3,8], Shuyu Xiang[4,9], Yifeng Li[5,10]

[1]Faculty of Electronic Information and Engineering, Tongji University, Shanghai, 201804, China
[2]Case School of Engineering, Case Western Reserve University, Cleveland, 10900 Euclid Ave, Cleveland, OH 44106, America
[3]School of Big Data and Information Engineering, Guizhou University, Guizhou, 550025, China
[4]Woodsworth College, University of Toronto, ON M5S1A1, Canada
[5]Alevel, Changjun High School International Department, Changsha, 410013, China

[6]tonmoregulus@gmail.com
[7]ziming.cui1998@gmail.com
[8]2047665441@qq.com
[9]shuyu829@gmail.com
[10]157147113@qq.com.
[11]corresponding author

Abstract. In the medical field, modern recommendation systems face significant challenges due to distributional shifts in data. We propose utilizing Distributionally Robust Optimization (DRO) and Distributionally and Outlier Robust Optimization (DORO) methods to address this issue. This paper aims to develop suitable DRO and DORO frameworks for the medical domain and validate their effectiveness through extensive experiments. We employ the DDXPlus dataset for our investigations and cluster patients based on age, sex, and initial evidence to partition the data into distinct distributions. Using a simple three-layer neural network, we incorporate CVaR and CHISQ as DRO methods and their respective DORO forms. The experimental results show that the overall DRO approach demonstrates more significant enhancements while all four methods exhibit improvements over the original distributional scenarios. Our research contributes to optimizing deep learning models in the medical domain and enhancing their robustness. Furthermore, we intend to use these methods to estimate and provide best-fit patient therapies, addressing real-world medical challenges. The application of these approaches has the potential to enhance the performance and practicality of medical recommendation systems, offering improved medical services to patients.

Keywords: Distributionally Robust Optimization, Medical Diagnosis, DDXPlus dataset.

1. Introduction

Machine learning mechanisms are widely employed in the medical field to customize the best therapy methods for diverse patients. However, distributional shifts in the data sets and training sets often lead to inaccuracies in assessing patients' conditions. For example, if a machine learning model is trained on a dataset comprising female lung cancer patients, its ability to provide suitable therapies for male lung cancer patients might be compromised.

To tackle this issue, we adopt a Distributionally Robust Optimization (DRO) framework to mitigate the impact of distributional shifts [1,2]. This approach allows decision-making to transcend reliance on a single "most likely" scenario and instead considers a range of plausible scenarios. By providing a worst-case estimate, this method enhances the robustness and generalization capabilities of predictive models. Moreover, its ability to handle a broader spectrum of uncertainties, including stochastic, non-parametric, and data-driven uncertainties, makes it well-suited for real-world problems where data is uncertain, incomplete, or subject to change.

However, in practical medical scenarios, numerous patients exhibit severe conditions that significantly deviate from typical cases, giving rise to outliers. These outliers can potentially disrupt the accuracy of machine learning algorithms within the DRO framework [3]. As medical conditions often progress through several stages, patients' conditions may vary significantly at each stage. This variation can lead to a challenge where the condition of a single patient, if notably worse than others, can disproportionately influence the DRO-based machine, resulting in a solution unsuitable for the majority. In contrast, the Distributionally and Outliers Robust Optimization (DORO) framework disregards outliers to provide a more robust, reliable, and consistent solution, even in challenging and uncertain conditions for the majority [4]. By combining the principles of DRO and outlier robustness, the DORO framework creates models that effectively handle a variety of uncertain scenarios, including those involving outliers. Hence, our work leverages the DRO framework and incorporates DORO methods to address these challenges effectively.

2. Related Work

2.1. DDX-Plus Dataset

DDXPlus has yielded much experimental data for building AD and ASD systems. This dataset is formatted similarly to publicly available datasets but differs in crucial ways [5].

First, in the large DDX Plus sample, clinical symptoms are clearly distinguished from causality in doctor-patient interactions, and the roles of the two in doctor-patient interactions are not identical. Second, the dataset is much larger. Third, unlike pre-existing data containing only dichotomous signs and causal relationships, DDXPlus also has categories, multiple-choice symbols, and causal relationships. Different characters and precursors of the condition are chosen to collect data better. In addition, some of the symptoms are hierarchical, so they can interact with the patient rationally. Ultimately, each patient receives a precise diagnosis and actual pathology, and the DDXPlus dataset extends existing diagnostic systems for AD and ASD by incorporating and validating their clinical utility and comparing them with discriminatory diagnoses to understand their reasoning better.

This information collection includes approximately 1.3 million patients and is built on a private medical knowledge base. This knowledge base collects a large amount of medical literature. It is categorized according to the main symptoms, which include coughing, sore throat, or difficulty in breathing that accompanies these symptoms. The knowledge base organizes evidence into binary, categorical, and multiple-choice systems. Based on this, an automatic knowledge-based diagnosis method is proposed. Each patient has information about their age, gender, symptom descriptions, and associated differential diagnoses. Compared to existing datasets in the autonomy diagnosis and autonomy spectrum disorder literature, this dataset has the following benefits:

1. Unlike the SymCAT and the Muzhi datasets, this dataset contains diverse evidence types, including categorical and multiple-choice evidence, which can better match how physicians ask patients questions and improve the efficiency of evidence collection.
2. A clear distinction is made between antecedents and symptoms.
3. The severity of each pathology is flagged to help design solutions that can address severe pathologies.

In summary, this dataset contains differential diagnostic information, diverse types of evidence, clearly differentiated antecedents and symptoms, and labeled pathology severity levels. These features make this dataset an essential advantage in autonomic diagnosis and spectrum disorder research [6].

2.2. Distributionally Robust Optimization

With the wide application of data-driven decision-making in various fields, addressing the uncertainty and outliers in real-world data has become crucial. Against this backdrop, Distributionally Robust Optimization (DRO) and Distributionally and Outlier Robust Optimization (DORO) have been proposed and extensively studied.

The theoretical foundation of distributionally robust optimization presumes that uncertain parameters in the optimization problem might come from a tentative set of distributions. The aim is to identify a solution that performs well across all potential distributions. This is an effective way of dealing with model parameter uncertainty, especially applicable to real-world problems characterized by high data uncertainty, incompleteness, or volatility. The core of this approach lies in constructing an appropriate set of distributions that captures the tension without being overly conservative. Generally, this construction requires considering various distribution characteristics, such as mean, variance, and skewness. In this process, concepts such as Conditional Value at Risk (CVaR) [7] and Chi-square distribution (CHISQ) [8] are extensively used to describe uncertainty.

However, while DRO methods have achieved some success in dealing with distributional uncertainty, they do not fully consider the impact of outliers on the optimization outcome. In specific practical applications, outliers can significantly affect the optimization results. For instance, in healthcare, our data often come from different types of patients with varying health conditions. These differences may result in outliers in the data, significantly distinct from the majority, which can severely affect our optimization results, reducing the predictive performance of our model.

Researchers have proposed the Distributionally and Outlier Robust Optimization (DORO) framework to address this issue. DORO not only considers the uncertainty of the distribution but also potential outliers. It combines the basic idea of DRO with outlier robust optimization techniques, aiming to create a model that performs well even under various uncertain conditions, including outliers. Specifically, DORO seeks an optimal strategy that performs well under all possible distributions and outliers.

Distributionally robust optimization methods have been widely applied in several domains, such as supply chain management, financial engineering, and energy systems. However, in healthcare, unresolved challenges persist, such as handling the impact of outliers and addressing the differing data distributions among various types of patients. In this study, we apply DRO and DORO methods to the healthcare domain, aiming to address the issues above and provide more robust and reliable data-driven solutions for medical decision-making.

3. Methodology

This paper employed four different methods to train on the DDXplus medical dataset: CVaR, CVaR+DORO, CHISQ, and CHISQ+DORO. Below, we provide detailed descriptions of these methods.

3.1. CVaR

The CVaR method deals with distribution uncertainty by minimizing the tail risk of the uncertain distribution. It concentrates on the most severe losses by sorting the losses and choosing the top alpha portion. This strategy helps improve the model's performance when facing the worst-case scenarios. The pseudocode below is a simplification of the CVaR portion in our code.

```
train:
    for epoch:
        outputs = model(inputs)
        loss = criterion(outputs,targets)

        n = int(alpha * len(inputs))
        rk = torch.argsort(loss, descending=True)
        loss = loss[rk[:n]].mean()

        loss.backward()
        optimizer.step()
```

During the training process, the CVaR method works as follows: Firstly, the model generates predicted outputs using input data and then calculates the loss between the results and the targets. This loss is then used to calculate the CVaR risk measure. Specifically, it calculates the number of worst loss samples to be considered (determined by the product of the alpha parameter and the number of input samples), then sorts the losses in descending order, takes the top n pieces with the highest losses, and calculates the average loss of these samples as the CVaR value for this iteration. Once the CVaR value is obtained, the loss gradient concerning the model parameters is calculated based on this value, and the optimizer updates the model parameters. In this way, the model is adjusted to minimize the average loss under the worst-case scenario (maximum loss) in each training iteration, thereby improving the model's predictive performance under uncertainty.

3.2. CVaR+DORO

The CVaR+DORO method considers both the distribution's uncertainty and possible outliers. Firstly, it sorts the losses in descending order and then selects them from the eps percentile to the gamma percentile. In this way, it tries to ignore possible outliers, thereby enhancing the robustness of the model. This is the pseudocode derived from the CVaR+DORO part of our code.

```
train:
    gamma = eps + alpha * (1 - eps)
    for epoch:
        outputs = model(inputs)
        loss = criterion(outputs,targets)

        n1 = int(gamma * len(inputs))
        n2 = int(eps * len(inputs))
        rk = torch.argsort(loss, descending=True)
        loss = loss[rk[n2:n1]].sum() / alpha / (len(inputs) - n2)

        loss.backward()
        optimizer.step()
```

In the training process of CVaR+DORO, an extra parameter, epsilon, is introduced to handle outliers in the data. First, we calculate an adjusted alpha value, namely gamma, derived from combining the original and epsilon parameters. Then, we compute two indices n1 and n2, which are the integer parts of the products of gamma and epsilon with the number of input samples. We sort the losses in descending order, then select the losses that rank between n2 and n1 for processing. The purpose of this operation is that the samples before n2 might be outliers in the data, so we choose to ignore them; while the pieces after n1 have more minor losses, and their contributions to the CVaR value are relatively small, so we mainly focus on the losses between n2 and n1. The sum of these losses, divided by alpha and the number of samples after deducting n2, becomes the loss for this iteration. Finally, based on this loss, we calculate the gradient and use the optimizer to update the parameters. By this method, we consider both the original CVaR optimization and the robustness to outliers, making the model perform well in facing uncertainties and outliers.

3.3. CHISQ

The CHISQ method uses the chi-square distribution to characterize uncertainty, dealing with the distribution's uncertainty by minimizing the loss's chi-square statistic. The CHISQ method seeks the optimal eta value within the square error of the failure such that the sum of eta and the standard deviation of the loss multiplied by a factor C is minimized. This is a simplified pseudocode derived from the CHISQ part in our code.

```
train:
    max_l = 10
    C = math.sqrt(1 + (1/alpha - 1) ** 2)
    for epoch:
        outputs = model(inputs)
        loss = criterion(outputs,targets)

        foo = lambda eta: C * math.sqrt((F.relu(loss - eta) ** 2).mean().item()) + eta
        opt_eta = sopt.brent(foo, brack=(0, max_l))
        loss = C * torch.sqrt((F.relu(loss - opt_eta) ** 2).mean()) + opt_eta

        loss.backward()
        optimizer.step()
```

In the training process of the CHISQ method, a parameter max_l is first set, which is used to define the upper limit of our search range for the optimal eta. Here, eta is a dynamically adjusted threshold used to measure uncertainty risk. Then, we define a coefficient C, calculated based on the alpha parameter we set. Alpha is a risk-averse parameter; when alpha approaches 1, the value of C will also come 1, implying that we lean towards minimizing average risk; when alpha deviates from 1, the value of C will increase, signifying a focus on the tail risk, i.e., larger risk values. In each training cycle, we get the prediction result through the model and then calculate the loss between the forecast and the target. Following this, we define a function 'foo' that takes an eta value as input and returns the risk measurement value with eta as the threshold. This measure is a weighted sum of the squared loss and eta. Then, using the Brent method, we find the eta value that minimizes the 'foo' function within the range of 0 to max_l, i.e., opt_eta. This opt_eta is essentially a balance that reduces the risk measurement value. Finally, we take opt_eta as the threshold, calculate a new loss value, and perform gradient computation and parameter updates. Thus, in each training iteration, the CHISQ method seeks an optimal risk threshold, enhancing the model's robustness to tail risk and thereby improving the model's overall performance.

3.4. CHISQ+DORO

The CHISQ+DORO method combines the chi-square method, which handles distribution uncertainty, with the DORO method, which takes outliers. When calculating the square error of the loss and looking for the optimal eta value, it disregards the failure of the largest eps percentile. This helps to reduce the possible impact of outliers on the optimization results, improving the robustness of the model. Below is the pseudo-code simplified from the CHISQ+DORO part of our code.

```
train:
    max_l = 10
    C = math.sqrt(1 + (1/alpha - 1) ** 2)
    for epoch:
        outputs = model(inputs)
        loss = criterion(outputs,targets)

        n = int(eps * len(inputs))
        rk = torch.argsort(loss,descending=True)
        l0 = loss[rk[n:]]
        foo = lambda eta: C * math.sqrt((F.relu(l0 - eta) ** 2).mean().item()) + eta
        opt_eta = sopt.brent(foo, brack=(0, max_l))
        loss = C * torch.sqrt((F.relu(l0 - opt_eta) ** 2).mean()) + opt_eta

        loss.backward()
        optimizer.step()
```

In the training process of the CHISQ+DORO method, we first define a maximum loss upper limit, max_l, and a coefficient, C, derived from the alpha parameter. Then, the model generates prediction results for each training cycle and calculates the loss between the predicted results and the target. Next, we calculate an index, n, through a scaling factor, eps (representing the expected proportion of outliers). We sort the loss values in descending order and take all loss values after the index n, i.e., those relatively significant losses (potentially outliers) we are primarily concerned with. After that, we define a function, foo, which accepts a threshold value, eta, and returns a risk metric based on eta. As with the CHISQ method, this metric is the weighted sum of squared loss and eta, but here, the squared loss is calculated only for those losses with an index greater than n, which are the potential outliers we are concerned with. Subsequently, we use the Brent optimization method to search for the eta value, opt_eta, that minimizes the foo function within the range from 0 to max_l. Finally, we use this optimal threshold, opt_eta, to calculate the new risk metric, then perform gradient calculation and parameter updating. In summary, the CHISQ+DORO method dynamically adjusts the risk threshold. It pays more attention to potential outliers, making the model more robust during training and better equipped to handle tail risk.

4. Experiment

4.1. Setup

Dataset Splitting K-means is a clustering algorithm that groups data points into K clusters based on their similarities [9]. We used the primary purpose of K-means clustering to discover inherent patterns and structures within the data when the actual labels or classes are unknown. Thus, we can use K-means clustering to help split DDXPlus dataset into training and testing sets. This helps in extracting underlying patterns and structures present in the data.

Domain Definition On DDXPlus [5], we define six domains (subpopulations). Our domain definitions cover several types of subpopulation shift, such as different ages, differential diagnosis, sex, pathology, evidence, and initial evidence. The two domains of differential diagnosis and proof are probabilistic data, and they can cause problems in the model training process, so we remove them from the dataset. With the probabilistic data released, the inputs and outputs of the model are more explicit and transparent, making it easier to build an appropriate model.

Training Compared to the regression models [10], we use a three-layer feed-forward neural network activated by ReLU on DDXPLUS. On each dataset, we run Based, CVaR, Chi^2-DRO, CVaR-DORO, and Chi^2-DORO[4]. Each algorithm is run 50 epochs on DDXPLUS. We collect the model achieved at the end of every age for each method and select the best model through validation.

Model Selection Given that the test set includes information about each instance's domain membership, the optimal model is selected based on achieving the highest worst-case test accuracy.

However, it should be noted that choosing a model for validation without group labels proves to be a challenging undertaking based on our trials. Nonetheless, it is essential to clarify that model selection is not the main focus of this study.

4.2. Figures and Tables

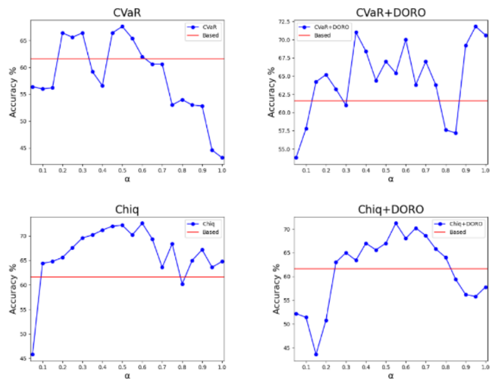

Figure 1. The Based, DRO and DORO accuracy rate with different alpha (ε = 0.01).

Table 1. The accuracy rate of Based, DRO, and DORO on best models. (%)

Dataset	Method	Best Accuracy
DDX-Plus	Based	61.60 ± 5.46
	CVaR	67.60 ± 3.85
	CVaR-DORO	71.80 ± 2.17
	Chiq-DRO	72.60 ± 2.07
	Chiq-DORO	71.20 ± 3.56

5. Analysis

From the experimental results in Figure 1 and Table 1, we can see that DRO and DORO significantly improve the accuracy of the experimental results. In the original case, our accuracy can only reach about 61.6%, but by DRO and DORO our accuracy value can be improved to more than 70% in the optimal condition. In our experiments, we found that the volatility of Cvar is intense, which is manifested in the fact that the accuracy is much higher than the original function when the Alpha value is in the intervals of 0.2-0.3 and 0.45-0.6. In contrast, the accuracy value gradually decreases under other value conditions. In our experiments, we found that the volatility of Cvar is intense, which is manifested in the fact that the accuracy is much higher than the original function when the Alpha value is in the intervals of 0.2-0.3 and 0.45-0.6.

In contrast, the accuracy value gradually decreases in other value conditions. When we add DORO in Cvar, while keeping the eps value constant at 0.01, the accuracy of the experimental results generally shows an upward trend, and the total number of experimental accuracy values can be guaranteed to be greater than the initial value when the Alpha value is taken to be greater than 0.15. In the experimental result graph of Chisq, we can find that when the Alpha value is in the interval of 0.1-0.5, the practical result accuracy value is always more significant than the original state accuracy value. With the increase of the Alpha value, the accuracy value rises gradually; when the Alpha value is more important than 0.5, the overall trend of its accuracy value decreases and fluctuates up and down in a particular range, but the overall accuracy value is still higher than that of the initial state. When we add DORO in the Chisq condition, we still control the EPS value to be constant at 0.01, and get the results; we find that when Alpha takes a value between 0.25-0.8, the accuracy value is always higher than the initial state accuracy value. Compared with DRO in the Chisq condition, the DORO fluctuates within a specific range. In the case of Alpha taking a weight of 0.55, the accuracy value is always higher than the initial state accuracy value. However, the overall accuracy value is still higher than the initial state accuracy value. Reaches the highest value and then shows a decreasing trend.

For the above phenomenon, we believe that the main reason is that both DRO and DORO select only the part of the loss values with more significant deviations and ignore the piece with too small deviation values, thus improving the overall accuracy. In Cvar, we can see that adding DRO and DORO significantly improves the experimental results because we optimize the bias caused by the worst-case scenario. The large gap in the DRO results compared to DORO is due to the sensitivity of DRO to outliers in the dataset, which is shown in DORO: Distributional and Outlier Robust Optimization: Distributional and Outlier Robust Optimization, 2021. In Chisq, this conclusion is even more apparent, as the DRO is more volatile when Alpha takes a more significant value. In contrast, the change in the precision value is smoother with the addition of DORO, even though the precision value still fluctuates up and down.

6. Conclusion

In this work, we have inserted DRO and DORO models on the original DDXPlus to improve the actual data's accuracy value and provide better prediction capability. Also, we found that the accuracy of the experimental results of DORO can be maintained at a high level when the Alpha value is taken in the interval of 0. 4-0.75 when making a prediction. In contrast, DRO's highest accuracy value range is more demanding in Cvar. The main contribution of this work is that when a specific symptom is present, combined with information about its age and gender, we can make a quick assessment based on the data to determine the disease, reducing the determination time and improving the diagnostic efficiency. From the above experiments, we can conclude that adding DRO or DORO to the assessment and prediction process can significantly improve the prediction accuracy and reduce the probability of misdiagnosis. The effect of DORO is better than DRO.

Although our experiments successfully demonstrated that adding DRO and DORO can significantly improve the experimental precision with a constant value of eps of 0.01, there are still some potential directions to explore and some questions that require more in-depth experiments to be answered. Specifically, we hope to find larger data sets, conduct multiple experiments, and find more helpful values for accuracy improvement by continuously adjusting the values of eps and alpha. At the same time, we want to build more extensive neural convolutional networks to support more complex deep learning requirements.

References

[1] Erick Delage, Yinyu Ye, Distributionally Robust Optimization Under Moment Uncertainty with Application to Data-Driven Problems. Operations Research 58(3):595-612, 2010.

[2] Fengming Lin, Xiaolei Fang, Zheming Gao. Distributionally Robust Optimization: A review on theory and applications. Numerical Algebra, Control and Optimization, 2022, 12(1): 159-212. doi: 10.3934/naco.2021057.

[3] Hamed Rahimian, Sanjay Mehrotra, Distributionally Robust Optimization: A Review, in: Open Journal of Mathematical Optimization, Volume 3 (2022), article no. 4.

[4] R. Zhai, C. Dan, J.Z. Kolter, P. Ravikumar, DORO: Distributional and Outlier Robust Optimization, in: Proceedings of the 38th International Conference on Machine Learning, PMLR 139:12345-12355, 2021.

[5] A.F. Tchango, R. Goel, Z. Wen, J. Martel, J. Ghosn, DDXPlus: A New Dataset For Automatic Medical Diagnosis, in: Proceedings of the Neural Information Processing Systems-Track on Datasets and Benchmarks, 2, 2022.

[6] Jindong Wang, Xixu Hu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Haojun Huang, Wei Ye, Xiubo Geng, Binxin Jiao, Yue Zhang, Xing Xie "On the Robustness of ChatGPT: An Adversarial and Out-ofdistribution Perspective" arXiv:2302.12095v4 [cs.AI] for this version.

[7] Daniel Levy, Yair Carmon, John C. Duchi, Aaron Sidford, Large-Scale Methods for Distributionally Robust Optimization, in: Advances in Neural Information Processing Systems 33 (NeurIPS 2020).

[8] Lancaster, H. O., and Seneta, E. (2005). Chi-square distribution, p. armitage and t. Colton in Encyclopedia of Biostatistics, Wiley, Chichester.

[9] J.A. Hartigan, M.A. Wong, Algorithm AS 136: A K-Means Clustering Algorithm, in: Journal of the Royal Statistical Society. Series C (Applied Statistics), Vol. 28, No.1 (1979), pp. 100-108. DOI: https://doi.org/10.2307/2346830.

[10] Ruidi Chen, Ioannis Ch. Paschalidis, A Robust Learning Approach for Regression Models Based on Distributionally Robust Optimization, in: Journal of Machine Learning Research 19 (2018) 1-48.