Machine Learning for Information Extraction from Pathology Reports and Adaptive Offline Value Estimation in Reinforcement Learning

by

Briton Park

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Statistics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Bin Yu, Chair
Professor Haiyan Huang
Assistant Professor Samuel Pimentel

Spring 2022

Abstract

Machine Learning for Information Extraction from Pathology Reports and Adaptive Offline Value Estimation in Reinforcement Learning

by

Briton Park

Doctor of Philosophy in Statistics

University of California, Berkeley

Professor Bin Yu, Chair

The thesis is divided into two parts. The first part focuses on a healthcare-related application of machine learning, and the second part focuses on offline evaluation of reinforcement learning agents, which is critical for estimating values of policies in high-risk and high-cost applications of reinforcement learning, such as patient care.

The first part is comprised by the pathology report parsing work on data from UCSF. Personalized medicine has the potential to revolutionize healthcare by enabling practitioners to tailor treatment and assessment for each individual patient. However, personalized care depends on the ability to leverage patient data intelligently. One major source of clinical data are pathology reports which are currently stored electronically at clinical institutions. However, much of the pathology report data cannot be easily leveraged since they exist in unstructured and semi-structured text; they must be parsed in a structured form before being used in downstream clinical applications. Furthermore, manual extraction of the data is a time-consuming and expensive process for a human annotator. Thus, researchers have studied ways to algorithmically parse reports via machine learning. Despite advancements in machine learning, particularly deep learning, building accurate parsers is still challenging due to the amount of training data that is required. In our work we focus on the sample efficiency of machine learning parsers using limited annotations. In the first part of the thesis, we develop machine learning-based data extraction methods for pathology reports at UCSF based on natural language processing with limited training data sizes. For each specific data field, such as the location of the tumor or the stage of the cancer, we train a model to automatically extract the information from each report using a limited set of human annotations as the training targets. We also analyze the sample efficiency of state-of-the-art methods compared to our approaches and study practical considerations in the deployment of such data extraction systems. We find that our proposed algorithms are able to achieve accuracies comparable to the state-of-the-art using fewer annotated data points.

In the second section, we focus on offline reinforcement learning, which is a data driven approach for reinforcement learning. Offline reinforcement learning relies on learning from static datasets, unlike the less restrictive, online setting which assumes learning is done via a feedback loop between the agent and the environment. The offline setting is crucial for applications in healthcare and robotics where deploying untrained or partially trained agents can be costly or dangerous. Leveraging historical data without further data collection is a unique challenge, because validating models must be done on data deriving from a different model or set of models. In chapter 5, we focus on adaptive weighting of predictions based on model stability for the goal of evaluating reinforcement learning policies, which is known as offline evaluation. We experiment with two state-of-the-art evaluation methods: fitted Q-evaluation and model-based evaluation. We propose a new estimator based on weighting each model based on conditional stability estimates via ensembling, which is inspired by online weighting of predictors for online prediction [11, 5] and the pessimistic reinforcement learning literature [40, 46]. We benchmark the offline evaluation methods on simulated environments detailed in chapter 5. We find that stability stemming from ensembling is a promising avenue for adaptively weighting model estimates in the setting where model selection and validation is difficult.

To my family

Contents

Acknowledgments

I would like to give special thanks to my collaborators and mentors who I have worked with over the years. Their help and guidance have made my PhD possible and have allowed me to grow both academically and personally.

First, I would like to acknowledge my advisor Bin Yu who has made an immense impact on my life. She has been a great catalyst for my growth over the years, and I am lucky to have been mentored by such a great role model as a researcher and leader. As my advisor, I am thankful for her encouragement and talks through difficult moments of my journey as a PhD student of which there were many and her growth mindset perspective in academics and life. I would like to give special thanks for her patience with me. I am also extremely thankful for the collaborations and opportunities that were not possible without her support.

I am also grateful for Anobel Odisho who I consider as an informal advisor throughout my PhD. He has been a constant source of help and guidance for navigating UCSF protocols and resources, setting up workstations, providing constant medical expertise, and more. The UCSF project which comprises the majority of my thesis, would not have been possible without Anobel.

I would also like to give special acknowledgement to Nick Altieri who I have worked with together on the UCSF pathology project and who I consider as a mentor over the first three years of the PhD. I have learned so much from Nick on how to manage myself as a researcher and PhD student and effectively communicate with others. He set a great example to strive for as a senior PhD student. I would also like to thank Yan Shuo Tan and Raaz Dwivedi who I worked with on the stable causal subgroups project. They have also set great examples of researchers and mentor and have helped me to transition to being a senior student in our research group. I would also like to thank Mian Wei who I had the privilege to work with for her contributions to the project. I would like to also thank Angela Zhou and Carrie Wu on providing feedback and expertise and helping me navigate a project in an area I was new to and not comfortable with. I would like to thank select members of the Yu Group who've I talked to along the way including Jamie Murdoch, Hue Wang, Tiffany Tang, Chandan Singh, Xiao Li, Wooseok Ha, James Duncan, Abhi Agarwal, Robert Netzorg, Aliyah Hsu, Rebecca Barter, Soren Kunzel, and Yuansi Chen. Outside the group, I would like to especially thank Olivia Angiuli for being a great office mate, keeping tabs on me and the discussions we've had over the years.

Next, I am grateful for my collaborators at the Information Comments at UCSF which include Eugenia Rutenberg, Dima Liutiev, and Lakshmi Radhakrishnan. Completing and deploying the pathology parser would not have been possible without their support.

Throughout the last couple years I have had great experiences as an intern. I am fortunate to have had the opportunity to collaborate with Levi Boyles, Eren Manavoglu, Shuayb Zarar, Howard Tai, and Guangchao Yuan. Most recently at Citadel Securities I am thankful to have worked on stable feature selection problems with Tao Shi and Bai Jiang who have both helped me in deciding my post-PhD plans.

I have been fortunate to have been supported financially by the Citadel Fellowship, a UCSF research grant from the Bakar Institute of Computational Health Science, Professor Peng Ding's support via the RTG grant from NSF, and Professor Bin Yu's graduate resesearch funding.

Additionally, I would like to thank La Shana in the statistics department, who has helped me navigate administrative and logistical aspects of the PhD program over the years, as well as Sam Pimentel, Haiyan Huang, and Avi Feller for serving on my qual committee.

I am grateful for my girlfriend Heidi Kim for constantly supporting me throughout my PhD and being my best friend. And also to Changmin Kook who I shared great memories with during the first 2 years of my PhD and have helped balance out my life, and one of my longest-known friends Phong Nguy for all the discussions we've had and time spent over the past year to decompress. Lastly, I would like to thank my family for supporting me unconditionally throughout my PhD and my entire life. None of this would have been possible without their help.

Chapter 1

Overview

1.1 Extracting Information Across Pathology Reports Across UCSF

Personalized healthcare depends on detailed and accurate patient data. The massive amounts of unstructured medical text in electronic health records are a primary source of this data, and the ability to reliably extract clinical information is a crucial enabling technology. Unfortunately, much of the relevant clinical data, such as cancer stage and histology, are stored as free text in lengthy unstructured or semi-structured reports. [9] Leveraging the data contained in these reports for precision medicine applications relies on manual efforts by annotators with domain expertise for many downstream automated methods. As a result, there has been much interest in natural language processing and information extraction methods [87, 9, 55, 58], to tackle healthcare text data which have been used in health informatics, precision medicine, and clinical research [71, 93].

Implementing such extraction systems in practice remains challenging, as many systems rely on large amounts of annotated textual data to perform well. However, annotating healthcare text is a largely manual and time-consuming process that requires training and medical knowledge. Combined with privacy considerations that limit sharing of corpora, it can be difficult to obtain sufficient amounts of annotated data across many clinical domains. While deep learning has been shown to be extremely powerful in NLP, it can underperform in biomedical applications due to smaller training sets. Therefore, it is of considerable practical importance to develop methods in biomedical NLP that perform well with small amounts of labeled data.

We focus on pathology reports which contain clinical data that are particularly vital for cancer treatment and risk assessment. An estimated 1.8 million Americans will be diagnosed with cancer in 2020. [74] In nearly all cases, diagnosis is made via tissue analysis, described in detail in a pathology report, which is stored in most electronic medical record systems as unstructured free text. Without manual data abstraction, these important details are unavailable for scalable and algorithmic approaches for case identification, risk stratification, prognostication, treatment selection, clinical trial screening, and surveillance. [71, 93] Moreover, access to these data in structured formats can drive algorithmic personalized treatment strategies based on pathologic information. For nearly 50 years investigators have worked to develop natural language processing (NLP) algorithms to extract these details from pathology reports. [45, 9] However, only a limited number of categorical data elements are typically extracted, model outputs often lack reliable uncertainty estimates and furthermore these methods typically require a large labeled dataset, limiting the clinical applicability of these systems, only 10% of which have been reported to be in real-world use. [9]

We detail the results of our sample efficiency and uncertainty estimation experiments using state-of-the-art methods as well as our techniques for building a sample-efficient information extraction system deployed across various cancer types for pathology across UCSF.

More specifically, in chapter 2, we investigate the amount of annotated data required to adequately train state-of-the-art machine learning methods for pathology report information extraction methods. We also investigate the uncertainty estimates of each method, which is of practical importance in the clinical setting. In chapter 3, we propose a novel annotation method which gives location-based information from pathology reports, in addition to document-level annotations, and an algorithm called supervised line attention (SLA) for leveraging the additional location-based data. In chapter 4, we propose transfer learning and text-based similarity algorithms based off of SLA in cases where information is shared across cancers and where the number of possible labels is large.

In addition to information extraction from electronic health records, we also focus on the reinforcement learning setting. Specifically we focus on offline evaluation, a subarea of reinforcement learning which revolves around estimating the value of policies using a static dataset.

1.2 Adaptively combining estimates for offline evaluation of reinforcement learning agents

Reinforcement learning is a subfield of machine learning where the goal is to train an agent to behave well in a given environment. Unlike supervised learning where the goal is to predict a label given a set of features, reinforcement learning agents are trained via an interactive feedback loop with the environment where the agent chooses actions given states and rewards provided from the environment. Reinforcement learning has been studied extensively for decades, and in recent years deep learning applied to the reinforcement learning setting has lead to promising results across a variety of data problems [84, 49, 57, 75].

One aspect of reinforcement learning that limits its adoption is the fact that learning is done online. Agents iteratively collect more data by interacting with the environment, while improving its behavior or policy to collect optimal rewards [82]. In many instances, it is assumed that a policy initialized randomly is deployed to the environment to start the learning process. This kind of online interaction is not always feasible because data collection can be costly or dangerous if the agent is not properly trained. Examples of data problems where online training is especially an issue include patient care, autonomous driving, and robotics [53, 43, 76].

Offline reinforcement learning is a subarea of reinforcement learning where a policy is learned on a static dataset in a data-driven fashion [50]. Unlike online reinforcement learning, data collection via repeated interactions with an environment is not possible. All learning is done using historical data, and the goal is to construct a policy that maximizes rewards if it were run on the actual environment. Unfortunately, offline reinforcement learning presents a challenge that is not present in the online setting. Significant distributional shift may exist between the policy or policies that generated the offline dataset (i.e. the behavior policy) and the target policy. This problem can negatively affect an agent's behavior if the agent learns to select out-of-distribution actions whose values are overestimated [24]. To address this issue, learned policies are generally trained in a way that is "close" to the behavior policy [24, 32, 89].

In chapter 5, we focus on a sub-goal of evaluating reinforcement learning agents within the offline setting, which is known as offline evaluation [67]. Offline evaluation methods strive towards accurately estimating the value of reinforcement learning agents or policies without online data collection. Evaluation is important because it is needed for model selection and hyperparameter tuning which can avoid costly or dangerous interactions with the true environment. Unfortunately, similar to training offline reinforcement learning agents, training offline evaluation models is challenging because validating and tuning the evaluation models themselves without further interactions with the environment is an open question. However, improving offline evaluation is essential for safe practice of offline reinforcement learning and the adoption of reinforcement learning as a whole.

In this work, we develop an adaptive model weighting mechanism and experiment with two state-of-the-art offline evaluation methods, fitted q-evaluation (FQE) and model-based offline evaluation to improve conditional value estimation of agents using static, logged data only. Inspired by work in weighted online learning [11, 5], where past predictability is used to weight online predictors, we use model stability as a signal to weight model estimates adaptively in offline evaluation for reinforcemetn learning. We propose two ways methods, soft stability weighting and soft uncertainty weighting via partial rollouts, which leverage stability stemming from ensembling neural networks. The principle underlying both methods is that stability is a positive signal for the precision of a certain model in the local state space, which cannot be deduced accurately in the offline setting. We compare the offline evaluation methods on simulated environments with varying behavior and evaluation policies against a number of baseline methods, and perform stability checks to test the robustness of such methods.

Chapter 2

Information Extraction from Pathology Reports with Uncertainty Estimation in Limited Data Environments

In this chapter, we evaluate the performance and uncertainty estimates of machine learning based parsers on varying data sizes using pathology reports from UCSF. We find that simpler machine learning based methods can accurately parse reports using small datasets and calibration methods are often required to obtain more precise uncertainty estimates.

Automated information extraction from pathology reports has traditionally been approached using rule-based methods [58, 59, 27, 60]. However, designing rules is labor intensive and requires deep involvement of clinical experts. The complexity and conflicts between rules grow rapidly as the number of rules increases, and as the underlying documents shift, rules quickly become ineffective. [21] NLP has been applied to pathology report information extraction with promising results, using both classic NLP (boosting over a bag-of-ngrams representation of the document) and deep learning approaches (convolutional, recurrent, and hierarchical attention networks).[92, 25] While most work focuses on classification tasks involving fields with a small number of labels (such as histology or margin status), Li and Martinez (2010) investigate categorical fields as well as numeric fields such as the tumor size and the number of lymph nodes examined. [51] Furthermore, many other information extraction tasks and methods have been applied to pathology reports, such as Coden et al [14] which creates a knowledge representation model to represent cancer disease characteristics; Si et al [73] which uses a frame-based representation to extract information from clinical narratives focusing on cancer diagnosis, cancer therapeutic procedure, and tumor description; Xu et al [91] which considers attribute detection as a sequence labeling problem; and Oliwa et al [62] uses NLP to classify gastrointestinal pathology reports into internal and external reports and uses Named Entity Recognition to label accession number, location, date, and sub-labels.

Despite these developments, there has been comparatively little effort in understanding two additional important criteria that are the basis for reproducibility and real-world use. The first is evaluating performance as a function of training data size, which informs practitioners about how much data they may need to deploy similar systems. Creating an annotated corpus is costly and time consuming, and accurate assessment of necessary sample size can aid deployment. [16, 70, 61, 77, 22] Second, accurate uncertainty estimates for the predicted results are critical for clinical deployment, as different uses have varying acceptability thresholds. Having accurate uncertainty estimates means that for all cases where the model score outputs a probability p, it is correct p percent of the time. An example of a model with inaccurate uncertainty estimates would be one that gives a predicted probability of correctness of 90% on all examples, but is actually only correct 10% of the time. Accurate uncertainty estimates are important for deployment, as lower certainty may be acceptable if the results are used for initial screening with manual verification to follow, but higher certainty is required for a clinical decision support system. Resources can be directed to verification for cases of high uncertainty, supplanting the need for full manual abstraction. The source code for this project will be made available under an open source license to facilitate adoption of NLP tools in cancer pathology.

OBJECTIVE

Our objective was to investigate two practical issues that arise when deploying machine learning-based information extraction systems to pathology reports, using prostate cancer as a test case. First, we evaluate the performance of models as a function of dataset size for tasks that involve categorical values, such as histologic grade or presence of lymphovascular invasion, as well as numeric values, such tumor size Second, we describe an approach to model calibration and calculation of uncertainty estimates for each prediction and assessing the quality of the model's uncertainty estimates. We address these gaps in the literature to guide practitioners as they implement these systems in real-world settings.

MATERIALS AND METHODS

Data Sources

We used a corpus of 3,232 free text pathology reports for patients that had undergone radical prostatectomy for prostate cancer at the University of California, San Francisco (UCSF) from 2001 - 2018, which were extracted from UCSF's electronic health record (Epic Systems, Verona, WI). For each document, annotations for 17 pathologic features, such as Gleason scores, margin status, extracapsular extension, and seminal vesicle invasion were extracted (Table 2.1) in the Urologic Outcomes Database, which is a prospective database that contains clinical and demographic information about patients treated for urologic cancer. Since 2001, data have been manually abstracted by trained abstractors under an institutional review board (IRB) approved protocol. This study was separately approved by the IRB.

The full corpus was divided into four parts, 64% training, 20% validation, 10% test, and 10% true test. We looked at the true test only while compiling results. In order to handle our diverse set of fields, we used two separate information extraction methods. For categorical fields, we used a document-based classification method which has been previously applied to information extraction from pathology reports. [92, 25] For fields with a large number of possible values (such as numeric quantities), we used a sequence labeling task to extract individual tokens from the document. [36] We applied our methods to the full training dataset as well as randomly selected subsets of 16, 32, 64, 128, and 256 reports. All models are implemented in using scikit-learn and pytorch.

Document Classifier Methods

For categorical data fields, such as the presence of lymphovascular invasion, we treat it as a document classification problem. These fields have between two to six possible classes (Table 2.1). We apply classical methods, such as logistic regression, random forests, support-vector machines (SVM), and adaptive boosting (AdaBoost) on bag-of-n-gram features, as well as deep learning methods, such as convolutional neural networks (CNNs), and long short-term memory networks (LSTMs).

Token Extractor Methods

Many critical clinical data elements, such as tumor volume, are not suited for classification because they are not categorical in nature. In order to broaden the variety of data fields extracted from the reports, we employ an additional approach which we refer to as *token extractor methods*. These methods are well-suited to extract numerical quantities from a document (such as the estimated tumor volume or the patient's medical record number, Table 2.1). For these fields, we take each token's surrounding context of k words represented as a bag of n-grams as the primary features. We additionally append the token type encoded as a vector to the bag of n-grams context vector. The token type vector specifies whether a particular token is an ordinary word, a numeric value, or a hybrid of the two. These features are used to predict whether or not the token is the token we aim to extract using logistic regression, AdaBoost, or random forest methods. Unlike the document classifier methods, we excluded SVMs and deep learning methods for the token classifier due to the impractical computational requirements for our compute resources. Because our labeled data did not contain the location information of the token of interest within the document, we labeled all tokens that matched our label as a positive example at the time of training. At test time for each token we compute the score under the model that this token should be extracted and then choose the token with the largest score as our final prediction. This token extraction method is applied to the following fields: pathologic T, N, and M stage, prostate weight, and tumor volume. For additional details regarding the pathologic stage field, we refer the reader to the supplementary material. We would like to give a comparison with a related but slightly different information extraction task of Named Entity Recognition (NER), which classifies named entities in text into categories. Like token extraction, this too is a sequence labeling task. In NER, this involves labeling each token into a predefined category and in our case, for a given field, we label each token with a 0 or 1 as to whether or not it is the desired token for this field and document. As a clarifying example for the distinction between the tasks, an NER system with procedure as a predefined category would label all mentions of procedures in a pathology report as the procedures class. However, this is not what we want, as pathologists will often discuss multiple procedures in a report, but we are interested in only the specific procedure that resected the tumor.

Dataset Size and Performance

We investigate the performance over varying data-regimes, since for informaticists who wish to build a machine learning parser on their data, a critical question is the quantity of data points needed for adequate performance and which methods are most likely to perform well. We fixed the training set size to 16, 32, 64, 128, and 256 reports, which were randomly drawn from the full training set and averaged the results over 5 random draws.

Evaluation Metrics

For each field we report the weighted F1 score of the classifier, which is the weighted sum of the F1 scores for each class in the field, where the term for each class is weighted by the portion of true instances of the class. In the Supplementary Tables 1-4, we report the micro F1 and macro F1 to better compare to existing literature. For token extractor models, we compute the accuracy of whether the token extracted from the report was correct.

Hyperparameter Tuning

To tune hyperparameters for the classification models on the full data, we used random search with a validation set to tune each method. For each model, we randomly select 20 model-specific hyperparameter configurations, train the model on the training set, and obtain weighted F1 scores on the validation set. The model with the hyperparameter configuration with the highest score is used to obtain results on the test set. To tune hyperparameters for the classification models in the low data regimes (training on $\leq$ 256 reports), we used random search across 20 configurations of hyperparameters but with 4-fold cross validation to calculate weighted F1 scores. For extractor models, we used random search with 20 hyperparameter configurations and *4*-fold cross-validation for both the full data and low data regimes.

We chose 4-fold cross validation as it provided a good balance between performance and computational cost in preliminary experiments. For more details regarding hyperparameter ranges for different models, we refer the reader to the supplementary materials.

Calibration of Systems

To support multiple use cases for the outputs of our model, it is desirable to estimate the model's uncertainty reflecting the true probability of correctness for each predicted value. For example, values that have a low probability of being correct can be flagged for manual verification, or results can be limited to only those with a high probability of being correct. More rigorously, for a model f and data distribution X ideally we would like a function P^* such that

$$P_x\left(f\left(x\right)=y|P^*\left(x\right)=p\right)=p \text{ for all } p\in\left[0,1\right].$$

One common definition of the discrepancy between the model's predicted probability of correctness and its true probability of correctness is given by the expected calibration error which is the expected difference between the models confidence and its true probability of being correct. [95]

$$E_x\left[\left|P\left(f\left(x\right)=Y|\widehat{P}\left(x\right)=p\right)-p\right|\right],$$

where $f\left(x\right)$ is the model's prediction for a datapoint point x, Y is the true value, and $\hat{P}$ (x) is the model's predicted probability of being correct for point x. However, this is typically not able to be measured in practice, for example if $\widehat{P}\ \left(x\right)$ takes on a continuous set of values, so instead $\widehat{P}\ \left(x\right)$ is discretized into bins and the Expected Calibration Error (ECE) [15] is defined as follows:

$$ECE = \sum_{m=1}^{M} |B_m|/n|acc\left(B_m\right) - conf\left(B_m\right)|,$$

where B_m is the mth bin, $acc\left(B_m\right)$ is the average accuracy of the model in bin m, and $conf\left(B_m\right)$ is the average value of $\hat{P}$ (x) of the model in bin m.

To improve the calibration of our system we apply isotonic regression. [95] In the binary case it takes the confidence of the models output of the positive class and fits a monotonic function where the x-axis represents the model's confidence score and the y-axis represents whether or not the model was correct. In the multivariate case, the calibration method attempts to calibrate the probability estimate of each class. It does this by first calibrating the probability of each class in a one-vs-all setting, then after fitting, estimating the probabilities by normalizing the one-vs-all probability for each class.

Error Analysis

To understand the potential failure modes of our models, for each field we manually analyzed 10 errors randomly chosen in our test set split of the best models in Table 2.2 by comparing the model output and annotated label with the text of the report to check the source of the error. If there were fewer than *10* errors for a field, we analyzed all the model's errors

If the error was a result of an incorrect label in our original data set, it was named as an annotation error. Model errors occurred when the model extracted the incorrect value for a certain field. Next, an error was classified as a report anomaly if there was something wrong with the raw text of the report, such as if the sentences of a report were repeated many times in the text or there was internal inconsistency in the report. Lastly, the evaluation error means that the extracted value was correct but the evaluation method incorrectly penalized the model such as if the correct extracted token was 2 for volume of tumor and the model extracted 2cm.

RESULTS

Document Classifier Performance

We calculated the weighted F1-score for each data field using the true test set (Table 2.2). When working with the full training corpus (n= 2,066), convolutional networks perform the best (mean weighted F1 0.972 across all 12 clinical data elements). However, we see that the best non-deep-learning method is not far behind with AdaBoost having a weighted F1 score of 0.965.

Token Extractor Performance

For token extraction we measure the accuracy of extracting the correct token from each document (Table 2.2). In greater detail, we choose the most probable token over all tokens in the document and compare this to the ground truth. We observe that random forests perform the best out of all the methods with a mean accuracy of 0.883 across 5 fields.

Performance as a Function of Dataset Size

For the classification fields, the classical machine learning methods (logistic regression, SVM, AdaBoost, and random forests) clearly outperform the deep learning methods on average, likely due to the small amount of training data available. The results also show that 128 reports are needed for the best methods to achieve a 0.90 weighted F1 on average across all classification fields. For the token extractor fields, the results seem to plateau at 64 reports. Our experiments show that a training set size in the thousands is not always needed to adequately extract structured data from these pathology reports, an important observation for practitioners weighing the cost of creating an annotated dataset.

Effect of Calibration

We apply calibration to two of our models. For the classification model, we apply isotonic calibration to boosting and for the extractor model we apply isotonic regression to the random forest model. [17] For the extractor case, we treat the probability of the token with the highest probability as the confidence score of the model. We fit our isotonic regression calibration methods on the development test set and evaluate the expected calibration error on the test set, binning our uncertainty estimates $\hat{P}$ (x) into bins of width .1. (Table 4.2). Additional experiments investigating the expected calibration error (ECE) as a function of the bin size, which we include in supplementary Figures 1 and 2, show that while the average expected calibration error increased, the difference in the average expected calibration error between the smallest bin size (4) and the largest (64) was less than .02 for both classification and extraction tasks. [15]

We find that for most classifications fields, the model had expected calibration scores less than .1 and that isotonic regression generally improves upon this. Since for each class the one-vs-all probabilities are calibrated, the calibrated model's predictions may differ from the original model if it is not a binary classification problem, so in addition to the expected calibration error of the model, we list the weighted F1 score of the calibrated model. Conversely, extractor models are not well calibrated out of the box in general, but surprisingly, by only using the probability of the token with greatest probability, performing isotonic regression on this single value is enough to achieve sub .05 expected calibration errors.

We also examined when the model was most overconfident, where we look for examples with high estimated probabilities of being correct, but which were nevertheless wrong. We found the most overconfident example in each field and observed that in 10 of the 15 examples the algorithm was correct and the label was actually incorrect.

Error Analysis

The most common type of evaluation error for the token extractor occurred when the model extracted the right token, but the evaluation method did not correctly score the model (Table 6). For example, if the label for the estimated volume of tumor was 2 (in centimeters) and the model extracted 2cm, the model would be penalized. The most common type of report anomaly occurred when the text in the report was repeated. For example, in one case, each sentence in the report was repeated 3 times. This was an issue in the raw text of the report and was not an aberration in preprocessing. Overall, error analysis shows that the scores given for the models are likely underestimates and the models actually perform better than the raw results show.

For a comprehensive breakdown of errors, we refer the reader to Table 5) in the supplementary material. Because the pathologic stage errors are highly correlated (due to the fact that the different types of stages are encoded in the same token in the text), only the results for the pathologic T-stage are shown.

DISCUSSION

We have investigated several practical issues in the clinical deployment of a machine learning based pathology parsing system and developed a system that can accurately parse prostate reports across a variety of fields and provide reliable per-label uncertainty estimates. Furthermore, we evaluated the number of samples required for adequate performance to guide outside practitioners who are considering using a learning based parsers for their datasets.

The dual classification/extraction approach to our pipeline was developed to accommodate a larger variety of data fields. Yala et al (2017) applied boosting across twenty binary fields on 17,000 labeled breast cancer reports and observed strong performance with F1 scores above .9 for many fields.[6] Gao et al (2018) applied hierarchical attention networks to predict tumor site and grade from pathology reports within the NCI-SEER dataset and noted improvement in micro-F1 (up to 0.2 greater) compared to baselines across two fields (primary site and histologic grade) for a dataset of lung and breast cancer pathology reports. [12] Much of the previous work does not attempt to extract all relevant data fields since they rely primarily on document classification methods which cannot handle continuous values, such as tumor size or prostate weight or perform the related but slightly different task of NER. Although Li and Martinez (2010) attempt to extract data fields based on numeric values using a hierarchical prediction method, the final prediction step relies on a rule based method that has no clear way to be calibrated. [58] Furthermore, while our two methods are not run on the same fields, our algorithm appears to have higher performance in general. Our solution is developing a sequence tagging algorithm that extracts tokens corresponding to the desired values directly, as well as employing classifier methods to extract categorical data fields. Each method is also capable of outputting a score that can be directly calibrated using isotonic regression. One limitation of our extraction methods is that we only consider simple bag-of-n grams based representations and it would be interesting to see how sample efficiency or calibration errors change under a deep learning approach.

Second, we investigated the necessary number of reports needed for accurate classification for our pathology reports by varying the size of the training set of reports from 16 to 256 across both classification and extraction. While others have performed sample efficiency analysis of NLP algorithms across many tasks [4, 29, 72] to our knowledge, this has not been investigated for the important application of clinical information extraction from pathology reports, with the exception of Yala et al. who plot dataset size vs performance over only one method (boosting) and over fields that only take two values. [13] Overall, we found that only 128 labeled reports were needed for the best methods for classification and only 64 for the token extractor, a small number compared to the dataset sizes used in prior work. It is important for practitioners who have a smaller dataset to understand approximately how much performance to expect from a machine learning based approach as it can be expensive and time-consuming to create a large corpus of annotated documents. We hope this encourages more groups to explore these approaches, as large datasets may not always be required. Our study is limited by focusing on a single cancer from one institution, and further work can assess generalizability to other cancers and sites. Of note, there was heterogeneity in report structure and style over 20 years.

Another important observation is that the classical statistical learning methods outperformed deep learning methods by a large margin when fewer than 256 data points were available, while deep learning only slightly outperformed logistic regression when using all 2,066 reports in the training set. This suggests deep learning only adds marginal value and the complexity of the problem, at least for the reports we worked with, is more suited to classical methods.

Finally, we investigated the reliability of uncertainty estimates of the model, which to the authors' knowledge, has not been investigated in other pathology information extraction work. Knowing which reports are likely to be incorrect can decrease the time needed to manually verify extracted data and filter uncertain predictions for tasks like clinical research with small populations, where each predicted value may have a large impact on conclusionsThrough our calibration work, we observed that the classification model was typically well calibrated without any modification, whereas our token extraction algorithm was not. However, by just using the probability of the selected token, isotonic regression was a very effective calibration solution. We furthermore investigated when the model is most likely to be overconfident and found that two-thirds of these errors were due to incorrect annotation labels, not incorrect algorithm outputs.

CONCLUSION

Creating annotated datasets and reliable systems are serious practical concerns when developing and deploying biomedical information extraction systems due to the high cost of creating annotations and the impact of errors on patients outcomes. We show when applying machine learning to pathology parsing, accurate results can be obtained using relatively small annotated datasets and calibration methods can improve the reliability of per-label uncertainty estimates.

TABLES AND FIGURES

Table 2.1. Data elements extracted from pathology reports

Document Classifier Algorithm Fields	
Gleason Grade Primary, secondary, tertiary	Histologic grading of tumor aggressiveness based on the Gleason grading system. Each specimen is assigned a primary, secondary, and occasionally a tertiary score, each of which are whole numbers from 1-5
Tumor histologic type	Primary histologic type, such as acinar adenocarcinoma, ductal adenocarcinoma, small cell neuroendocrine carcinoma
Cribriform pattern	Whether the cells exhibit a cribriform growth pattern (Gleason 4 only)
Treatment effect	Indicator whether there is evidence of a prior treatment, such as hormone treatment or radiation therapy
Margin status for tumor	To evaluate surgical margins, the entire prostate surface is inked after removal. The surgical margins are designated as “negative” if the tumor is not present at the inked margin and “positive” if tumor is present at the inked margin.
Margin status for benign glands	The benign margins are designated as “positive” if there are benign prostate glands present at the inked margin and “negative” otherwise
Perineural Invasion	Whether cancer cells were seen surrounding or tracking along a nerve fiber within the prostate
Seminal vesicle invasion	Invasion of tumor into the seminal vesicle
Extraprostatic extension	Presence of tumor beyond the prostatic capsule
Lymph node status	Whether tumor was identified in lymph nodes
Token Extractor Algorithm Fields	
Pathologic Stage Classification T (primary tumor) N (regional lymph nodes) M (distant metastasis)	Based on American Joint Committee on Cancer TNM staging system for prostate cancer. Based on the edition used in each report (5^{th} - 8^{th} edition)

continued on next page

continued from previous page

Tumor volume	Amount of tumor identified in prostate specimen (cubic centimeters)
Prostate weight	Overall weight of the prostate (grams)

Table 2.2. Weighted F1 scores for classification fields and mean accuracy for token extractor fields on full training data sample (n = 2,066)

Data Element	**Logistic regression**	**Adaboost classifier**	**Random Forest**	**SVM**	**CNN**	**LSTM**	**Majority Class Accuracy**
Gleason Grade - Primary	0.978	0.971	0.941	0.932	0.981	0.628	0.709
Gleason Grade - Secondary	0.958	0.943	0.913	0.912	0.968	0.576	0.467
Gleason Grade - Tertiary	0.923	0.930	0.844	0.886	0.930	0.741	0.901
Tumor histology	0.989	0.995	0.995	0.993	0.995	0.994	0.991
Cribriform pattern	0.963	0.981	0.963	0.968	0.987	0.966	0.997
Treatment effect	0.981	0.979	0.981	0.981	0.981	0.973	0.985
Tumor margin status	0.941	0.953	0.888	0.918	0.950	0.630	0.799
Benign margin status	0.977	0.975	0.972	0.981	0.978	0.967	0.997
Perineural invasion	0.944	0.978	0.938	0.929	0.972	0.613	0.771
Seminal vesicle invasion	0.943	0.974	0.940	0.965	0.976	0.784	0.904
Extraprostatic extension	0.954	0.953	0.882	0.939	0.961	0.778	0.712
Lymph node status	0.983	0.952	0.983	0.973	0.986	0.824	0.570
Mean weighted F1	0.961	0.965	0.937	0.948	0.972	0.790	0.817
T Stage	0.951	0.954	0.948	-	-	-	-
N Stage	0.954	0.954	0.948	-	-	-	-
M Stage	0.972	0.969	0.969	-	-	-	-
Estimate Tumor Volume	0.605	0.765	0.873	-	-	-	-
Prostate Weight	0.846	0.855	0.914	-	-	-	-
Mean Accuracy for token extractor models	0.866	0.899	0.930	-	-	-	-

LSTM: Long Short-Term Memory Neural Network
CNN: Convolutional Neural Network
SVM: Support Vector Machine

Table 2.3. Mean weighted F1 score ± standard deviation for classification models for classification models and mean accuracy ± standard deviation for token extractor models on increasing numbers of reports (n) after 5 trials

Model	**n = 16**	**n = 32**	**n = 64**	**n = 128**	**n = 256**
Classification Models (mean weighted F1 score across all classification fields ± SD)					
Logistic	0.781 ± 0.175	0.846 ± 0.117	0.875 ± 0.090	0.911 ± 0.059	0.934 ± 0.041
AdaBoost	0.829 ± 0.140	0.878 ± 0.100	0.907 ± 0.066	0.928 ± 0.049	0.945 ± 0.034
Random Forest	0.795 ± 0.169	0.835 ± 0.128	0.867 ± 0.101	0.882 ± 0.088	0.901 ± 0.070
SVM	0.738 ± 0.214	0.763 ± 0.209	0.786 ± 0.194	0.842 ± 0.112	0.860 ± 0.140
CNN	0.720 ± 0.225	0.790 ± 0.163	0.851 ± 0.122	0.893 ± 0.086	0.935 ± 0.055
LSTM	0.688 ± 0.205	0.729 ± 0.187	0.743 ± 0.203	0.739 ± 0.214	0.739 ± 0.212
Token Extractor Models (mean accuracy across all token extractor fields ± SD)					
Logistic	0.844 ± 0.085	0.897 ± 0.079	0.892 ± 0.096	0.902 ± 0.087	0.896 ± 0.092
Adaptive Boost	0.877 ± 0.097	0.892 ± 0.080	0.890 ± 0.084	0.896 ± 0.082	0.890 ± 0.092
Random forest	0.897 ± 0.180	0.898 ± 0.064	0.915 ± 0.054	0.920 ± 0.041	0.924 ± 0.038

LSTM: Long Short-Term Memory Neural Network
CNN: Convolutional Neural Network
SVM: Support Vector Machine

Table 2.4: Upper: Classifier accuracy and expected calibration error for boosting before and after isotonic calibration. Lower: Expected Calibration Error for Random Forest Model before and after isotonic calibration.

Classification Calibration				
Data Element	**Weighted-F1**	**ECE**	**Isotonic Weighted-F1**	**Isotonic ECE**
Gleason Grade - Primary	0.95	0.03	0.93	0.03
Gleason Grade - Secondary	0.94	0.08	0.92	0.14
Gleason Grade - Tertiary	0.91	0.05	0.91	0.03
Tumor histology	0.99	0.009	0.99	0.007
Cribriform pattern	0.995	0.007	0.995	0.017
Treatment effect	0.99	0.007	0.99	0.003
Tumor margin status	0.96	0.15	0.94	0.013
Benign margin status	0.994	0.007	0.995	0.019
Perineural invasion	0.95	0.26	0.96	0.02
Seminal vesicle invasion	0.987	0.16	0.97	0.02
Extraprostatic extension	0.96	0.12	0.96	0.01
Lymph node status	0.96	0.04	0.98	0.01

Extractor Calibration		
Data Element	**ECE**	**Isotonic ECE**
T Stage	0.155	0.016
N Stage	0.144	0.013
M Stage	0.007	0.005
Estimated Volume of Tumor	0.221	0.021
Prostate Weight	0.278	0.033

Chapter 3

Supervised line attention for tumor attribute classification from pathology reports

In chapter 2, we found that limited annotated data is a major obstacle for building accurate information extraction systems for pathology reports. In this chapter, we attempt to address this challenge by developing a more sample efficient annotation and information extraction technique. Unlike previous works, we leverage location-based information which we call enriched annotations from pathology reports and a hierarchical algorithm for information extraction leveraging enriched annotations which we name supervised line attention. We compare supervised line attention with the state-of-the-art on 250 colon cancer reports and 250 kidney cancer reports at the University of California, San Francisco. Our proposed method is able to achieve performance on par with previous state of the art techniques using only half the amount of annotated data on average.

BACKGROUND

The abundance of textual data in the clinical domain has led to increased interest in developing biomedical information extraction systems. These systems aim to automatically extract pre-specified data elements from medical documents, such as physician notes, radiology reports, and pathology reports, and store them in databases. Converting the originally free-text data into a structured form makes them easily available to clinical practitioners or researchers.

For categorical attributes, the information extraction task can be viewed as an instance of document classification that classifies the tumor attribute based on document contents. For a given attribute, the value is one of a fixed set of options selected based on information in the document. As an illustration, the set of values for the attribute "presence of lymphovascular invasion" could consist of the values "present", "absent", and "not reported". Both classical and deep learning classification methods have been applied to this task in the prior work discussed below.

There has been success in applying classical machine learning techniques to classifying attributes of tumors from pathology reports. Yala et al. classified over 20 binary attributes from breast cancer pathology reports using boosting over n-gram features. [92] Jouhet et al investigated applications of Support Vector Machines (SVMs) and Naive Bayes classifiers to the task of predicting International Classification of Diseases for Oncology (ICD-O-3) from cancer pathology reports. [35]. More recently, there has been success in applying deep learning techniques to pathology report classification. Qiu et al. applied convolutional neural networks (CNNs) to predicting ICD-O-3 from breast and lung cancer pathology reports. [69] Gao et al. applied hierarchical attention networks to predict tumor site and grade from pathology reports within the NCI-SEER dataset and noted improvement in micro-f1 of up to 0.2 compared to baselines across primary site and histologic grade for lung cancer and breast cancer reports. [25]

There has also been work addressing pathology report classification in the absence of a large amount of labeled data. Odisho et al. analyzed performance of machine learning methods for extracting clinical information from prostate pathology reports across various data regimes and found that, while deep learning performed best when trained on the full dataset of 2,066 labeled documents and achieved a mean weighted-F1 score of 0.97 across classification attributes, simpler methods such as logistic regression and adaBoost performed best in smaller data regimes (<256 reports). [60] Additionally, Zhang et al, investigated the problem of unsupervised adaptation across attributes in breast cancer pathology reports. [7] Given a set of attributes with labels and a new attribute without labels but with relevant keywords, they used adversarial adaptation with semi-supervised attention to extract data. We use all of the above methods as baselines for our system to compare against, with the exception of Zhang et al. due to the difference in tasks.

MATERIALS AND METHODS

Data Sources

Our data consists of 250 colon cancer pathology reports and 250 kidney cancer reports from 2002-2019 at the University of California, San Francisco. The data was split into two sets, a set of 186, which we used for training and validation, and a test set of size 64. We list the tumor attributes and their corresponding possible values in Table 3.1. Institutional Review Board approval was obtained for this study.

Our data consists of 250 colon cancer pathology reports and 250 kidney cancer reports from 2002-2019 at the University of California, San Francisco. The data was split into two sets, a set of 186, which we used for training and validation, and a test set of size 64. We list the tumor attributes and their corresponding possible values in Table 3.1. Institutional Review Board approval was obtained for this study.

Data Annotation Methods

Pathology reports consist of free text describing a patient's clinical history and attributes describing the excised specimen, such as surgical procedure, cancer stage, tumor histology, grade, cell differentiation, and presence of invasion to surrounding tissues. More recent pathology reports also contain a synoptic comment section, which is a condensed semi-structured summary of relevant cancer attributes. While many of the most clinically important attributes are reported in this synoptic comment, this is not always the case. All attributes in the College of American Pathology reporting guidelines are annotated for each cancer, but for this paper we restrict our investigation to attributes for which some label appears in at least 90 % of reports [10]. These include tumor site, histologic type, procedure, laterality, tumor grade, and lymphovascular invasion for both cancers. Additionally, we have the cancer specific attributes of laterality and perineural invasion for kidney and colon cancer specifically. We list all attributes, their set of possible values, and the frequency of each value in Table **??**. The Multi-document Annotation Environment (MAE) [78] was used to annotate the documents.

Enriched Annotations

In previous work, annotations consisted of only the label for each attribute in a document. [92, 60, 25] However, in this work, for each attribute of interest the annotator highlighted all occurrences relevant to the label throughout the document, in addition to the label itself. This provides us with the specific location within the text that directly indicates the attribute's label. Each highlight is classified into the corresponding College of American Pathologists (CAP)-derived category. We investigate two types of annotation: the first we refer to as the "reduced annotation set", a minimal set of annotations containing the line of a given attribute value's first occurrence in the synoptic comment, or, if not in the synoptic comment, the line of where that information is referenced elsewhere in the document. The incremental time required to annotate this location is marginal because the annotator does not need to read any more of the document than that required to annotate the first occurrence of the attribute value. In fact, "We investigated the amount of additional time required to create these enriched annotations and found that it took 20 percent longer on average, primarily due to the time it took the annotator to navigate the attribute drop-down menu. This could perhaps be improved through user interface (UI) considerations. In addition to a reduced annotation set, we also investigate performance with all the occurrences relevant to the final classification highlighted, a more laborious annotation scheme. For our results, unless stated otherwise, we are using the reduced annotation set due to its comparable annotation time to labeling the attribute values alone.

Data Preprocessing

For all methods, we replace all words that occur fewer than two times in the training data with a special <UNK> token, and remove commas, backslashes, semi-colons, tildes, periods, and the word "null" from each report in the corpus. For colons, forward slashes, parentheses, plus, and equal signs, we added a space before and after the character. The spaces were artificially added to preserve semantic value important to the task. For instance, colons often appear in the synoptic comment, and so if an n-gram contains a colon, it can indicate that the n-gram contains important information. If multiple labels for an attribute occurred within a report, we concatenate them to form a single composed label. For example, if the report contains both grade 1 and grade 2 as labels for histologic grade, we label the histologic grade of the report as "grade 1 and grade 2" .

Baselines

For all classical baselines, we represent each document as a union of a set of n-grams where n varies from 1 to N, where N is a hyperparameter. For all methods we use random search [8] with 40 trials to tune our hyperparameters according to the 4-fold cross validation error which we found in preliminary experiments to be a good compromise between performance and computational efficiency.

Logistic regression

We use sklearn's [63] logistic regression model with L1 regularization and the liblinear solver. We use balanced class weights to up-weight the penalty on rare classes. We generate 500 points from -6 to 6 logspace for the regularization penalty, and sample 40 points at random.

Support Vector Classifier

We use sklearn's SVC model with balanced class weights. We define our parameter space as 500 points evenly generated from -6 to 6 in log space for the error penalty C of the model; the kernel as linear or rbf; and the parameter of the kernel as either 0.001, 0.01, 0.1, or 1. We then sample 40 points at random from this space.

Random Forest

We use sklearn's random forest classification model with balanced class weights. The parameter space consists of the number of estimators from 25, 50, 100, 200, 400, 600, 800, and 1000; the minimum number of samples for a leaf from 1 to 128 in powers of 2; max depth of a tree from 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100; and whether to bootstrap samples to build trees or not. We sample 40 points randomly from this parameter set.

Boosting

We use the *xgboost* model from chen2016xgboost. The hyperparameters were 500 points from -2 to -0.5 in logspace for the learning rate; a max depth between 3 to 7; a minimum split loss reduction to split a node that is 0, 0.01, 0.05, 0.1, 0.5, or 1; a subsample ratio that is 0.5, 0.75, or 1; and an L2 regularization on the weights that is 0.1, 0.5, 1, 1.5, or 2. We sample 40 points at random from this parameter set.

Hierarchical Attention Network

We implement the hierarchical attention method from Gao et al. This model represents the document as a series of word-vectors. For each sentence in the document it runs a gated recurrent unit (GRU) [12] over the word vectors. It then uses an attention module to create a sentence representation as a sum of the attention-weighted outputs of the GRU. To generate the document representation, a GRU is run over the sentence representations, followed by another attention module is applied to the GRU outputs. The document representation is the attention-weighted sum of the GRU outputs.

For our hyperparameters we use random search across the learning rate, which is either 1e-2, 1e-3, or 1e-4; the width of the hidden layer of the attention module, which is either 50, 100, 150, 200, 250, or 500; the hidden size of the GRU, which is either 50, 100, 150, 200, 250, or 500; and the dropout rate applied to the document representation, which is either 0, 0.2, 0.4, 0.6, or 0.8. We use a batch size of 64 and ADAM [42] as our optimizer.

OUR METHOD: SUPERVISED LINE ATTENTION

In order to take advantage of annotations enriched with location data, we propose a two-stage prediction procedure in which we first predict which lines in the document contain relevant information. We then concatenate the predicted relevant lines and use this string to make the final class prediction using logistic regression.

Finding Relevant Lines

The first stage predicts which lines are relevant to the attribute. We do this by training an xgboost binary classification model that takes a line represented as a bag of n-grams as its input and outputs whether or not the line is relevant to the attribute. The relevance of each line is predicted independently by this initial classifier.

We then take the top-k lines with the highest scores under the model (where k is a hyperparameter). Groups of adjacent lines are conjoined into one line so that sentences which span multiple lines are presented to the model as a single line.

Finally, we represent each line as a set of n-grams vectors and compose a document representation as the weighted sum of each vector representation, which is weighted by the score of that line under the xgboost model. If a line is conjoined, its weight is the maximum of all the xgboost scores for each line in the conjoined line. Mathematically, this is represented as

$$d_r\left(l_1, ..., l_n\right) = \sum_{l_i \epsilon S_k} v\left(l_i\right) m\left(l_i\right),$$

where d_r represents the vector representation of a document d , S_k are the top-k lines with the highest scores under the xgboost model, v is the mapping from a line l_i to its set of n-grams representation, and $m\left(l_i\right)$ is the xgboost score for line l_i .

With this final weighted representation concatenated with a vector of the line scores, we train another xgboost model which acts as the final classifier to predict the final class.

We refer to this method as "supervised line attention" due to its relationship to supervised attention in the deep learning literature which predicts relevant locations and creates a weighted representation of the relevant regions' features. Supervised attention in the deep learning literature has been used to match a neural machine translations attention distribution to match an unsupervised aligner [52] and to match a sequence-to-sequence neural constituency parser's attention mechanism with traditional parsing features [38], for example. Our approach can be viewed as a form of supervised attention for document classification. The principle difference from existing work is that in supervised attention in the deep learning literature the method is trained in an end-to-end fashion with neural networks, whereas we train each module independently with classical methods and our feature representation for sentences are sets of n-grams instead of dense real-valued vectors.

Rule-based line classifier

As a baseline, we also include a line classifier that selects relevant lines by searching for expert-generated keywords and phrases. After the lines are selected, the final representation is generated the same way, with the exception that all lines are given a weight of 1; thus,for all $l_i \ \varepsilon S_k$, $m\left(l_i\right) = 1$.

Oracle Model

In addition to the line attention model, we also evaluate a model that uses the correct relevant lines from the annotator directly as input to the final classifier, which we refer to as the "oracle model" . Using the oracle lines, the final representation is generated the same way as the rule-based line classifier, where all lines are given a weight of 1.

Hyperparameter tuning

Similar to our baselines, we perform random search for 40 iterations and choose the hyperparameters that minimize 4-fold cross-validation error. The hyperparameters for our shallow attention method are an n-gram size for finding relevant lines between 1 and 4; an n-gram size for the second stage of making the final classification between 1 to 4.

For *xgboost*, the hyperparameters were 500 points from -2 to -0.5 in logspace for the learning rate; a max depth between 3 to 7; a minimum split loss reduction to split a node that is 0, 0.01, 0.05, 0.1, 0.5, or 1; a subsample ratio that is 0.5, 0.75, or 1; and an L2 regularization on the weights that is 0.1, 0.5, 1, 1.5, or 2.

For the final classifier, the L1 penalty is chosen from 500 evenly spaced points from -6 to 6 in logspace. Additionally, since the final representation is a weighted representation of the features of the top-k lines under the line classifier model, we have a hyperparameter k which determines how many lines to use, where k is between 1 and 5.

Ablation Experiments

For our ablation experiments, we investigate the relative contribution of each component in our model.

No weighting

Here we investigate if weighting the features in each line by the classifier scores increases performance compared to weighting the features in each line by one.

No joining

Here we investigate how joining affects the results when information spans multiple lines. Instead of conjoining lines that occur adjacent to each other, we leave them as separate lines for our final classifier.

No weighting and no joining

Here we neither weight the features vectors representing each line nor do we join adjacent predicted lines.

Error Analysis

To better understand model performance, we inspect all errors that the supervised line attention model makes for each attribute and cancer domain. In our investigation we find 6 primary types of errors, which we define below:

Attribute Qualification Error occurs when the model correctly extracts the relevant lines, but fails to classify the final label correctly because the label text is negated or qualified by an additional phrase indicating information is not available, such as in the following example: "If we were to classify the tumor, it would be grade 2 but due to the treatment effect it is unclassified."

Rare Phrasing Error occurs when the model correctly predicts the relevant lines, but the relevant lines contain rare or unusual phrasing and the model assigns an incorrect final classification.

Irrelevant Lines Error occurs when the model includes irrelevant lines in its final predictions, which can influence the final classification.

Multi-Label Error occurs when a report contains a conjoined label (such as “grade 1 and grade 2”), but the model only correctly predicts one of the labels.

Annotator Error occurs when the model’s prediction is correct, but on re-review we noted that the annotator’s label was incorrect.

U*nknown error* occurs when the underlying cause of the error is not known. This often occurs when the model correctly extracts out the relevant line but assigns an incorrect final label.

RESULTS

We trained our methods using various training set sizes of 32, 64, 128, and 186 with 4-fold cross validation. We take the average of 10 runs where we reshuffle the data and generate new splits each time and compute 95% confidence intervals for all methods using bootstrap resampling, with the exception of the HAN method due to computational limitations. For our results, unless stated otherwise, we are using the reduced annotation set due to its comparable annotation time to labeling the attribute values alone. As shown in Figure 3.1 and Table 3.4, our shallow attention model frequently improves substantially over existing methods in terms of micro and macro-f1, particularly in the lowest data regimes. For example, for colon cancer we see an absolute improvement of 0.10 micro-f1 and 0.17 macro-f1 over previously existing methods with 32 labeled data points. Furthermore, SLA frequently tends to perform as well or better than state of the art methods with only half the labeled documents. Two exceptions are in kidney cancer micro-f1 scores, where boosting performs .01 better in micro-f1. We see that the rule-based line classifier method tends to do better than existing methods with 64 labeled data points or fewer, but its performance plateaus and XGBoost outperforms it with 128 and 186 labeled data points. Furthermore, we see that the rule-based line classifier consistently performs worse than supervised line attention.

Ablation Results

We plot the results of our ablation experiments in Figure 3.2, using the same setup as our main result where we have training set sizes of 32, 64, 128, and 186 with 4-fold cross validation. Again, we take the average of 10 runs where we reshuffle the data and generate new splits each time and compute 95% confidence intervals for all methods using bootstrap resampling. We see mixed results for joining adjacent predicted lines; it appears to be inconsequential for colon cancer and detrimental for kidney cancer. However, weighting the features by line predictor seems beneficial for the macro-f1 scores. This seems to suggest that weighting helps primarily for rare classes since the macro-f1 score weights the f1 scores of each class equally.

Full Annotations

Here we compare how well reduced annotation compares to the more laborious full annotation setting where we highlight all areas in the document relevant to final classification. We use the same setup as for our main results as wells as our ablation experiments and present our results in Figure 3.3. We can see that the full annotation set leads to a consistent increase in performance. However, it is unclear whether the extra time required to create this full annotation scheme is beneficial overall as it would lead to fewer documents annotated in the same amount of time.

Error Analysis

We provide a compilation of the number of errors across attributes in Table 3.2 and Table 3.3 for colon and kidney cancer, respectively. We see that the most common error is the multi-label error. This is primarily problematic for colon cancer histologic grade, where pathologists will describe a range of grades such as “grade 1-2” and tumor site for colon and kidney cancer as tumors can inhabit multiple sites. This suggests that treating this as a multi-label classification problem instead of naively conjoining multiple labels may reduce many of the errors.

DISCUSSION

We have investigated the efficacy of location-enriched annotations and a corresponding simple and interpretable method, which we call Supervised Line Attention, for extracting data elements from pathology reports across colon and kidney cancers at UCSF. By leveraging location annotations, our two-stage modeling approach can lead to increases of micro-f1 scores up to 0.1 and macro-f1 scores up to 0.17 over state-of-the-art methods and typically reduces the required annotation by 40% to achieve the performance of existing methods.

Our SLA approach with enriched annotations was primarily developed to tackle the problem of achieving accurate performance with minimal labeled data. Previous approaches that attempt to leverage additional data use multi-task learning and transfer learning using information from other cancer domains with complex modeling architectures. For example, [69] investigated using transfer learning with convolutional neural networks to extract data from 942 breast and lung cancer reports, achieving 0.685 and 0.782 micro-f1 scores, respectively. [2] implemented multitask learning with convolutional neural networks to classify tumor attributes in 942 pathology reports for breast and lung cancers, and achieved 0.77, 0.79, and 0.96 micro-f1 scores for tumor site, histologic grade, and laterality, respectively.

An important observation is that our approach is more interpretable than previous machine learning methods, since addition to outputting the probability and predicted value for a certain report, our system outputs the exact lines of the text used to make the classification as well. This enables practitioners to easily check predictions by examining the lines output by the extraction system, and verify the system is working as expected before making clinical decisions. The hierarchical attention approach used by [26] also can output the most pertinent sentences for a classification by using the attention mechanism to hierarchically filter out pieces of text. However, our experiments show that HAN requires a large training size to achieve adequate performance due to the more complex architecture used, and requires significantly more development and computational time to search the hyperparameter space. Additionally, there have been recent concerns regarding the interpretability of attention distributions from neural networks [31].

Our study has a few limitations. Although we observe high performance of our methodology in both colon and kidney cancer reports at UCSF, our investigation was done at a single institution; this may limit the generalizability of our findings to other institutions that use different pathology reporting or data collection systems. Second, within the field of natural language processing, there has been strong empirical evidence showing the benefit of pre-trained contextualized representations for a variety of tasks, both in and out of clinical applications [64, 18, 48, 30]. In preliminary experiments, we investigated the efficacy of using biomedical word vectors [68] as feature representation input to our SLA model, but did not see an improvement in results. However, it would be interesting to investigate the effect that more sophisticated contextualized representations may have on downstream performance, and this may increase performance of SLA even further.

CONCLUSION

Our work has shown that leveraging location-based information in addition to document-level labels can reduce the number of labeled documents to adequately train machine learning based extraction models for pathology reports. We formulate one such method, supervised line attention, to incorporate location-based information for tumor attribute classification. Our experiments show that supervised line attention can achieve accuracies comparable to the state-of-the-art while relying on much less annotated data and allowing for greater interpretability.

Tables

Table 3.1. Extracted attributes and their possible values

Tumor Site	
Colon	Cannot be determined, cecum, colon not otherwise specified, hepatic flexure, ileocecal valve, left descending colon, other, rectosigmoid junction, rectum, right ascending colon, sigmoid colon, splenic flexure, transverse colon, or not reported
Kidney	Upper pole, middle pole, lower pole, other, not specified, or not reported
Histologic Type	
Colon	Adenocarcinoma, adenosquamous carcinoma, carcinoma, type cannot be determined, large cell neuroendocrine carcinoma, medullary carcinoma, micropapillary carcinoma, mucinous adenocarcinoma, neuroendocrine carcinoma poorly differentiated, other histologic type not listed, serrated adenocarcinoma, signet-ring cell carcinoma, small cell neuroendocrine carcinoma, squamous cell carcinoma, undifferentiated carcinoma, or not reported
Kidney	Acquired cystic disease associated renal cell carcinoma, chromophobe renal cell carcinoma, clear cell papillary renal cell carcinoma, clear cell renal cell carcinoma, collecting duct carcinoma, hereditary leiomyomatosis and renal cell carcinoma-associated renal cell carcinoma, mit family translocation renal cell carcinoma, mucinous tubular and spindle renal cell carcinoma, multilocular cystic clear cell renal cell neoplasm of low malignant potential, oncocytoma, other histologic type, papillary renal cell carcinoma, papillary renal cell carcinoma type 1, papillary renal cell carcinoma type 2, renal cell carcinoma unclassified, renal medullary carcinoma, succinate dehydrogenase sdh deficient renal cell carcinoma, t611 renal cell carcinoma, tubulocystic renal cell carcinoma, xp11 translocation renal cell carcinoma, or not reported
Procedure	

continued on next page

continued from previous page

Colon	Abdominoperineal resection, left hemicolectomy, low anterior resection, not specified, other, polypectomy, right hemicolectomy, sigmoidectomy, total abdominal colectomy, transanal disk excision, transverse colectomy, or not reported
Kidney	Total nephrectomy, partial nephrectomy, radical nephrectomy, other, or not reported
Laterality	
Colon	Not applicable to colon cancer
Kidney	Left, right, or not reported
Grade	
Kidney, Colon	Grade 1, 2, 3, 4, not applicable, or not reported
Lymphovascular Invasion	
Kidney, Colon	Present, absent, or not reported
Perineural Invasion	
Colon	Present, absent, or not reported
Kidney	Not applicable for kidney cancer

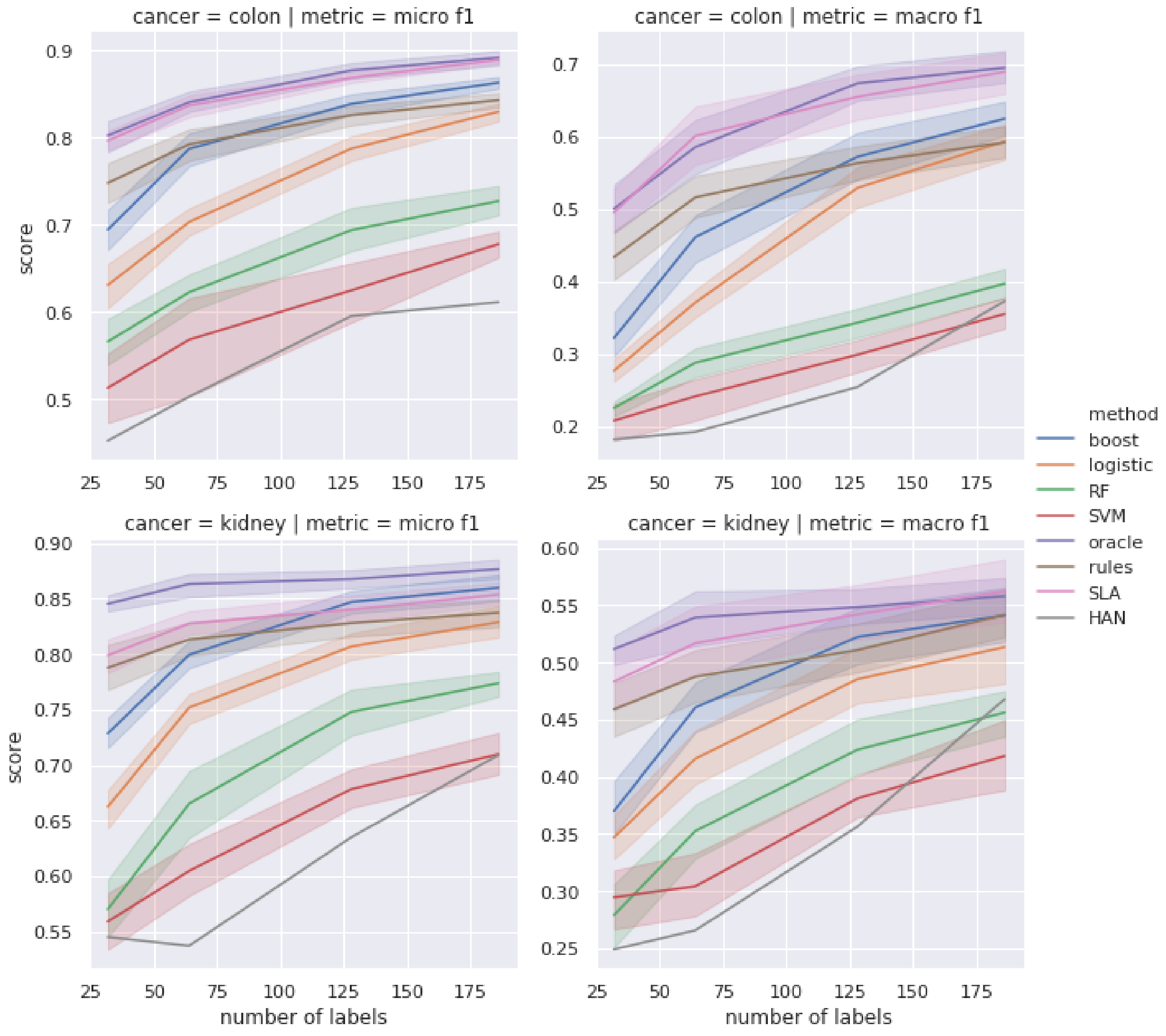
Table 3.2: Error analysis: Colon cancer

Attribute	Histologic Grade	Histologic Type	Perineural invasion	Lymphovascular invasion	Procedure	Tumor Site	Total
Attribute Qualification Error	1	0	0	0	0	0	1
Rare phrasing	0	0	0	**1**	3	0	4
Irrelevant Lines Error	1	0	0	0	5	0	6
Annotator Error Error	3	**1**	**1**	0	5	0	10
Multi-label Error	**6**	0	0	0	0	**6**	**12**
Unknown error	1	0	0	0	**6**	0	7
Total by attribute	12	1	1	1	**19**	6	40

Table 3.3: Error analysis: Kidney cancer

Attribute	Histologic Grade	Histologic Type	Specimen Laterality	Lymphovascular invasion	Procedure	Tumor Site	Total
Attribute Qualification Error	0	0	0	0	0	0	0
Rare phrasing	0	0	0	**1**	**5**	0	6
Irrelevant Lines Error	**1**	0	0	**1**	1	1	4
Annotator Error	**1**	2	0	**1**	1	0	5
Multi-label Error	0	**4**	0	0	2	**6**	**12**
Unknown error	**1**	**4**	0	**1**	1	5	**12**
Total by attribute	3	10	0	4	10	**12**	39

Figure 3.1: Average micro-f1 and macro-f1 performance across attributes of different methods as a function of 32, 64, 128, 186 labeled examples on colon cancer and kidney cancer pathology reports. SLA: supervised line attention; oracle: oracle model that gets access to the true lines as input; rules: line prediction done with a rule-based method and logistic regression to predict the final class; boost: XGBoost; SVM: Support Vector Machine; logistic; logistic regression; RF: Random forest; HAN: Hierarchical attention network. We present the mean result across 10 random shufflings of the data as well as 95% bootstrap confidence intervals. We see that our method SLA outperforms existing methods in almost all cases. Furthermore, we see that predicting relevant lines outperforms our rule-based method to extract relevant lines.



Colon								
	HAN	RF	SVM	Boost	Logistic	Rules	SLA	Oracle
Micro-F1								
32	.45	.57	.51	.69	.63	.75	**.80**	.80
64	.50	.62	.57	.79	.70	.79	**.84**	.84
128	.60	.69	.62	.84	.79	.83	**.87**	.88
186	.61	.73	.68	.86	.83	.84	**.89**	.89
Macro-F1								
32	.18	.22	.21	.32	.28	.43	**.50**	.50
64	.19	.29	.24	.46	.37	.52	**.60**	.59
128	.25	.34	.30	.57	.53	.56	**.66**	.67
186	.37	.40	.35	.62	.59	.59	**.69**	.70
Kidney								
Micro-F1								
32	.54	.57	.56	.73	.66	.79	**.80**	.85
64	.54	.67	.60	.80	.75	.81	**.83**	.86
128	.63	.75	.68	**.85**	.81	.83	.84	.87
186	.71	.77	.71	**.86**	.83	.84	.85	.88
Macro-F1								
32	.25	.28	.29	.37	.35	.46	**.48**	.51
64	.27	.35	.30	.46	.42	.49	**.52**	.54
128	.36	.42	.38	.52	.49	.51	**.54**	.55
186	.47	.46	.42	.54	.51	.54	**.56**	.56

Table 3.4: Average micro-f1 and macro-f1 performance across attributes of different methods as a function of 32, 64, 128, 186 labeled examples on colon and kidney cancer. Highest performing non-oracle method is bolded.

Figure 3.2: Ablation studies for SLA measuring the average micro-f1 and macro-f1 performance across attributes of different methods as a function of 32,64,128,186 labeled examples on colon cancer and kidney cancer pathology reports. We investigate the impact of joining adjacent selected lines prior to featurization as well as the impact of weighting the features by the line classifier scores. We present the mean result across 10 random shufflings of the data with 95% bootstrap confidence intervals. While it appears that joining adjacent predicted

lines leads to mixed or potentially even negative performance over not joining adjacent predicted lines, weighted methods seem to outperform their unweighted alternatives, especially for macro-f1 scores, suggesting that weighting helps in particular for rare classes

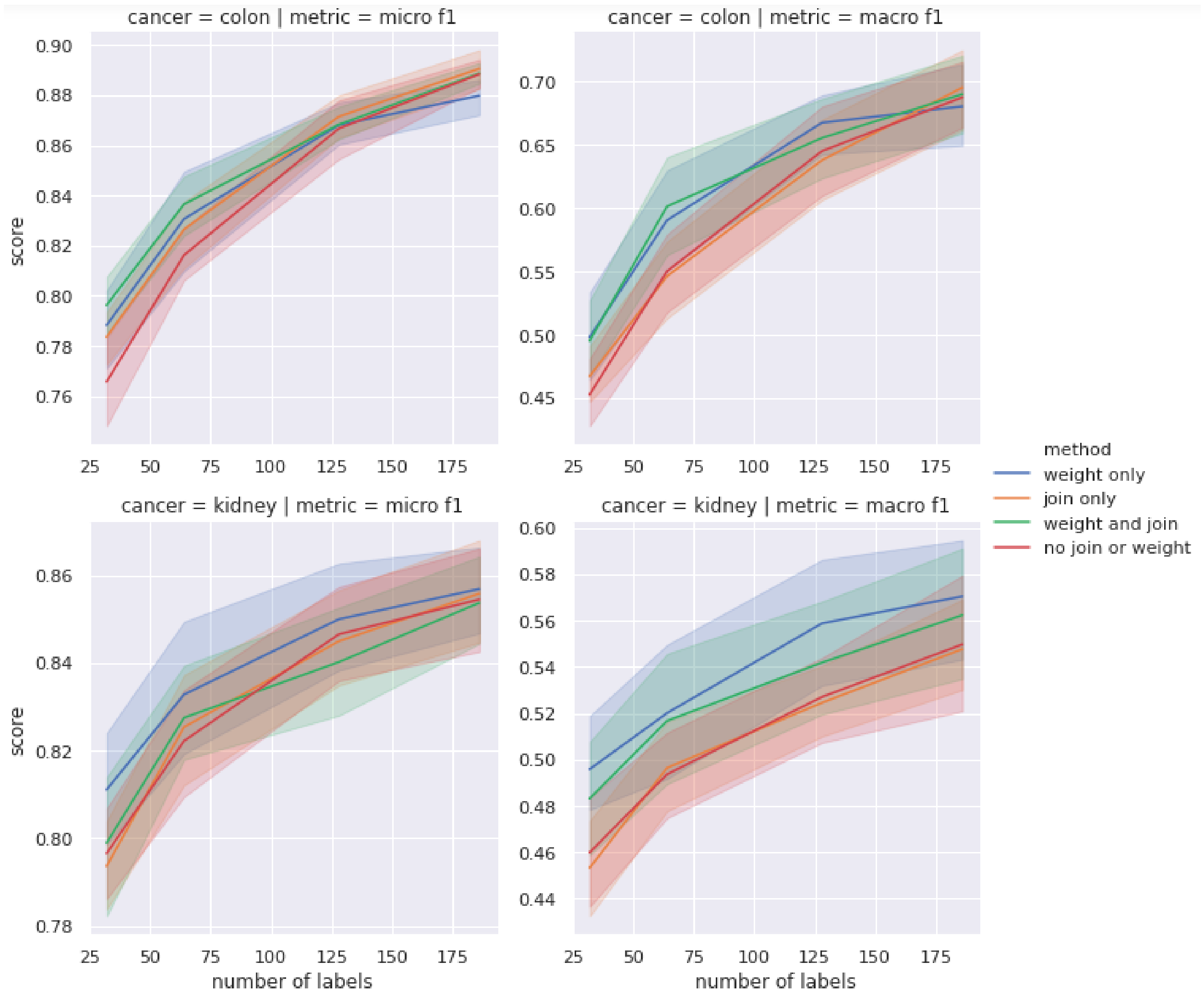
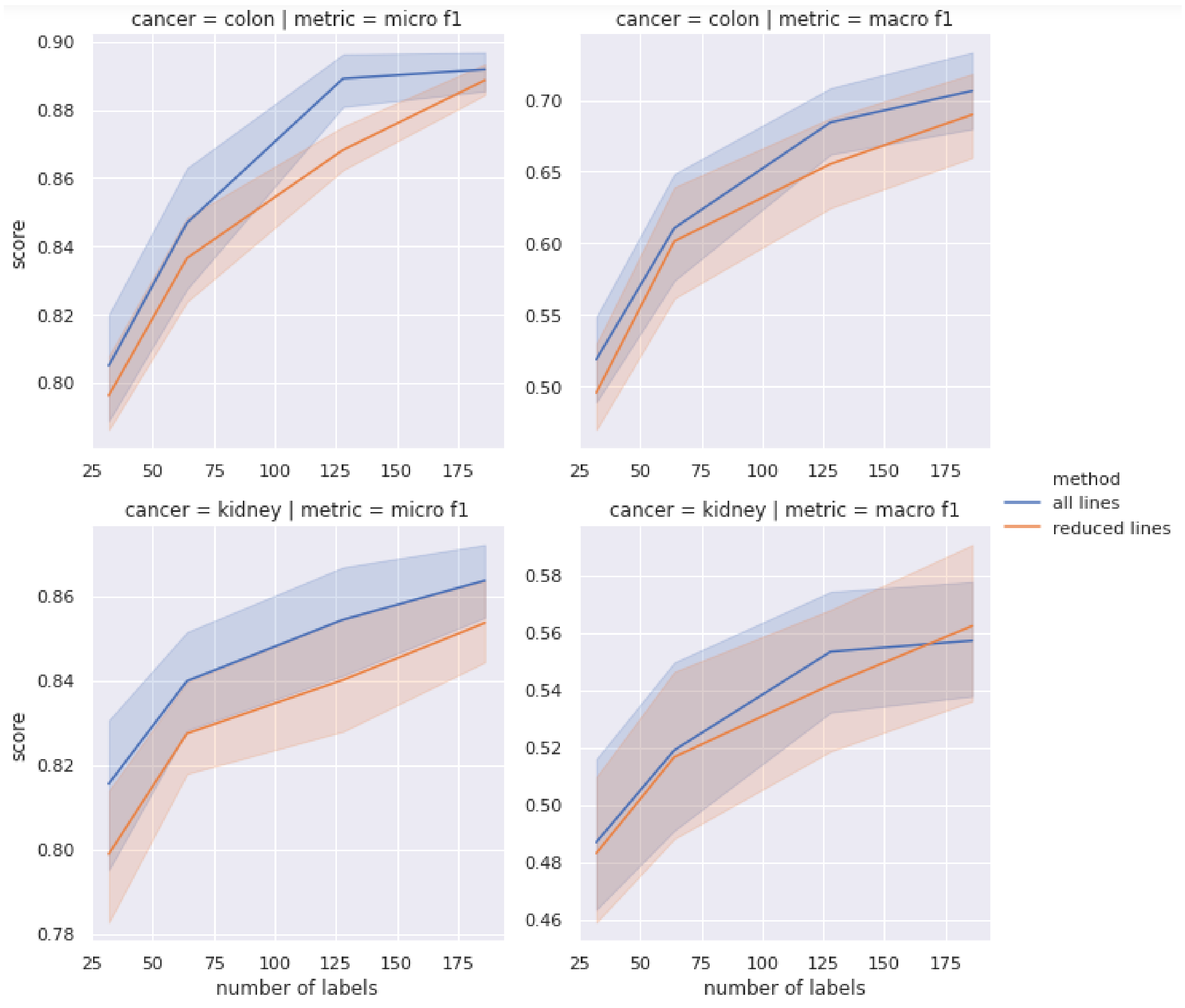


Figure 3.3: Comparing the more laborious scheme of annotating the location information for all relevant lines for a given attribute as compared to the more lightweight annotation method of only annotating the first line in the synoptic comment, if the synoptic comment contains the information, or the first relevant line in the document otherwise. We see that having the additional information yields a consistent, though sometimes small, benefit.



Chapter 4

Improving information extraction from pathology reports using transfer learning and text similarity

In this chapter, we develop methods building upon supervised line attention in two cases. In the first case we incorporate different types of transfer learning when information is shared across cancers. This allows us to leverage annotated pathology reports from different cancer types to expand the training set size for a particular cancer of interest. In the second case, we leverage text similarity techniques to facilitate zero-shot learning which is particularly important in cases when the amount of labels is particularly large compared to the amount of training data. These techniques help us achieve accuracy comparable to the state-of-the-art while requiring much less annotated data. We compare our proposed methods against existing methods in the literature on lung, colon, and kidney cancer reports at the University of California, San Francisco.

BACKGROUND

Despite a long history of approaches to biomedical information extraction which include rules-based methods [59], classical machine learning methods [92, 55], and deep learning methods [69, 26],few works have focused on sample efficient learning. Yala et al [92] carried out a performance analysis of boosting tree extraction models and found that approximately 400 training examples were required to obtain an accuracy of 0.9 for 20 breast cancer attributes, though they only considered tumor attributes that take on present or absent values. In chapter 2, we showed non-deep learning methods largely outperformed deep learning methods with data sizes below 256 for prostate cancer reports.16 In chapter 3, we outlined developed a novel supervised line attention (SLA) approach using more fine-grained, location-based annotations and showed the fully supervised location-based approach outperformed the state-of-the-art methods using training data sizes below 186 for colon and kidney cancer reports. However, these methods still require hundreds of labeled examples.

Transfer learning has been shown as a promising approach to improve medical data extraction performance. Qiu et al. found that intra-class transfer learning on convolutional neural networks provided improvements of up to 0.04 in micro-F1 and macro-F1 scores for lung and breast tumor sites [69]. Alawad et al. show that multi-task convolutional neural networks trained across cancer types achieve up to 0.17 improvement in the macro-F1 score over single cancer registry models [3].

Zero-shot learning is also a promising avenue for achieving better sample efficiency in limited data settings [1, 90]. It is a setting where the model learns to classify test instances with labels not previously seen in the training set. Typically, a zero-shot learning approach learns to make a prediction by using the original features of an instance and auxiliary information of classes, which are related to the feature space. For example, for a document classification task, the features could include the document text, while the class name and description could be used as the class auxiliary information. The learned relational information between auxiliary information and features allows the model to generalize to new classes when auxiliary information is available.

In this work, we extend the existing SLA approach based on enriched annotations using transfer learning and zero-shot learning. For tumor attributes with labels that are shared across colon, kidney, and lung cancers, we develop a cancer-to-cancer transfer learning procedure to leverage cancers with many labeled examples for cancers with few labeled examples. Transfer learning is applicable here, since much of the language is shared when reporting an individual attribute that is shared across different cancers. For tumor attributes with labels that are unique to a specific cancer type, we develop zero-shot string similarity (ZSS) methods to augment our SLA approach. ZSS finds the predicted label by calculating string similarity scores between the label and text. Note that character-based similarities can be calculated for unseen labels as long as the label name is available at prediction time. Since ZSS only requires a string similarity score to make a prediction, ZSS can generalize to labels never seen during training.

MATERIALS AND METHODS

Data Sources

We randomly sampled 250 pathology reports each across colon, lung, and kidney cancers from the University of California, San Francisco from 2002 to 2019. For each cancer type, we sampled and annotated 250 reports to create 10 random cancer-specific, train-validation-test splits. Each split for a cancer consists of the same 250 annotated reports overall, but the individual training, validation, and test sets differ due to randomness. The full train, validation, and test sets consist of 40, 20, and 190 annotated reports, respectively. We chose to place a majority of the data in the test set and limit the number of reports in the training and validation splits, since we are interested in performance in the low data regime (10–40 examples). Each experiment is run separately on each of the 10 splits. We obtain confidence intervals for the evaluation metric scores computed on the test set of each of the 10 splits.

Tumor attributes

Tumor attributes of interest are histologic grade and the presence of lymphovascular invasion for transfer learning on shared labels across cancers and tumor site, histologic type, and the surgical procedure carried out on a patient for the text based similarity method.

Enriched Annotations

Our pathology reports contain annotations for ground-truth labels as well as highlighted text throughout the report relevant to the label as in the previous chapter. These fine-grained annotations provide the lines in the report that determine the value of a tumor attribute. Similar to the previous chapter we use the "reduced annotation set", which consists of the minimal set of annotations containing the line of a given data field's value in the synoptic comment or the first line that contains the relevant information if the report does not contain a synoptic comment. These synoptic comments are typically common in more recent pathology reports and are a brief standardized portion of the text where relevant cancer attributes are reported. As mentioned in the previous chapter, reduced location-specific annotations take 20 percent longer on average than typical annotations.

Before any kind of vectorization of the text, we replace words that occur once in the training data and words that never appear in the training data with a special <UNK> token. We then remove commas, backslashes, semi-colons, tildes, periods, and the word "null" from each report and add spaces around colons, forward slashes, parentheses, plus and equal signs. Additionally, we also concatenate labels if multiple labels are assigned to a report for a specific field. For example, if the report contains both "upper" and "lower" labels for the field tumor site, then we aggregate the labels so the final label is "upper and lower."

SUPERVISED LINE ATTENTION

Our models are based on the SLA framework, outlined in chapter 3. The goal is to predict the lines in the report that contain information on a specific tumor attribute and then use the predicted lines to make the final class prediction for an attribute. There are 2 separate classifiers for the line prediction task and the class prediction task trained using location-based and label annotations. As in previous work, separate XGBoost models are used for the line prediction task and the class prediction task.

Tumor attributes are divided into 2 distinct categories. The first category contains tumor attributes with shared labels across cancers, such as the histologic grade. Most cancers are graded on a numeric or ordinal scale, and while the underlying biology and clinical significance of the grades differ, the labels are similar. The second category contains tumor attributes whose labels are not shared across cancers. An example is the procedure; for each organ system or cancer type, there are a different set of surgical procedures for resecting tumors. The first group is a natural candidate for a transfer learning approach, whereas transfer learning is less applicable for the second group, since the labels are not shared across cancer types. We propose 2 methods to perform extraction depending on whether the labels are shared across cancers.

1) Shared labels

When labels are the same across cancers, knowledge can be transferred from one cancer type to another. For example, for the presence of lymphovascular invasion, identifying the relevant lines in a report is domain-independent because lymphovascular invasion is a relevant attribute for many cancers. Furthermore, identifying the correct label is again domain-independent because the categories (present and not identified) are the same across cancers. Shared knowledge is important because reports from other domains can be used to improve performance through data augmentation.

We create a transfer learning technique to learn data extraction using the shared information across cancer types for the relevant tumor attributes. We build off SLA by training both the line classifier and the final classifier on reports from all domains, which we refer to as hierarchical cancer to cancer transfer learning (HCTC). We apply cancer-to-cancer transfer learning hierarchically at both stages of SLA: predicting the relevant lines in the report and predicting the final classification of a report. As a sensitivity analysis, we report results with ablations to HCTC where we only share information for the line classifier (HCTC-line) or the final classifier (HCTC-final).

2) Unique labels case

Unlike the shared labels case, we opt against a transfer learning approach as the applicability is uncertain due to a different label space for each cancer. Furthermore, some attributes have a large number of labels (there are 32 possible labels for kidney histologic type). It is highly likely that we will encounter labels at test time that were not present in the training data. Typical machine learning models need a sufficient number of examples for each possible label to learn classification tasks and generalize to new data. As seen by the large set of possible labels for our attributes, it is possible to see few or no examples of a class during training. Consequently, a technique that can handle a large set of labels is essential here and in particular, a method capable of zero-shot learning is necessary.

We develop a novel method that enables a more sample efficient method capable of zero-shot learning, referred to as ZSS. At a high level, ZSS first predicts the relevant lines for the label using the line classifier as in Altieri et al [6], then calculates the string similarity score between each possible label and the concatenated text of the top 3 lines output from the line classifier using a subroutine we call the fuzzy jaccard score (Algorithm 1). Finally, we take the label with the highest fuzzy jaccard score as the final prediction. The possible labels we use are defined in the College of American Pathology reporting guidelines.

ZSS involves calculating pairwise character-based similarity scores between a predicted line of a report and each possible label using the fuzzy jaccard score as a subroutine (Algorithm 1). We use the line in the report with the highest probability computed using the line classifier as the predicted line. The similarity between the predicted line and a candidate label is computed with the Ratcliff-Obershelp algorithm. We evaluated several character-level string similarity algorithms, such as the Jaro–Winkler similarity, Levenshtein similarity, and Hamming distance but found that the Ratcliff–Obershelp approach performed best according to the mean F1-micro score using the training set averaged across the all splits for lung, colon, and kidney cancers for each data size. The label with the highest Ratcliff–Obershelp score is used as the final prediction. The full routine is described in Algorithm 2.

Zero-shot similarity (ZSS)

Input: Predicted line in a report, the probability output from the line classifier, a set of labels for a given field (the possible values that a data field can take), and a learned cutoff parameter used to predict 'NA' or not reported

Output: Predicted label

1. *For eachcandidate label, calculate its fuzzy jaccard score with the predicted line*
2. *Take the label with the highest fuzzy jaccard score. If there is a tie between multiple labels, take the label with the most characters as the prediction*
3. *If the fuzzy jaccard score is less than 0.5, then predict "other"*
4. *If the line probability is less than the cutoff, then predict "NA"*

Fuzzy jaccard algorithm

Input: Predicted line in a report and a candidate label

Output: Similarity score

1. *For each unique word in the candidate label, calculate its string similarity score with each unique word in the predicted line using the Ratcliff-Obershelp contiguous matching subsequence algorithm and find the max similarity score.*
2. *Sum up the max similarity scores across unique words for the candidate label*
3. *Scale the resulting sum by the number of unique words in the label*

Ensembling String Similarity with the SLA Approach:

While we found ZSS to be effective on its own, we found that it suffered from a few weaknesses. In particular, we found that the "other" class particularly challenging, as it consists of all possible values the field can take outside of the defined label set in the CAP protocols. For example, if the field is "procedure", then the "other" class corresponds to all other possible procedures not listed in the CAP protocols, which will all have low string similarity to the label "other". Furthermore, there are cases where the label in general is very dissimilar to how it occurs in the text. With these examples, no matter how many data points our algorithm is trained on, it will never get these right. Therefore, we aim to get the best of both ZSS and the SLA approach by developing a hybrid approach to the problem. If the final string similarity score is above a learned threshold, then we output the ZSS prediction. Otherwise, we output the SLA prediction. We call this method ZSS-thresholding.

We include an oracle method for the sake of comparison that chooses the SLA prediction if it is equal to the ground truth and the ZSS prediction otherwise. This oracle method serves as an upper bound on the performance of our string similarity enhancement of the SLA method. Our final method is ZSS-doc which is using ZSS on the entire text of the report instead of the lines output from the line classifier. This allows us to gauge how necessary the location targeting approach is for the ZSS methods.

BASELINE METHODS

1) Shared labels

Our first set of baselines is document-level classification methods, such as logistic regression, XGBoost, random forest, and support vector machines. These methods take as input all the tokens in a given report and predict the class value of a particular data field. The bag of words approach is used to vectorize the text in each document and train the outlined machine learning methods. This approach only uses the final document-level labels and is trained on the cancer of interest as well as the out-domain cancer reports.

Our next baseline is the hierarchical attention network (HAN) for document classification [26]. In particular, we study the hierarchical attention network (HAN) in terms of transfer learning. We pretrain the model on out-domain reports for a shared field and then fine-tune the model on in-domain reports.

We also use the SLA approach outlined in the previous chapter as an additional baseline. XGBoost is used as the line prediction model, while the logistic regression is used as the final label classifier. Location-based annotations are used to train the line prediction model, while the document-level label annotations are used to train the final classifier.

2) Unique labels

The baselines in the unique labels scenario include the ordinary document-level classifiers and the SLA approach similar to the previous case. All the methods in the unique labels case are trained on a single cancer domain.

HYPERPARAMETER TUNING

For all our experiments, we train each method with 40 randomly chosen hyperparameter values and choose the set that maximizes the average 4-fold cross-validation score.

Model-based parameters

For the *xgboost* method we sample from 1000 points from -2 to -0.5 in logspace for the learning rate; values 3 to 7 for max depth; a minimum split loss reduction to split a node from 0, 0.01, 0.05, 0.1, 0.5 and 1; a subsample ratio values from 0.5, 0.75, or 1; and L2 regularization values from 0.1, 0.5, 1, 1.5, or 2. For *logistic regression*, we sample 1000 evenly spaced points from -6 to 6 in logspace for the L1 regularization parameter.

SLA parameters

The SLA method has additional model-independent hyperparameters which are the n-gram size for the line classifier which is between 1 and 4, the n-gram size for the final classifier which is again between 1 and 4, the number of selected lines to use as input to the final classifier which is between 1 and 5.

Text similarity parameters

The text similarity approach has an additional model-independent cutoff parameter sampled from 0.05, 0.1, 0.15, 0.2, 0.25, and 0.3 which is used to handle NA values for a given data field. We specifically use the maximum line classifier probability from the positively predicted lines. If this probability is less than the cutoff value, then the predicted value from the string similarity method is replaced by "NA."

SLA + string similarity parameters

The SLA data augmented variation contains a threshold parameter that is sampled from 0.8, 0.85, 0.9, 0.95, and 1.0. This parameter is used to select instances outside the train set to augment the training data. All instances which have a similarity score with a label greater than the threshold value is used along with the string similarity prediction for data augmentation.

Similarly, the SLA thresholding variation contains a threshold parameter sampled from 0.8, 0.85, 0.9, 0.95, and 1.0. If a given instance has a string similarity score with a label greater than the threshold value, then the SLA prediction is replaced with the string similarity prediction.

RESULTS

We run two sets of experiments across lung, colon, and kidney cancers: one for the shared field and shared labels case and another for the shared field and unique labels case. Each method is trained on training data sizes of 10, 20, and 40 in-domain reports along with 372 out-domain reports with validation set sizes of 5, 10, and 20 in domain reports, respectively. The test set consists of 186 held out reports from the domain in question.

Each experiment is run 10 times where the training, validation, and test splits are randomly formed. We compare across methods using the mean micro-F1 and macro-F1 scores and obtain uncertainty bounds around the means.

Shared labels

Our experiments show that HCTC and HCTC-final consistently outperforms all other methods (Table 4.3). Compared to boosting, which performs the best among baselines on macro-F1, HCTC achieves performance gains by 0.03–0.04 in macro-F1 across data sizes. Compared to SLA, which performs the best among baselines on micro-F1, HCTC requires half the data to perform better in both macro-F1 and micro-F1. Additionally, we find that for data sizes 17 and 33, HCTC-final outperforms HCTC, suggesting the main benefit of transfer learning comes from the final classifier and not the line classifier.

Unique labels

ZSS-thresholding also requires approximately half the data to perform similarly or better than the baseline methods for the unique labels setting (Table 4.4). ZSS-thresholding with 8 points achieves an increase of 0.14 in micro-F1 and 0.16 in macro-F1 over boosting trained on 20 data points. Furthermore, ZSS-thresholding trained on 17 data points achieves an increase of 0.05 in micro-F1 and an increase of 0.06 in macro-F1 compared to boosting trained on 40 data points. A similar trend holds when computing differences in extraction quality between ZSS-thresholding and SLA which suggests that the string similarity approach enhances models trained on small data.

For ZSS and ZSS-thresholding, we additionally include micro-F1 and macro-F1 scores computed on test instances which have labels never seen during training across colon, kidney, and lung cancers (Figures 4.1 and 4.2). For colon cancer, the zero-shot performances are near or above 0.3 macro-F1 and 0.4 micro-F1 for both methods. For lung cancer, the performances are consistently near or above 0.25 macro-F1 and 0.4 micro-F1. For kidney cancer, the performance we see benefits of up to 0.1 macro-F1 and 0.25 micro-F1. These metrics show ZSS is a viable zero-shot approach for this application and is able to learn to predict classes never observed in the training set.

DISCUSSION

We have developed 2 ways to improve the performance of learning-based extraction systems when the amount of annotated reports is limited. For attributes where the tumor attribute and labels are shared across domains, it is natural to aggregate annotations across domains to augment the data used to train the models. Our experiments with enhancing the SLA method show that the gain in performance is consistent across data sizes; there is a 0.09 increase in micro-F1 and 0.02 increase in macro-F1 for data size 8 and 0.09 increase in micro-F1 and 0.04 increase in macro-F1 for data size 33 over the state-of-the-art averaged across the 3 cancers. We note that, to the authors' best knowledge, this is the first work to investigate transfer learning techniques across more than 2 cancers in NLP.

In the case of attributes where the labels differ across domains, we opt for a string similarity enhancement instead of a transfer approach. Because the categories for these attributes are unique for each domain, there is less room for improvement via transfer learning due to cancer-unique labels. String similarity is a more viable approach because typically in this case the text will contain strings close to the label names. Our experiments show that interpolating learning-based solutions with string similarity prediction can lead to a significant increase in performance—up to 0.26 micro-F1 and 0.23 macro-F1 for data size 8 and 0.04 micro-F1 and 0.06 macro-F1 for data size 33 over the state-of-the-art averaged across the 3 cancers. In terms of zero-shot performance of ZSS, the results vary across cancers and tops at 0.55 micro-F1 and 0.34 macro-F1 for a specific cancer.

Zero-shot learning has previously been studied in the context of medical information extraction, specifically on the public MIMIC II and MIMIC III datasets [34]. Rios and Kavululu [1] used natural language descriptors of labels and label space structure as auxiliary information to achieve zero-shot learning. Their approach matches textual summaries of reports obtained from attention-based CNNs to feature vectors of labels obtained from graphical neural networks and achieves recall-at-10 scores of up to 0.362 on MIMIC II and 0.495 on MIMIC III for zero-shot labels. Lu et al [54] similarly use pre-defined label relations, label descriptions, and pre-trained word embeddings as auxiliary information. Information from multiple graphs built on label descriptions, label taxonomy, and label co-occurrences obtained from graph convolutional networks are matched with document embeddings obtained from CNN encoders. They achieve recall-at-10 scores of up to 0.462 on MIMIC II and 0.553 on MIMIC III. While deep learning has been shown to be extremely effective in the presence of a large amount of labeled data, it can often struggle on smaller datasets. We note that MIMIC II and III contain 18,822 and 37,016 patients, respectively, orders of magnitude larger than our dataset size of 40.

Our findings motivate future directions for information extraction with small data regimes. While in preliminary experiments, we found that using pre-trained word embeddings to measure similarity performed worse than our string-based method, we believe one promising direction is taking advantage of models pre-trained using large corpuses of text on language modeling tasks. Recent work in NLP has shown fine-tuning such models on specific tasks with small amounts of data lead to improvements in performance for a given task. Combining such models, such as BERT [19] with the SLA framework can potentially improve upon ZSS-based methods especially for cases when synonyms of class names are used in the report in lieu of the class name. Furthermore, we did not study how much transfer learning benefits learning across different attributes for a particular cancer. Though most attributes have different label sets for a given cancer, there are instances where knowledge can be transferred. One such case is when a pathologist denotes that a particular attribute is not reported in the text which is applicable to many tumor attributes. Hence a fully unified extraction model may perform better than a model trained on a specific tumor attribute. In practice, it is also easier to maintain one model over maintaining many individual models.

Another promising direction is improving the ensembling method between machine learning methods and rules-based or string similarity methods for the unique labels case. Our results on the oracle ensembling model shows that there is still much room for improvement when combining string similarity predictions with machine learning predictions. For example, the oracle model has up to 0.075 improvement in both macro and micro-F1 over our method of thresholding based on the learned similarity score cutoff. Potential approaches include basing the decision-making process on the uncertainties of each algorithm or combining model probabilities and string similarity scores for each possible label and outputting the label with the highest score

CONCLUSION

Large datasets in medical contexts are expensive to generate, limiting the generalizability of many NLP systems. We develop a novel cancer-to-cancer transfer learning approach and a ZSS approach that can halve the amount of labeled data required, which potentially opens doors to more widespread implementation of these systems in the real world.

TABLES

Table 4.1. Extracted attributes and their possible values for the shared fields and shared labels case

Field	Possibles values
Histologic grade	Grade 1, grade 2, grade 3, grade 4, or not reported
Lymphovascular invasion	Present, absent, or not reported

Table 4.2. Extracted attributes and their possible values for the shared fields and unique labels case

Tumor Site	Possible values
Colon	Cannot be determined, cecum, colon not otherwise specified, hepatic flexure, ileocecal valve, left descending colon, other, rectosigmoid junction, rectum, right ascending colon, sigmoid colon, splenic flexure, transverse colon, or not reported

continued on next page

continued from previous page

Kidney	Upper pole, middle pole, lower pole, other, not specified, or not reported
Lung	Upper lobe, middle lobe, lower lobe, bronchus, or not reported
Histologic Type	Possible values
Colon	Adenocarcinoma, adenosquamous carcinoma, carcinoma, type cannot be determined, large cell neuroendocrine carcinoma, medullary carcinoma, micropapillary carcinoma, mucinous adenocarcinoma, neuroendocrine carcinoma poorly differentiated, other histologic type not listed, serrated adenocarcinoma, signet-ring cell carcinoma, small cell neuroendocrine carcinoma, squamous cell carcinoma, undifferentiated carcinoma, or not reported
Kidney	Acquired cystic disease associated renal cell carcinoma, chromophobe renal cell carcinoma, clear cell papillary renal cell carcinoma, clear cell renal cell carcinoma, collecting duct carcinoma, hereditary leiomyomatosis and renal cell carcinoma-associated renal cell carcinoma, mit family translocation renal cell carcinoma, mucinous tubular and spindle renal cell carcinoma, multilocular cystic clear cell renal cell neoplasm of low malignant potential, oncocytoma, other histologic type, papillary renal cell carcinoma, papillary renal cell carcinoma type 1, papillary renal cell carcinoma type 2, renal cell carcinoma unclassified, renal medullary carcinoma, succinate dehydrogenase sdh deficient renal cell carcinoma, t611 renal cell carcinoma, tubulocystic renal cell carcinoma, xp11 translocation renal cell carcinoma, or not reported

continued on next page

continued from previous page

Lung	Adenocarcinoma in situ mucinous, adenocarcinoma in situ nonmucinous, adenocarcinoma acinar predominant, invasive adenocarcinoma acinar predominant, pulmonary adenocarcinoma acinar predominant, adenosquamous carcinoma, atypical carcinoid tumor, carcinoma, combined small cell carcinoma, fetal adenocarcinoma, invasive adenocarcinoma micropapillary predominant, invasive adenocarcinoma papillary predominant, invasive adenocarcinoma lepidic predominant, invasive mucinous adenocarcinoma, invasive squamous cell carcinoma keratinizing, invasive squamous cell carcinoma non-keratinizing, invasive squamous cell carcinoma basaloid, large cell carcinoma, large cell neuroendocrine carcinoma, lymphoepithelioma, minimally invasive adenocarcinoma, minimally invasive adenocarcinoma mucinous, mucinous adenocarcinoma, mucoepidermoid carcinoma, non-small cell carcinoma, small cell carcinoma, squamous cell carcinoma in situ, solid adenocarcinoma with mucin, typical carcinoid tumor, squamous cell carcinoma, other, or not reported
Procedure	Possible values
Colon	Abdominoperineal resection, left hemicolectomy, low anterior resection, not specified, other, polypectomy, right hemicolectomy, sigmoidectomy, total abdominal colectomy, transanal disk excision, transverse colectomy, or not reported
Kidney	Total nephrectomy, partial nephrectomy, radical nephrectomy, other, or not reported
Lung	Bilobectomy, completion lobectomy, wedge resection, lobectomy, segmentectomy, pneumonectomy, other, or not reported

Table 4.3. Average micro-f1 and macro-f1 performance as a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports

	Macro-f1			**Micro-f1**		
In-domain training sizes	10	20	40	10	20	40
Hierarchical attention network	0.298	0.287	0.355	0.580	0.574	0.718
Logistic	0.344 (0.055)	0.441 (0.059)	0.467 (0.073)	0.634 (0.039)	0.676 (0.037)	0.708 (0.047)
Random forest	0.276 (0.025)	0.307 (0.034)	0.340 (0.044)	0.586 (0.039)	0.614 (0.030)	0.641 (0.036)
SVM	0.221 (0.048)	0.269 (0.034)	0.310 (0.034)	0.519 (0.102)	0.560 (0.051)	0.570 (0.052)
Boost	0.436 (0.036)	0.468 (0.044)	0.548 (0.052)	0.704 (0.049)	0.732 (0.037)	0.789 (0.038)
SLA	0.211 (0.024)	0.338 (0.037)	0.466 (0.043)	0.579 (0.031)	0.700 (0.029)	0.790 (0.026)
HCTC	0.461 (0.038)	0.508 (0.034)	0.544 (0.028)	0.797 (0.023)	0.832 (0.022)	0.858 (0.018)
HCTC-final	0.421 (0.034)	0.502 (0.047)	0.584 (0.048)	0.776 (0.027)	0.842 (0.030)	0.882 (0.024)
HCTC-line	0.205 (0.013)	0.341 (0.035)	0.473 (0.040)	0.579 (0.044)	0.700 (0.041)	0.800 (0.025)

Table 4.4. Average micro-f1 and macro-f1 performance across a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports

	Macro-f1			**Micro-f1**		
In-domain training sizes	10	20	40	10	20	40
Hierarchical attention network	0.051	0.026	0.079	0.255	0.118	0.264
Logistic	0.208 (0.029)	0.277 (0.047)	0.354 (0.048)	0.473 (0.033)	0.578 (0.053)	0.651 (0.033)
Random forest	0.177 (0.045)	0.223 (0.024)	0.323 (0.043)	0.438 (0.044)	0.516 (0.042)	0.618 (0.028)
SVM	0.152 (0.029)	0.172 (0.031)	0.239 (0.030)	0.387 (0.066)	0.425 (0.036)	0.517 (0.044)
Boost	0.155 (0.021)	0.288 (0.040)	0.382 (0.034)	0.421 (0.029)	0.608 (0.051)	0.715 (0.028)
SLA	0.095 (0.015)	0.178 (0.012)	0.219 (0.016)	0.472 (0.036)	0.651 (0.023)	0.738 (0.016)
ZSS	0.442 (0.024)	0.436 (0.017)	0.428 (0.028)	0.743 (0.016)	0.737 (0.011)	0.742 (0.009)
ZSS-doc	0.359 (0.023)	0.356 (0.022)	0.341 (0.019)	0.546 (0.024)	0.540 (0.021)	0.528 (0.007)
ZSS-thresholding	0.441 (0.024)	0.447 (0.022)	0.449 (0.029)	0.739 (0.017)	0.765 (0.018)	0.780 (0.015)
Oracle	0.454 (0.031)	0.501 (0.029)	0.529 (0.024)	0.775 (0.019)	0.829 (0.017)	0.862 (0.011)

FIGURES

Figure 4.1: Average macro-f1 (A) and micro-f1 (B) performance for test instances where the label is not seen during training as a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports. The results presented include the mean performance using ZSS across 10 random splits of the data and 95 percent confidence intervals for the unique labels case. Note that the number of zero-shot test instances decreases as the number of training instances increase.

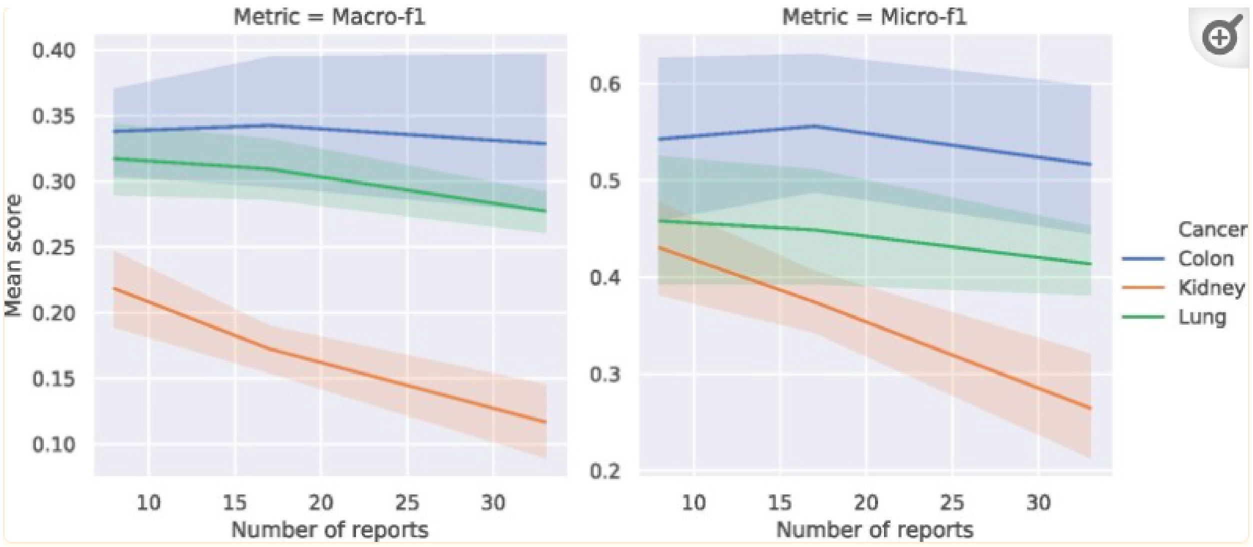
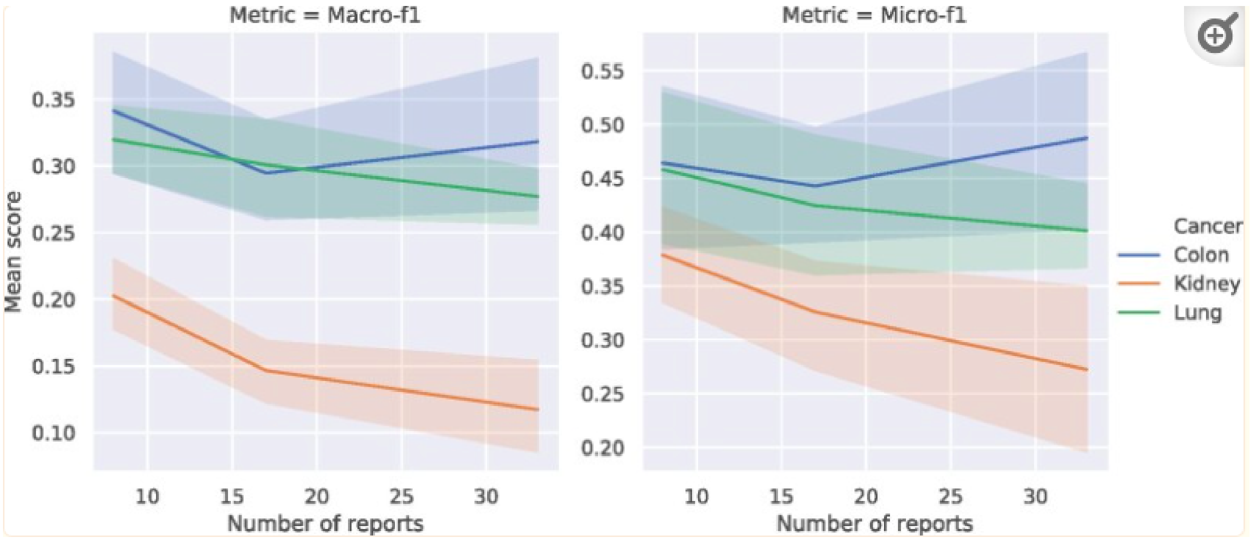


Figure 4.2: Average macro-f1 (A) and micro-f1 (B) performance for test instances where the label is not seen during training as a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports. The results presented include the mean performance using ZSS-thresholding across 10 random splits of the data and 95 percent confidence intervals for the unique labels case. Note that the number of zero-shot test instances decreases as the number of training instances increase.



Chapter 5

Improving fitted Q-evaluation and model-based evaluation via stability weighting

This chapter revolves around improving offline evaluation estimates of reinforcement learning policies. Offline evaluation is critical for estimating the value of agents in high risk or high cost settings. For example, in healthcare, it is not feasible to test out treatment policies on real patients in order to compare outcomes of patients under each possible policy [53]. Instead, offline methods leverage logged or offline datasets $\mathbb{D} = \{(s_i, a_i, r_i, s_i')\}$ which contain historical transitions from the actual environment. If the logged data is intelligently leveraged by a practitioner, risks or costs stemming from online data collection can be reduced. In actual data problems, qualitative checks and domain knowledge should first be applied to the logged data to validate whether the underlying dynamics of the target environment are accurately captured.

A unique challenge in offline evaluation is that it is not possible to validate such estimates, since it requires running policies on the real environment [67]. As a consequence, model selection is difficult compared to supervised learning because typical validation is not possible. This is a significant roadblock to offline reinforcement learning, because despite the fact that many offline evaluation methods have been proposed in previous work, knowing when to use one particular method over another is an open problem. In our work, we attempt to address this need by investigating using model stability as an avenue for adaptively combining offline estimates on simulated environments which can be viewed as a form of soft model selection.

This chapter is organized as follows. First we start with a background on reinforcement learning and offline evaluation. We next outline our proposed method for adaptively combining offline estimates from state-of-the-art offline evaluation algorithms using stability as a weighting mechanism. Then we detail the simulated environment which serves as a benchmark for offline evaluation and finally the experiments conducted on simulated environments.

BACKGROUND

Reinforcement learning

In the reinforcement learning setting, we assume we have an agent which interacts with some environment over time [82]. The environment is typically assumed to be a Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, r, \mu_0, \gamma)$ where $\mathcal{S}$ is the state-space, $\mathcal{A}$ is the action space, $p(s'|s,a)$ is the transition function given state s and action a, $r(s,a)$ denotes the reward function given a state s and action a, $\mu_0(s)$ is the initial state distribution, and $\gamma \in [0,1]$ is the discount factor. At each time step t, the agent receives some state s_t and selects an action a_t via $\pi(a_t|s_t)$ to take and receives reward r_t and the next state s_{t+1} from the environment. Thus, at each time step, there is an interaction with the environment and feedback, which the agent must use to plan actions in the future as shown in Figure 5.1.

Given a policy $\pi(a|s)$, the value of π is defined to be $v(\pi) = \mathbb{E}_{s\sim\mu_0}\left[V^\pi(s)\right]$ where

$$V^\pi(s) = \mathbb{E}_{s'\sim p(s,a), a\sim\pi}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t | s\right]$$

is the state value function in the infinite horizon setting and

$$V^\pi(s) = \mathbb{E}_{s'\sim p(s,a), a\sim\pi}\left[\sum_{t=1}^{N} \gamma^{t-1} r_t | s\right]$$

in the finite-horizon setting with maximum horizon length N. Related is the Q-function which restricts the action taken at state s and is defined to be

$$Q^\pi(s,a) = \mathbb{E}_{s'\sim p(s,a)}(r(s,a) + \gamma * V(s')).$$

In the typical reinforcement learning setting, the goal is to train a policy π such that the value function is maximized. Mathematically we can write this as

$$\max_\pi \mathbb{E}_{a_t\sim\pi(.|s_t), s_{t+1}\sim p(.|s_t,a_t)}\left[\sum_t \gamma^{t-1} * r(s_t, a_t)\right].$$

Many reinforcement learning algorithms have been proposed over the years, including policy gradient methods, model-based reinforcement learning, and temporal difference methods. Policy gradient methods require optimizing policies directly on the expected return from monte carlo rollouts on the environment via gradient descent [81, 37] but often require many interactions with the environment to perform well. Temporal difference learning methods involves fitting a value function by minimizing the temporal difference error [82] which can be written as

$$\min_Q \sum_{i=1}^{N} \left(Q(s_i, a_i) - r(s_i, a_i) - \gamma * V(s_i')\right)^2,$$

where N is the size of the dataset. A policy can then be extracted by taking actions which maximize the Q-function given a state. Popular temporal difference algorithms include SARSA [82] and Q-learning [88]. Model-based reinforcement learning methods are the most sample efficient and generally learns the dynamics, in other words the reward and transition functions, of the environment first and leverages the predicted dynamics for decision making [82]. Different variations of model-based reinforcement learning have been studied previously, including settings where dynamics models are used for planning at testing time or where a policy is learned along with the dynamics model by employing backpropagation through time [65]. These algorithms listed for policy gradients, temporal difference learning, model-based learning, and other hybrid methods, such as the actor-critic method, require access and interaction with the actual environment over time.

An additional goal in reinforcement learning is to evaluate the value of a policy π. The aim is to compute an estimate of $V^{\pi}(s)$ where s is directly given or where $s \sim \mu_0$. In the case where $s \sim \mu_0$, the simplest solution is to run π on the environment a number of times and average over the collected sum of discounted rewards. In other words, rewards collected along monte carlo rollouts on the actual environment are used to estimate the value of π. Additionally, temporal difference learning methods have commonly been employed for value estimation. Prominent temporal difference learning methods include temporal difference (TD) learning [79], gradient temporal difference (GTD) learning [80], temporal difference learning with gradient correction (TDC) [83]. Such methods train a Q-function which can then be used to evaluate the policy. An additional research area is off-policy evaluation or offline evaluation which is the focus of this work. This area revolves around estimating the value of a policy using data deriving from another policy. We outline offline evaluation in greater detail in the next section.

Offline setting

In this chapter, we focus on the offline reinforcement learning setting which is data-driven. Unlike the typical reinforcement learning setting, one only has access to logged data of the form $\mathcal{D} = (s_i, a_i, r_i, s'_i)$ and access to the real environment $\mathcal{M}$ is not available [67]. The logged data is derived from one or more behavior policies which is denoted by π_b. In the episodic reinforcement learning scenario, the logged data can also be written as $\mathcal{D} = \{\tau_i\}$ where $\tau = (s_1, a_1, r_1, s'_1, \ldots, s_n, a_n, r_n, s'_N)$ where the length of the episode N can vary across episodes. In the typical setting, the logged data is static and assumed to come from the target environment. Additional data collection from the actual environment is not possible. A visual depiction of offline reinforcement learning is shown in Figure 5.2.

The offline setting is generally more difficult than the typical reinforcement learning setting, also known as the online setting, because running a target policy on the true environment is not possible. This is especially a challenge if the logged data has insufficient coverage of the state-space. Thus, states and rewards that a target policy is likely to visit and collect may not be represented in the logged data if the behavior policy is unlikely to visit them. This issue is commonly referred to as distribution shift between the behavior policy and the target policy. In practice, trained policies are often kept similar to the behavior policy in order to mitigate the problem of distribution shift and the risk of overestimating values in states not sufficiently visited [89, 24, 32].

The goal of offline evaluation or off-policy evaluation is to estimate $v(\pi_e)$ where π_e denotes the evaluation policy. The evaluation policy is the policy which we want to estimate the value using the logged data $\mathcal{D}$ deriving from the behavior policy. Offline evaluation is especially difficult because the goal is estimate the value of a policy which is typically not represented in the logged data. Furthermore, the issue of insufficient coverage plagues offline evaluation as well, because states that the evaluation policy may visit may not be represented in the logged data. In comparison, evaluating the behavior policy π_b is straightforward, because the logged returns can directly be used to estimate $v(\pi_b)$.

Relation to PCS

The PCS (Predictability, computatibility, and stability) framework [94] outlines principles for a data science problem and an approach with an underlying aim of providing reliable, responsible, and transparent results in the data science life cycle. Many of the ideas outlined in the PCS framework are very applicable to this data-driven setting of reinforcement learning. For example, predictability is an important reality check when working with logged data. Before extrapolating results to the actual environment, reality checks on the logged data and trained policies or critics are required especially in high-stakes data problems that motivate offline reinforcement learning, such as clinical decision making, autonomous driving, and robotics.

Stability is also an essential check as results should be reproducible to small perturbations to data and models. Stability is especially important in reinforcement learning where the performance of particular algorithms rely on careful tuning and tricks in practice. Our work is tied to the stability principle in PCS as the main notion underlying the methodology is inspired by the stability of models. We further apply this principle to test the robustness of our proposed methods to human judgement calls in our experiments which can potentially impact results and conclusions.

OFFLINE EVALUATION METHODS

Many algorithms have been proposed to do offline evaluation of reinforcement learning agents, including importance sampling methods, doubly robust methods, fitted q-evaluation (FQE), and model-based evaluation [47, 66, 44, 13, 56].

The basic importance sampling estimator [66] uses importance weights $\frac{\pi_e(a|s)}{\pi_b(a|s)}$ to weight the observed rewards in the logged data to compensate for the distribution shift between the behavior π_b and evaluation policy π_e. If the behavior policy is known and $\pi_e(a|s) = 0$ whenever $\pi_b(a|s) = 0$ for all (s, a), then the importance sampling estimator is unbiased and consistent for the value of π_e. However, in practice, importance sampling estimators are prone to high variance especially in the scenario when π_e significantly deviates from π_b.

Doubly robust methods [33, 85] combine the importance sampling method with an estimator of the reward function. They are called doubly robust because the estimator is consistent and unbiased for the policy value if either the importance sampling estimate or the reward function estimator is unbiased. Doubly robust methods typically have lower variance than the importance sampling estimator. However, empirical work have shown the doubly robust estimator to have larger errors than counterpart offline evaluation methods due to high variance in importance sampling weights or stochastic transitions [86, 85].

In this work, we focus on FQE and model-based evaluation, which we cover in more detail in this section. We focus specifically on FQE and model-based evaluation, since both have empirically been shown to perform well across a variety of reinforcement learning tasks and are popularly used [23]. Note, however, that performance across different evaluation methods can vary greatly across different types of reinforcement learning tasks. Furthermore, it is not straightforward to estimate which particular algorithm or even model is suited for a specific task. This motivates a need for a way to extract some signal about when a particular method is suited for a task and/or region of the state space, since better estimates can be constructed by weighting towards more suitable methods.

Fitted Q-evaluation

Fitted Q-evaluation (FQE) is a off-policy temporal difference learning algorithm based on a slight variation of the fitted Q-iteration algorithm [47]. FQE involves learning the following Q function from the logged data:

$$Q^\pi(s, a) = \mathbb{E}^{\pi}_{s'\sim p(s,a)} \left(\sum_{i=1}^{N} \gamma^{i-1} r_i | s_1 = s, a_1 = a \right),$$

which can intuitively be interpreted as the value of taking action a at state s and then following policy π for the rest of the trajectory. Note that the value of the policy can be written in terms of the Q-function $\mathbb{E}[Q^\pi(s, \pi(s))]$ where $s \sim \mu_0$. The Q-function is trained by minimizing the following

$$\mathbb{E}_{(s,a,r,s')\sim\mathbf{D}}[(Q_\theta(s,a) - r - \gamma * Q_{\theta'}(s', \pi(s'))^2],$$

where θ are the parameters of the function class used to approximate the Q-function and θ' are the parameter values in the previous iteration of the training process.

FQE has been studied theoretically in a variety of literature. Much of the work relies on the assumption of completeness which says for any function $f \in F$, we have that $T^\pi f \in F$, where F is the function class and T^π is the Bellman operator defined as

$$T^\pi V^\pi(s) = \sum_{a\in A} \pi(a|s) \left[r(s,a) + \gamma * \sum_{s'\in S} P(s'|s,a) V^\pi(s') \right],$$

where $r(s,a)$ is the reward function, A is the set of possible actions, $V^\pi(s)$ is the value function, and $P(s'|s,a)$ is the transition function. Duan et al. [20] proved a minimax lower bound for FQE using a linear function approximation which matched the upper bound under completeness (i.e. the Bellman operator maps to state-action value functions that are a linear combination of the given features). Under the same assumption, Hao et al. [28] proved the FQE estimator with linear function approximation is asymptotically normal around 0 and its asymptotic variance achieves the Cramer-Rao lower bound for the value function. More recently, Zhang et al. [96] focused on FQE with general and differential function approximators using Z-estimation theory. They show the FQE estimation error is asymptotically normal, prove a finite sample error bound, show that bootstrap estimators are distributionally consistent, and prove that the general FQE estimator achieves the Cramer-Rao lower bound under completeness.

In practice, FQE has found more empirical success compared to importance sampling and doubly robust methods due to the lower variance of the estimate and generalizability from function approximation. Additionally, computing value estimates using FQE does not rely on simulating entire rollouts unlike the model-based method, where errors can compound if the horizon is especially long. However, a downside to FQE is that is unclear how to tune the parameters and architecture of FQE if function approximation is used.

FQE is typically implemented as shown below [47]. We assume an evaluation policy π_e, a function class $\mathbb{F}$, and a dataset $D = \{(s_i, a_i, r_i, s_i')\}_{i=1}^n$. The algorithm proceeds as follows:

1. Randomly initialize parameters of $Q_0^{\pi_e} \in \mathbb{F}$

2. for k from 1 to K

3. a) Compute FQE target $y_i = r_i + \gamma Q^{\pi_e}_{k-1}(s'_i, \pi_e(s'_i))$ for every i
   b) Construct training data as follows: $D_{FQE_k} = \{(s_i, a_i, y_i)\}_{i=1}^n$
   c) Solve $Q^{\pi_e}_k = argmin_{f \in \mathbb{F}} \frac{1}{n} \sum_{i=1}^n (f(s_i, a_i) - y_i)^2$
4. Output $Q^{\pi_e}_K$

Model-based evaluation

Model-based (MB) evaluation is similar to model-based reinforcement learning in that it involves learning a simulation of the real environment $\mathcal{M}$. More specifically, both the transitions $p(s'|s, a)$ and the reward function $r(s, a)$ are learned via the logged data $\mathcal{D}$ using standard supervised learning techniques. The fitted transition and reward functions are then used to simulate trajectories using the behavior policy. The observed rewards of the simulated trajectories can then be used to calculate values for the behavior policy. These trajectories can be referred to as monte carlo rollouts.

To estimate the Q-value using the fitted reward and transition models, we have that

$$\hat{Q}^{\pi}_{MB}(s, a) = \sum_{t=0}^{N} \hat{r}(s, a) + \gamma * \hat{V}_{MB}(s')),$$

where $s' \sim \hat{p}(s, a)$ and $\hat{V}(s)$ is given by

$$V^{\pi}_{MB}(s) = \mathbb{E}_{s' \sim \hat{p}(s,a), a \sim \pi_e} \left[\sum_{t=1}^{N} \gamma^{t-1} \hat{r}_t | s \right].$$

The model-based method performs well when the environment transition and reward functions are simple and can be easily approximated through function approximation. It is typically easier to tune the hyperparameters of dynamics models, compared to tuning FQE models, since a validation loss based on the observed transitions and rewards can be computed on a hold-out set. A downside to the model-based method is that estimates derived from simulated trajectories may compound errors over time if the maximum horizon length is long. This is the case since values are calculated from rewards along trajectories which are simulated autoregressively for the model-based method. This is not the case for FQE, which directly output q-values.

2D WORLD

In this section, we detail the simulator we created to benchmark the offline evaluation, the behavior policies, the evaluation policies, and the logged data generation process.

Environment

The environment is depicted in Figure 5.3. The agent observes its position relative to the x and y axes, its horizontal and vertical velocities, and the time step. Each episode has a maximum horizon of 300 time steps and terminates when the time step reaches the maximum horizon or when the agent reaches the goal (upper right corner given by $x \geq 4$ and $y \geq 4$). Note that the agent's x and y positions as well as the velocities are continuous values. The boundaries of the environment are given by the following lines: $x = 0, y = 0, x = 5, y = 5$. The agent is within the boundaries at all times. At each time step, the agent receives a negative reward conditioned on its x and y position outlined in Figure 5.3. If the agent successfully reaches the goal, the agent receives a completion reward of +10. At each step, the agent can choose from a set of 9 actions which correspond to moving to the left, right, up, down, and neutral as well as combinations of the horizontal and vertical moves.

The transition dynamics of the environment is detailed as follows. The horizontal and vertical velocities at time t are outlined by

$$vel_t = \max(\min(vel_{t-1} + a_t * f, 0.1), -0.1),$$

where $f = 0.001$, $a_t \in \{-1, 0, 1\}$. The velocities then affect the horizontal and vertical positioning as follows:

$$pos_t = \max(\min(pos_{t-1} + vel_t, 5), 0).$$

If the agent hits the horizontal or vertical boundaries of the environment, its corresponding directional velocity is set to 0. Note that the transitions and rewards are deterministic. However, the starting state of the agent is stochastic. The agent's x and y positions are uniformly sampled from $[0, 5/6]$ at time step $t = 1$, while the horizontal and vertical velocities are set to 0.

Policies

Two behavior policies were constructed by training deep Q-networks on the 2D world environment in typical online fashion. A multi-layer perceptron (MLP) network with 2 hidden layers with a hidden layer size of 50 were trained using the ADAM optimizer [41] with a learning rate of 0.001 and a batch size of 32 for both behavior policies with varying number of updates and initializations. Stochasticity was artificially embedded into the resulting Q-networks by adding a random probability of 0.25 where the agent performs a random action instead of the action given by its Q-network. The resulting behavior policies π_{b1} and π_{b2} had online value estimates of -79.4 and -83.4, respectively. The vertical and horizontal positions over time of each behavior policy are given by Figure 5.4.

Three evaluation policies were constructed in a similar fashion as the behavior policies with differing number of updates and initializations. Unlike the behavior policies, stochasticity was not embedded into the policy. The resulting evaluation policies π_{e1}, π_{e2}, and π_{e3} had online value estimates of -72.5, -85.0, and -92.0, respectively. The vertical and horizontal positions over time of each evaluation policy are given by Figure 5.5.

Offline datasets

Behavior policies π_{b1} and π_{b2} were used to generate two offline datasets from interacting with the actual environment. Datasets D_{b1} and D_{b2} were derived from running the corresppnding policy for 1000 episodes in total. Summary statistics, such as the size of the datasets and proportions of each action taken, are shown in Table 5.1.

OFFLINE EVALUATION TRAINING

We pair up each evaluation policy with each behavior policy to produce 6 pairs of behavior and evaluation policies. We also include two pairs that consist of each behavior policy paired up with itself as the evaluation policy. The goal is to estimate the value of the evaluation policy using the logged data deriving from the corresponding behavior policy for each pair of behavior policy and evaluation policy. The two offline evaluation methods we consider are FQE and model-based evaluation outlined in the previous section.

Fitted Q-evaluation

We detail how we train a FQE model on the logged data $\mathbb{D}$. A MLP with 2 residual blocks and a hidden layer size of 50 was initialized randomly. The neural network is trained on the tuples of $\mathbb{D}$ using the ADAM optimizer [41] with a learning rate of 0.001 using a batch size of 32. The FQE model was trained for a max number of iterations of 75000. At every 500 iterations, the temporal-difference error was computed on a validation set of 20 episodes from the logged data.

Typically in the supervised learning case, model selection is used via computing the target metric on a validation set. In this setting, we cannot compute the target metric on the validation set, because the validation set contains trajectories by following actions derived from the behavior policy. Instead, we can use the temporal difference error as a proxy for the target metric where the temporal difference error is defined by

$$Q_{\pi_e}(s,a) - r(s,a) - \gamma * Q_{\pi_e}(s',a'),$$

where (s,a,r,s',a') denotes the state, action, reward, next state, and next action. Note that only the true Q function of π_e minimizes the temporal difference error. Early stopping was applied by computing the absolute difference of the mean of the last 5 temporal difference error estimates and the previous 5 starting at the previous index. Mathematically we denote the stopping condition as

$$|TD_{i-5,i} - TD_{i-6,i-1}| < 0.001.$$

Model-based evaluation

We now detail how we train a model which learns the dynamics of the environment (reward and transition functions) to estimate values of offline agents. We first split the logged data $\mathbb{D}$ into a training set $\mathbb{D}_{train}$ and a validation set $\mathbb{D}_{val}$. We then initialize two neural network models, one that learns the reward function $r(s,a)$ and another that learns the transition function $p(s,a)$. Both neural network models were initialized with a hidden layer size of 200, with 3 hidden layers, and a learning rate of $5e^{-4}$ using the ADAM optimizer [41].

Both the reward and transition models are trained in a typical supervised learning fashion unlike FQE with the data being organized as $\{((s,a),r)\}$ and $\{((s,a),s')\}$ for the rewards and dynamics model, respectively. The mean squared error was used as the loss function train both the rewards and transition model. Each model trained using a max number of iterations of 50000. Every 1000 steps the loss was calculated on the validation data and early stopping was applied with a patience of 7.

INTEGRATING FQE AND MODEL-BASED ESTIMATES

In this section, we propose using empirical stability estimates as an avenue for improving FQE and model-based estimates of the evaluation policy conditioned on the state space and action. We weight the outputs from each method based on our empirical stability estimates across ensembles of randomly initialized neural networks. Our idea is inspired from previous work in weighted online learning, where past predictability is used to weight different online predictors [11, 5]. Cesa-Bianchi et al. [11] weight online boolean predictors using exponential weighting computing using the number of mistakes made by each predictor in the past. Altieri et al. [5] propose the CLEP (combined linear and expoential predictors) algorithm for forecasting covid-19 cases and deaths. CLEP weights each predictor based on recent predictive performance, where more accurate predictors are assigned higher weights. Unlike these works, we use the stability of ensembles conditioned on the state-space as a weighting mechanism instead of local or past predictive performance.

The main contribution of our work is leveraging model stability for the goal of adaptive model weighting between FQE and model-based estimation. In the offline evaluation setting, the same insufficient coverage problem of using logged data in leiu of the environment affects both FQE and model-based estimation. However, we hypothesize that due to the differing training objectives of FQE and dynamics model learning, FQE and dynamics models likely have varied success conditioned on the state space and reinforcement learning task.

EMPIRICAL EVIDENCE FOR LEVERAGING MODEL STABILITY ACROSS AN ENSEMBLE

In this section, we empirically show evidence that FQE and model-based evaluation can outperform one another conditioned on the state-space despite having been trained on the same logged data. We then show that stability from an ensemble of models initialized with different random seeds provides a positive signal for adaptive model weighting.

First, we compare FQE and model-based estimates on (s, a, r, s') tuples deriving from the evaluation policy. We first sample rollouts from the evaluation policy on the actual environment and sample 500 (s, a, r, s') tuples from the resulting dataset. For each sample tuple, we extract the q-value from the trained FQE and dynamics models and the online target estimate using the true environment via monte carlo rollouts. Figures 5.6 and 5.7 show the sample tuples and highlights which method outperforms the other on the conditioned state. We see that empirically success in extrapolation to states with lower coverage is dependent on the behavior policy and evaluation policy pair and the state itself. Tables 5.2 and 5.3 also show that the difference in errors of each method on partitions where one outperforms the other is large. This suggests that weighting between FQE and the model-based approach may considerably improve value estimation averaged across samples across the state-space if we are able to determine which method is likely to outperform the other adaptive to the state the agent is on.

Next, we show how stability via ensembling models is one viable avenue for extracting a positive signal for local predictiveness, which is related to weighted online learning. However unlike previous work which uses local predictiveness as a weighting mechanism, we use the stability of models. First we train an ensemble of 5 FQE models and 5 pairs of reward and transition models using a random initialization of the network weights using the same procedure outlined above. For each tuple in the sampled tuples deriving from the evaluation policy, we then compute the stability from the ensemble, defined by the standard deviation across the ensemble predictions. We report the proportion of times the method with more instability had the higher error conditioned on the state and action pair from the sampled tuples from the evaluation policy in Table 5.4. The high proportions across evaluation and behavior policy pairs show the positive relation between model stability and the error of the model in this offline evaluation setting. Note that this signal, although close to 0.5 for some pairs, is powerful in this setting where it is not possible to estimate model errors on a hold-out set, and thus do any type of model selection or tuning.

SOFT STABILITY WEIGHTING (SSW)

In this section, we propose one method to weight between FQE and the model-based method using stability as a weighting mechanism. We first assume an ensemble of FQE models and an ensemble of dynamics models are trained using the logged data deriving from the behavior policy. Given an arbitrary state s and action a, we can then compute an stability estimate for FQE (and analogously the model-based method) based on the predictions of the ensemble of FQE models. The stability estimate is based on the standard deviation of the predictions of the ensemble of FQE models conditioned on state s and action a, which we denote as $std_{FQE}(s,a)$. We normalize the previous standard deviation using quantiles of the distribution of standard deviations of ensemble predictions computed on samples from the behavior (logged) dataset, denoted as $quantile_{D_{\pi_b}}(std_{FQE}, c)$ where D_{π_b} is the dataset corresponding to the behavior policy π_b and c is the quantile. The overall stability value for FQE can then be written as

$$u_{FQE}(s,a) = \frac{std_{FQE}(s,a) - quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, c)}{quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, 1-c) - quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, c)},$$

where $1 - c > c$. We additionally bound the stability by 0 and 1 in the case where $std_{FQE} < quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, c)$ or $std_{FQE} > quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, 1-c)$. We also noted that normalizing the standard deviations instead of using the standard deviations directly helped performance empirically. We use quantiles instead of minimums and maximums for robustness against outliers in the logged data. Lastly, note that we weight between median values of each ensemble due to the robustness of the median to outliers.

Once both $u_{FQE}(s,a)$ and $u_{MB}(s,a)$ are computed, we can compute the weight α as follows:

$$\alpha(s,a) = \sigma\left(\log\left(\frac{u_{FQE}(s,a)}{u_{MB}(s,a)}\right)\right),$$

where σ is the sigmoid function and $0 \leq \alpha(s, a) \leq 1$. The final soft stability weight estimate can then be written as

$$SSW(s, a) = Q_{FQE}(s, a) * (1 - \alpha(s, a)) + Q_{MB}(s, a) * \alpha(s, a).$$

Intuitively, $\alpha(s, a)$ biases towards 0 and puts more weight on $Q_{FQE}(s, a)$ if the stability value of the FQE ensemble is low relative to model-based method. $\alpha(s, a)$ biases towards 1 and puts more weight on $Q_{MB}(s, a)$ if the stability value of the FQE ensemble is high relative to model-based method. In the case that the stability values for each method is similar, $\alpha(s, a)$ biases towards 0.5 which puts equal weight on $Q_{FQE}(s, a)$ and $Q_{MB}(s, a)$.

SOFT STABILITY WEIGHTING VIA PARTIAL ROLLOUTS (SSWPR)

In this section, we propose a second stability-based weighting method based on rollouts of length k from dynamics models where k is less than the maximum horizon length and rollouts are defined as simulated trajectories. We refer to these rollouts as partial rollouts since the rollouts are prematurely terminated at step k before the end of the simulated trajectory. This second stability weighting method is inspired from the previous stability based adaptive weighting mechanism in the previous section and the λ-k method studied in previous literature [82].

Given an ensemble of dynamics models, an ensemble of FQE models, and a specific state-action pair (s, a) we want a conditional value estimate for, we compute independent partial rollouts up to k steps starting from (s, a) from each of the dynamics models in the given ensemble. At each simulated step of each rollout, we compute the $SSW(s_t^i, a_t^i)$ between individual pairs of FQE and dynamics models where (s_t^i, a_t^i) is the state and action at time t in the ith partial rollout as well as CR_t^i which is the discounted sum of rewards of partial rollout i up to time t. Given a time point t and a partial rollout i, we then have an estimate of the value of (s, a) given as

$$\hat{val}_t^i(s, a) = CR_t^i + \gamma^t * SSW(s_t^i, a_t^i).$$

We compute stability values using the standard deviation across all partial rollouts for a fixed t as

$$u_t = std(\hat{val}_t(s, a)).$$

We can then compute weights using the softmax function across t:

$$\alpha_t = softmax(-u)_t,$$

where

$$\sum_{t=1}^{k} \alpha_t = 1.$$

We then compute final value estimate for (s, a) as

$$\sum_{t=1}^{k} \alpha_t * median(\hat{val}_t(s, a)).$$

where the median is computed across partial rollouts. Intuitively, we can interpret the SSWPR estimate as weighting across time where estimates at each time point is an interpolation between the SSW estimate at time t in the partial rollout and the collected rewards from partial rollout i up to time t. The combination of estimates at different simulated time points is the main difference of the SSWPR method from the SSW method. Unlike SSW, SSWPR can leverage value estimates from simulated states local to the original state that is being conditioned upon. If the value estimates at a simulated state at time t are more stable, then more weight is placed on $median(\hat{val}_t(s, a))$ relative to other times and vice versa. Again, we use the median instead of the mean due to the robustness of the median to outlier values. We show a depiction of the SSWPR method in Figure 5.8.

The full algorithm is given below. Note that for the experiments in this work, we used $k = 5$ steps for the partial rollout length. We chose to use a smaller value of 5 compared to the max time step of 300 in the 2D-world environment due to the computational cost of querying value estimates on partial rollouts. However, in our stability experiments we compare against using values of $k = 3$ and $k = 7$ to check the robustness of SSWPR to a perturbation of this parameter.

Soft stability weighting via partial rollouts algorithm

Inputs: An ensemble of FQE models $\{FQE_i\}$, an ensemble of dynamics models $\{(\hat{p}_i, \hat{r}_i)\}$ where $\hat{p}$ denotes the transition model and $\hat{r}$ denotes the rewards model, an ensemble size of $N_{ensemble}$, a maximum horizon of k, and a given state and action (s, a)

1. For i in 1 to $N_{ensemble}$
   a) Compute partial rollout PR_i from transition model $\hat{p}_i$ for up to k simulated steps
   b) For j in 1 to $N_{ensemble}$
      i. Compute $SSW(s_t^i, a_t^i)$ using FQE model FQE_j and dynamics model $(\hat{p}_j, \hat{r}_j)$ for each time step t in PR_i

ii. Compute conditional value for (s, a) using time step t as $\hat{val}_t^i(s, a) = CR_t^i + \gamma^t * SSW(s_t^i, a_t^i)$ where $CR_t^i = \sum_{j=1}^{t} \hat{r}_i * \gamma^{t-1}$ is the collected rewards from the partial rollout i up to time t

2. For t in 1 to k

   a) Compute standard deviation $std(\hat{val}_t(s, a))$ across rollouts

   b) Compute median value across rollouts $median(\hat{val}_t(s, a))$

3. Compute weights $\alpha_t = softmax(-std(\hat{val}_t(s, a)))_t$

4. Output $\sum_{t=1}^{k} \alpha_t * median(\hat{val}_t(s, a))$ as the final conditional value estimate

BASELINE METHODS

To compare against the proposed methods of incorporating stability in adaptively combining estimates from FQE and the model-based method, we outline baseline methods in this section. First, we use an ensemble of FQE models and an ensemble of dynamics models for model-based evaluation as two separate baselines for the stability methods. Ensembles are required to make a fair comparison, because the proposed methods require stability estimates resulting from ensembling. Note that we use medians instead of means of the ensemble due to the robustness of the median to outliers compared to the mean. Next, we take a simple average of conditional estimates from the FQE ensemble and the ensemble of dynamics models as a further baseline. This baseline shows whether the stability weighting mechanism has any performance boosts compared to non-adaptively combining estimates. The last baseline is an adaptive oracle selection between FQE and model-based method where value estimates conditioned on the state come from the method with the lower error. This final baseline is unrealistic in practice but serves as a useful benchmark for the potential improvement of the two proposed methods.

RESULTS

We outline the evaluation procedure of the two proposed methods, as well as the baseline methods outlined in the previous section. To evaluate each method trained using the dataset of a particular behavior policy and a given evaluation policy, we first sample 500 (s, a, r, s') tuples from monte carlo rollouts using the evaluation policy on the oracle environment. This setup allows us to evaluate how accurately the offline evaluation methods can estimate the value of the evaluation policy throughout the state-space where the evaluation policy is likely to visit. We compute conditional value estimates using each of the outline methods and compare them against the estimate using monte carlo rollouts using the target evaluation policy on the oracle environment. The latter estimate serves as ground-truth values to compare the offline evaluation methods against. This is done for each possible pair of behavior and evaluation policies.

The results for each pair of behavior and evaluation policies are shown in table 5.5, and summary results across pairs are shown in table 5.6. Empirically, SSW and SUPRW have the lowest mean average error of 9.2 and 9.1, respectively, across pairs of behavior and evaluation policies. Note that SSW and SUPRW outperform both FQE and the model-based method individually, which achieves average errors of 12.7 and 15.0. Table 5.5 also shows that neither SSW and SUPRW are outperformed by both FQE and the model-based method on any pair of behavior and evaluation policy. Half the time, SSW and SUPRW outperform both FQE and the model-based method. Lastly, SSW and SSWPR outperform a simple averaging of FQE and the model-based method, which achieves an average error of 10.6.

We visualize individual predictions of SSW, SSWPR, FQE, and the model-based method against the target values in Figures 5.9 to 5.16. Qualitatively, we see that in cases where the model-based method and FQE are biased in opposite directions, SSW and SSWPR tend to outperform both FQE and the model-based method, exemplified by Figure 5.12 and Figure 5.15. We also see that SSW and SSWPR tend to reduce the extremity of outlier values produced by either the model-based method or FQE, shown by Figure 5.12, 5.13, and 5.15. In the case that FQE and the model-based method are biased in the same direction, SSW and SSWPR tend to have less utility as shown in Figure 5.14. We additionally include histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method in Figures 5.17 to 5.24. Visually, we see that SSW and SSW tend to produce less extreme absolute errors compared with FQE and the model-based method as shown in Figure 5.20, 5.22, and 5.24. Both stability methods give lower errors in many cases unlike FQE as shown in Figure 5.20, and 5.22, 5.24.

We note that SSW and SSWPR have similar errors on average. We visualize the predictions in SSW and SSWPR in Figures 5.25 and 5.26. We see that the difference in predictions between SSW and SSWPR are small across most of the experiments on the environment. The experiments using behavior policy π_{e_3} tend to produce the most variation in predictions between SSW and SSWPR which imply there are cases where one can outperform the other, as shown in Table 5.9.

Stability of results on human decisions

We additionally test the stability of the results of SSW and SSWPR against ad-hoc decisions for the parameters. The first involves perturbing the soft weighting normalization outlined previously. Note that the following equation contains a human decision of using the 0.2 and 0.8 quantiles of the distribution of uncertainties stemming from the behavior dataset:

$$u_{FQE}(s,a) = \frac{std_{FQE}(s,a) - quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, 0.2)}{quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, 0.8) - quantile_{\mathbf{D}_{\pi_b}}(std_{FQE}, 0.2)}.$$

We check whether the results are robust across perturbations of the quantiles used. On this end we use two pairs of values $(0.15, 0.85)$ and $(0.25, 0.75)$ and rerun the SSW. Results are shown in table 5.7 and summarized in table 5.8.

We next check the stability of the SSWPR method against the decision of using max horizon length of 5 when creating partial rollouts. We use values horizon lengths of 3 and 7 and rerun SSWPR. Results are shown in table 5.7 and summarized in table 5.8. Note that on average deviations from the original settings of SSW and SUPRW are small, suggesting that the two methods are robust to these parameters. We even see that, on average, both perturbations to SSWPR perform marginally than the original SSWPR (8.9/9.1 vs 9.2 average errors). Thus, the results in table 5.5 and 5.6 do not seem to rely on the particular choices of parameters used.

Results on the mountain car task

As an additional check on SSW and SSWPR, we rerun experiments on a separate environment. We include the mountain car task as additional experiments [82]. Note that the set up and hyperparameters are kept the same mostly the same as the experiments on the 2D world environment. One difference is the stopping condition for FQE was changed to be

$$|TD_{i-5,i} - TD_{i-6,i-1}| < 0.00025,$$

instead of using 0.001 like in 2DWorld. The behavior policies π_{b_1} and π_{b_2} were trained and achieve mean values of -82.5 and -82.8, respecetively. The evaluation policies π_{e_1}, π_{e_2}, and π_{e_3} achieved mean values of -95.1, -74.5, and -75.9, respectively. The summary of results for each environment are summarized in tables 5.9 and 5.10. From Table 5.10, we see that SSW and SSWPR are able to outperform both FQE, the model-based method, and a simple average of FQE and the model-based method on average. However, there are cases where the model-based method outperforms SSW and SSWPR on individual rows of Table 5.9, which show that stability weighting may not always outperform either method.

Similar to the 2DWorld task, we visualize individual predictions of SSW, SSWPR, FQE, and the model-based method against the target values for the mountain car task in Figures 5.27 to 5.34 and include histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method in Figures 5.35 to 5.42. Like the results on the 2DWorld task, SSW and SSWPR tend to reduce the extremity of outlier estimates produced by the model-based method or FQE on average. We also include scatterplots comparing predictions between SSW and SSWPR for the mountain car task in Figures 5.43 and 5.44 which show that the predictions between the two methods are very similar. However, there are cases where moderate variation exist between two methods as shown in parts C and D in Figures 5.43 and 5.44 which imply one may outperform over the other.

DISCUSSION

We have investigated using stability across an ensemble of neural networks with different initializations as a weighting mechanism in the setting of offline evaluation of reinforcement learning agents and have proposed two methods, SSW and SSWPR, to incorporate stability for adaptively combining conditional model estimates. Our methods provide a positive signal for when a particular method is suited given a local region of the state space. The guiding principle behind the two methods is that if a model's prediction is unstable then its predictive estimate should not be trusted. This principle is powerful in the offline setting where it is unclear how to validate a model's prediction and compare against other models. These methods are particularly valuable in the offline evaluation setting, because a variety of algorithms exist for providing value estimates while model selection is an open problem.

We test our methods on three simulated environments across combinations of different behavior and evaluation policies. By leveraging stability values stemming from model ensembling, we are able to outperform one of FQE and the model-based method every time and both FQE and the model-based half the time. Our experiments suggest that using stability can provide improvements when used to combine estimates of different evaluation algorithms.

SSW and SSWPR are related to the idea of CLEP ensembling [5] in weighted online learning. CLEP produces a weighted average of predictions from individual time series models where the weight of a model's prediction given a set of features is based on the recent performance of the predictor on past data. Unlike the covid-19 forecasting setting, our offline evaluation models do not have comparable performance metrics due to the distributional shift between the behavior and evaluation policies. Thus, we rely on the stability of the model to extract a signal about local predictability. Our experiments show that stability is empirically correlated with the error of the model.

We note that both SSW and SSWPR are related to the idea of perturbation intervals outlined in the PCS framework which quantify the stability of target estimates to perturbations [94]. In our setting, the perturbations are at the data and model level, where randomization of the model initialization is used as perturbations. The notion underlying both methods are that the resulting variability in estimates outline regions of the state space which models are stable and unstable. The key assumption is that stability to such perturbations can help to identify in which scenarios a certain model or method is more reliable than another.

Our proposed methods SSW and SSWPR are also related to the pessimism principle studied in offline reinforcement policy learning. The main notion of the pessimism principle leveraged in these works is that areas of the state-space where the Q-function or dynamics model is uncertain or unstable should be avoided due to insufficient coverage of the behavior policy. Kidambi et al. [39] incorporate pessimism into model-based reinforcement learning and construct a pessimistic markov decision process using an ensemble of learned dynamics models which partitions the state space into known and unknown regions and artificially penalizes an agent with a negative reward for visiting unknown regions. Kumar et al. [46] train policies by maximizing the most conservative estimate from an ensemble of Q-functions as well constraining the policy to the support of the behavior policy

$$\max_{\pi \in \prod_{\epsilon}} \mathbb{E}_{a \sim \pi(.|s)} \left[\min_{j=1 \ldots K} \hat{Q}_j(s,a) \right],$$

where $\prod_{\epsilon}$ is the set of policies sharing the support of the behavior policy. We apply similar reasoning to the offline evaluation setting by using the stability of model ensembles to weight offline estimates.

Our study has a few limitations that should be mentioned. First, we have only provided empirical evidence for using stability as a model weighting mechanism across a few simulated environments. Further experimentation and testing needs to be done to better understand the soft weighting mechanism and under which cases the soft weighting is likely to lead to significant improvements in performance over baselines as well as benchmarking on more complex environments. Our experiments show that if the bias of FQE and the model-based method are in opposite directions, improvements over both FQE and the model-based method are likely. Additionally, our weighting methods outperform a simple average of FQE and the model-based method, which show the utility in using stability as a weighting mechanism over naive averaging. A second limitation is that our methods are based on parameters including the weighting normalization for SSW and the partial horizon length for SSWPR which may require tuning based on the specific application. Although our experiments shows that SSW and SSWPR are robust to perturbations of the values for the parameters used in the environments we test on, more experimentation should be done. Lastly, we use perturbations on the model level via different randomization of neural network weights to evaluate stability. However, other forms of perturbations can and should be experimented, including those at the data level, which can be challenging in the reinforcement learning setting. One future direction is to include bootstrapped data as an avenue of assessing stability.

CONCLUSION

Model selection and validation is a difficult problem in the offline reinforcement learning setting due to the lack of access to the environment for data collection. We propose using stability of evaluation functions as a weighting mechanism inspired by ideas from weighted online learning when typical validation is not possible. Our proposed methods SSW and SSWPR, which leverage stability via ensembling, have shown to improvement offline estimates from FQE and the model-based method, potentially increasing the viability of reinforcement learning to applications like healthcare where safety is paramount.

TABLES

Table 5.1. Summary statistics for each offline dataset corresponding to behavior policies π_{b1} and π_{b2} on the 2DWorld environment

Statistic	Dataset D_{b1}	Dataset D_{b2}
Dataset size	160,745	156,253
Mean steps per episode	159.7	155.3
Mean reward per episode	-159.9	-177.7
Proportion of time collecting -1 rewards	0.97	0.83
Proportion of time collecting -2 rewards	0.02	0.14
Proportion of time collecting -4 rewards	0.0	0.02

Table 5.2. Errors for FQE and the model-based (MB) method on partition A (sampled tuples from the evaluation policy where FQE outperforms the MB method) and partition B

(sampled tuples from the evaluation policy where the MB method outperforms FQE) using models trained on data corresponding to the behavior policy π_{b_1}) and the 2DWorld environment

Evaluation policy	FQE error on partition A	Model-based error on partition A	FQE error on partition B	Model-based error on partition B
π_{b_1}	3.7	6.5	9.3	3.4
π_{e_1}	NA	NA	11.6	2.8
π_{e_2}	8.0	13.4	16.3	4.9
π_{e_3}	19.1	83.5	18.0	5.3

Table 5.3. Errors for FQE and the model-based (MB) method on partition A (sampled tuples from the evaluation policy where FQE outperforms the MB method) and partition B (sampled tuples from the evaluation policy where the MB method outperforms FQE) using models trained on data corresponding to the behavior policy π_{b_2}) and the 2DWorld environment

Evaluation policy	FQE error on partition A	Model-based error on partition A	FQE error on partition B	Model-based error on partition B
π_{b_2}	5.9	27.0	19.9	7.7
π_{e_1}	3.2	4.8	36.0	8.6
π_{e_2}	7.8	18.0	18.5	7.5
π_{e_3}	12.1	49.4	7.5	14.0

Table 5.4. Proportion of times method with the higher stability out of FQE and the model-based method had the higher error across pairs of behavior and evaluation policies using the 2DWorld environment

Behavior policy	Evaluation policy	Proportion
π_{b_1}	π_{e_1}	0.65
π_{b_1}	π_{e_2}	0.63
π_{b_1}	π_{e_3}	0.62
π_{b_1}	π_{e_4}	0.77
π_{b_2}	π_{e_1}	0.63
π_{b_2}	π_{e_2}	0.59
π_{b_2}	π_{e_3}	0.74
π_{b_2}	π_{e_4}	0.49

Table 5.5. Mean absolute error for conditional value estimates from (s, a, r, s') tuples from monte carlo rollouts of the evaluation policy on the 2DWorld environment for proposed adaptive stability weighting methods and baseline methods)

Evaluation policy	Behavior policy	FQE	Model-based	FQE and model-based average	Oracle non-adaptive selection	SSW	SSWPR
π_{b_1}	π_{b_1}	5.9	3.7	4.0	3.7	4.4	4.1
π_{b_1}	π_{e_1}	5.0	3.8	3.2	3.8	3.5	3.9
π_{b_1}	π_{e_2}	13.3	8.7	9.3	8.7	8.6	8.6
π_{b_1}	π_{e_3}	18.0	51.1	21.4	18.0	11.8	11.9
π_{b_2}	π_{b_2}	10.5	5.4	7.4	5.1	6.3	6.0
π_{b_2}	π_{e_1}	20.1	8.1	14.1	8.1	8.6	8.6
π_{b_2}	π_{e_2}	11.5	8.1	14.1	8.1	13.5	14.6
π_{b_2}	π_{e_3}	17.1	31.1	17.3	17.1	17.8	16.1

Table 5.6. Error summaries for conditional value estimates from (s, a, r, s') tuples from monte carlo rollouts of the evaluation policy on the 2DWorld environment for proposed adaptive stability weighting methods and baseline methods

Method	Average absolute error	# times best performing	# times worst performing	# times outperforming
FQE	12.7	0	6	NA
Model-based	15.1	3	2	NA
Average FQE and model-based	10.6	1	0	2
SSW	9.2	3	0	4
SSWPR	9.2	2	0	3

Table 5.7. Mean absolute error for conditional value estimates from (s, a, r, s') tuples from monte carlo rollouts of the evaluation policy on the 2DWorld environment for perturbations on proposed adaptive stability weighting methods)

Evaluation policy	Behavior policy	SSW	SSW 0.15/0.85	SSW 0.25/0.75	SSWPR	SSWPR 3	SSWPR 7
π_{b_1}	π_{b_1}	4.4	4.4	4.9	4.1	4.2	4.4
π_{b_1}	π_{e_1}	3.5	3.4	3.6	3.9	4.0	3.9
π_{b_1}	π_{e_2}	8.6	8.6	8.5	8.6	8.7	8.6
π_{b_1}	π_{e_3}	11.8	11.0	12.7	11.9	12.1	12.6
π_{b_2}	π_{b_2}	6.3	6.3	6.3	6.0	6.1	5.6
π_{b_2}	π_{e_1}	13.5	14.5	13.4	14.6	13.3	13.2
π_{b_2}	π_{e_2}	7.9	8.5	7.8	8.0	7.4	7.3
π_{b_2}	π_{e_3}	17.8	17.6	18.2	16.1	16.8	15.8

Table 5.8. Error summaries for conditional value estimates from (s, a, r, s') tuples from monte carlo rollouts of the evaluation policy on the 2DWorld environment for perturbations on adaptive stability weighting methods and baseline methods

Method	Average absolute error
SSW	9.2
SSW 0.15/0.85	9.3
SSW 0.25/0.75	9.6
SSWPR	9.2
SSWPR 5	9.1
SSWPR 7	8.9

Table 5.9. Mean absolute error for conditional value estimates from (s, a, r, s') tuples from monte carlo rollouts of the evaluation policy on the mountain car task for proposed adaptive stability weighting methods and baseline methods)

Evaluation policy	Behavior policy	FQE	Model-based	FQE and model-based average	Oracle non-adaptive selection	SSW	SSWPR
π_{b_1}	π_{b_1}	6.8	9.6	7.6	6.8	5.6	5.4
π_{b_1}	π_{e_1}	14.0	3.1	5.9	3.1	3.5	3.5
π_{b_1}	π_{e_2}	7.4	5.0	5.9	5.0	5.6	5.5
π_{b_1}	π_{e_3}	12.1	5.2	8.2	6.2	5.0	5.9
π_{b_2}	π_{b_2}	6.1	5.0	4.8	5.0	4.8	4.8
π_{b_2}	π_{e_1}	16.3	14.7	15.3	14.7	13.9	12.6
π_{b_2}	π_{e_2}	7.6	7.4	7.3	7.4	7.2	6.7
π_{b_2}	π_{e_3}	15.5	6.8	10.8	6.8	6.6	6.3

Table 5.10. Error summaries for conditional value estimates from (s, a, r, s') tuples from monte carlo rollouts of the evaluation policy on the mountain car task for proposed adaptive stability weighting methods and baseline methods

Method	Average absolute error	# times best performing	# times worst performing	# times out-performing
FQE	10.7	0	7	NA
Model-based	7.1	2	1	NA
Average FQE and model-based	8.2	1	0	2
SSW	6.5	3	0	6
SSWPR	6.3	5	0	5

Figure 5.1: A depiction of online reinforcement learning where s,r,a, and s' is the state, reward, action, and next state and π_k is the policy which is a mapping from a state to an action to be taken by the agent. Rollout data consists of the (s, r, a, s') transitions taken from the policy π_k interacting with the environmenet. π_k is trained interactively with repeated interactions with the environment via a feedback loop

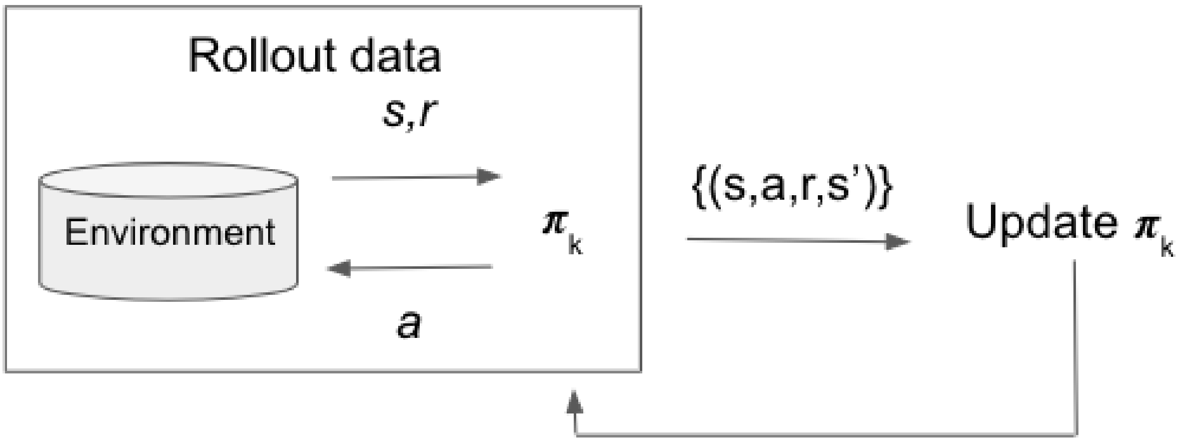


Figure 5.2: A depiction of offline reinforcement learning where s,r,a, and s' is the state, reward, action, and next state, π_B is the behavior policy (the policy that generates the offline data), and π is the policy, which can be either deterministic or stochastic, to be trained. Unlike online reinforcement learning, training π does not involve interacting with the environment.

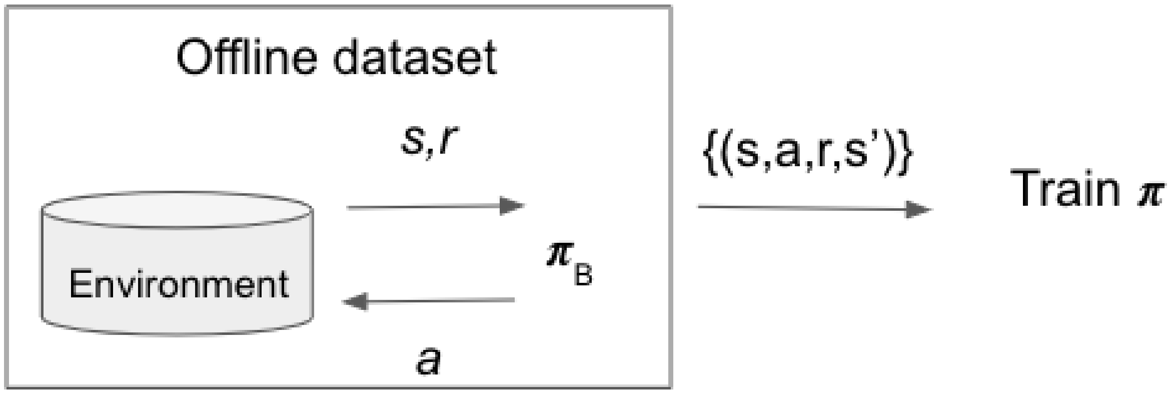


Figure 5.3: 2DWorld: positional depiction. Agent starts at a random position from $[0, 5/6]$ *in both its* x *and* y *positions and receives a negative reward at each time step condi-*

tioned on the subgrid the agent currently subsides. The agent receives a completion reward of +10 if the goal is reached ($[25/6, 5]$ *in both its* x *and* y *positions.)*

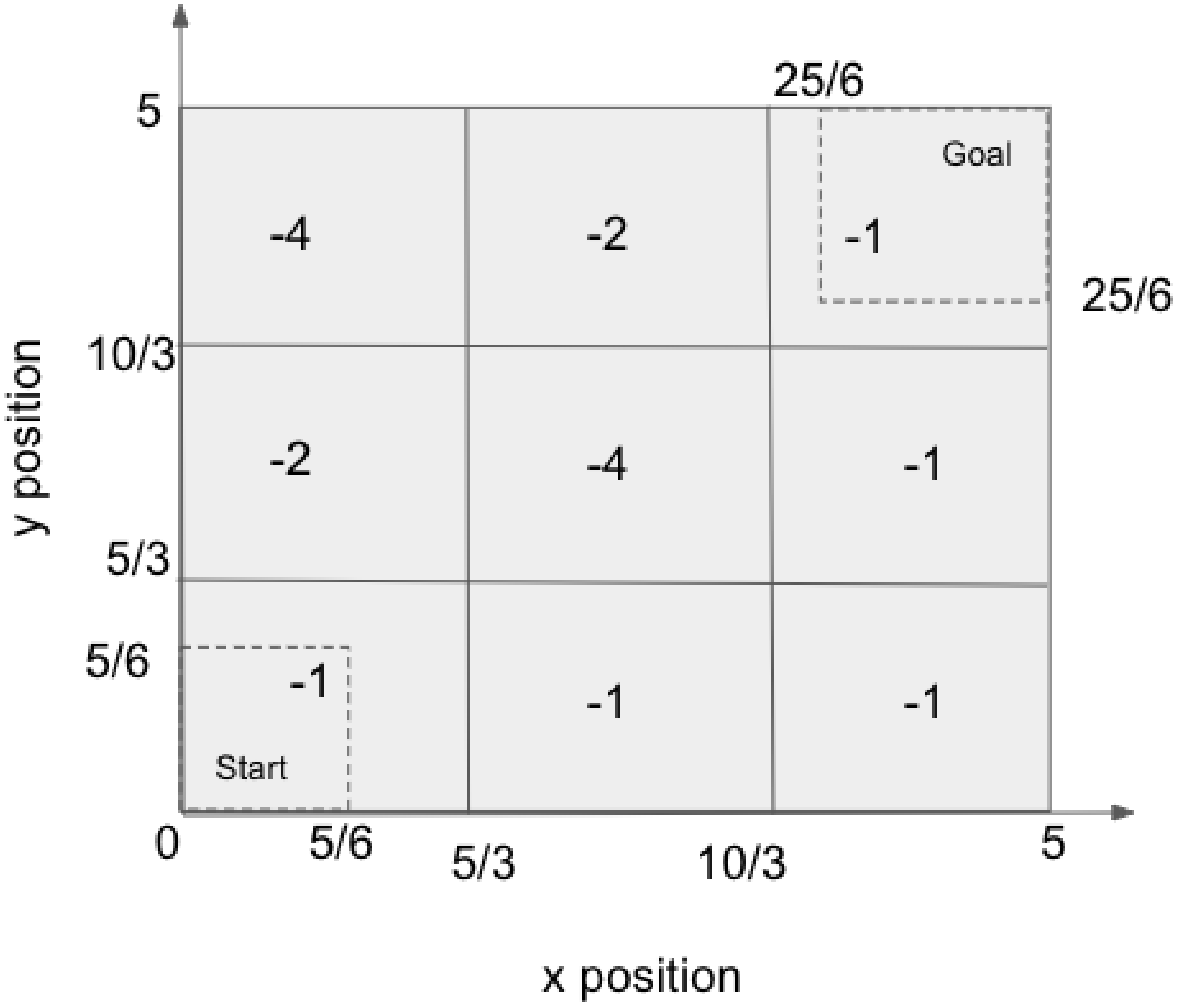


Figure 5.4: Vertical and horizontal positions over time from episodes derived from behavior policies π_{b1} *(A) and* π_{b2} *(B) trained on the 2DWorld environment*

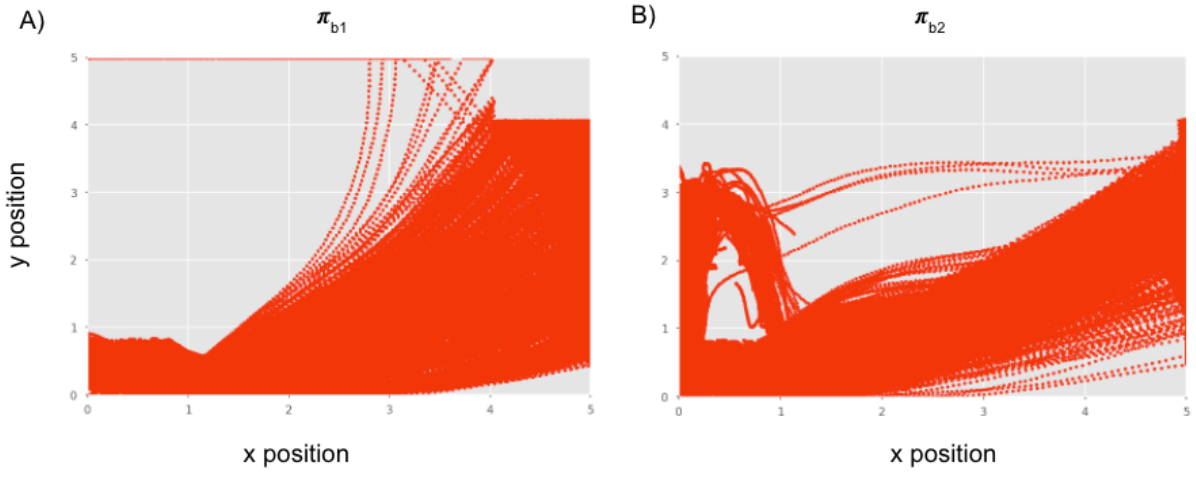


Figure 5.5: Vertical and horizontal positions over time from episodes derived from evaluation policies π_{e1} (A), π_{e2} (B), and π_{e3} (C) trained on the 2DWorld environment

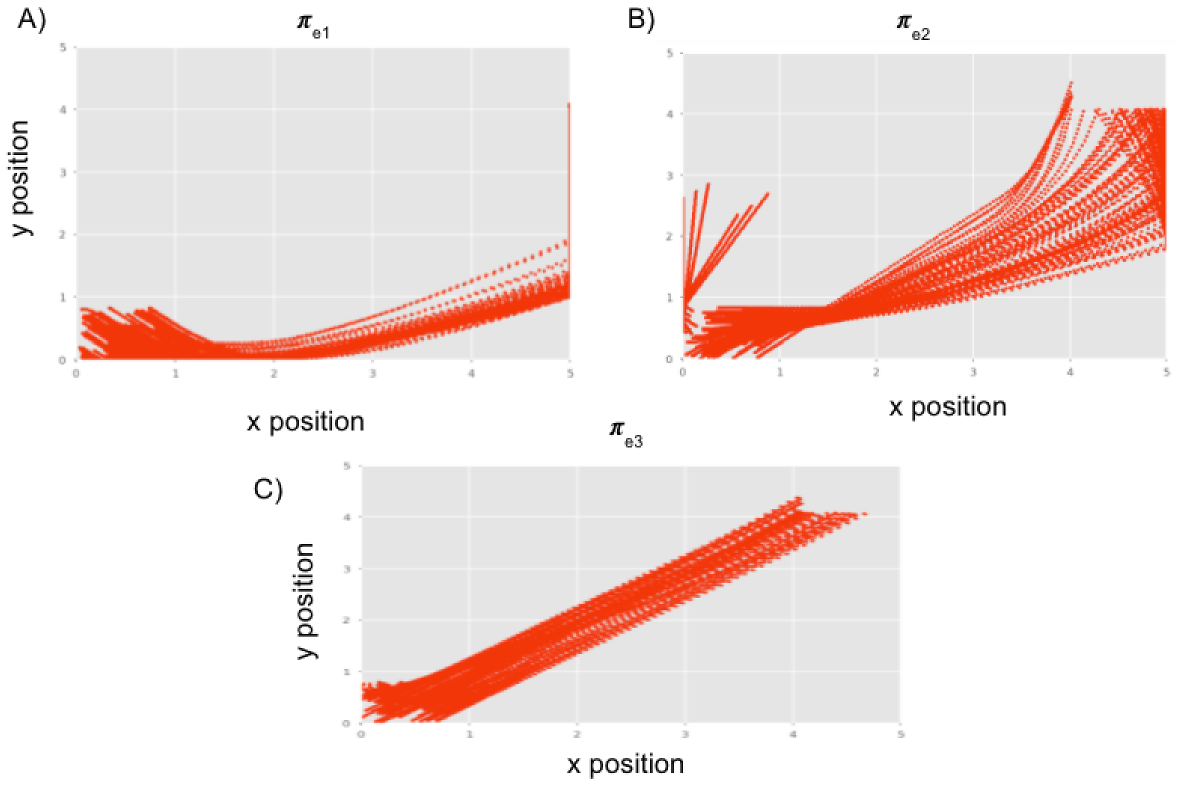


Figure 5.6: Scatterplots of the x and y positions of sampled tuples on the 2DWorld environment colored by the best performing method trained on the behavior dataset corresponding to π_{b_1}. The evaluation policies used include π_{b_1} (A), π_{e_1} (B), π_{e_2} (C), π_{e_3} (D).

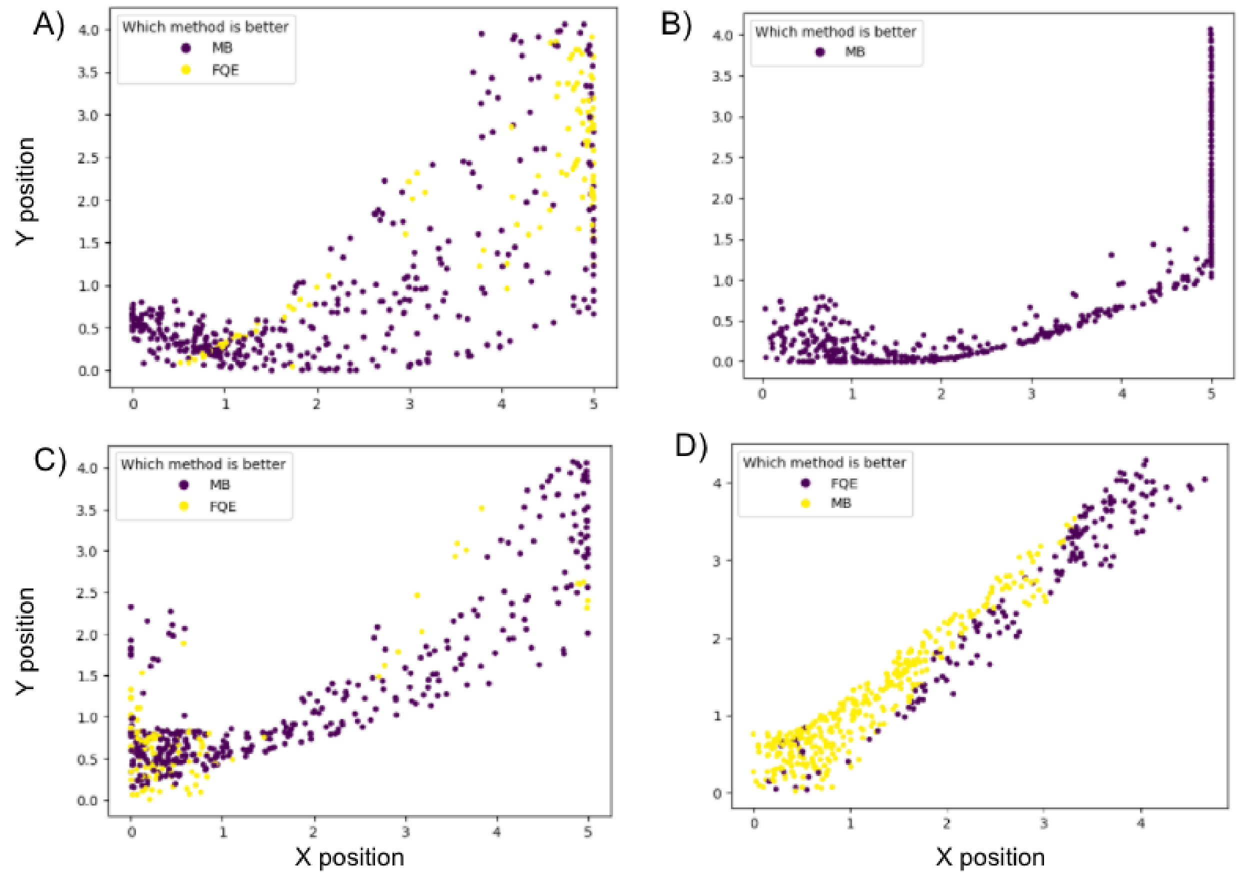


Figure 5.7: Scatterplots of the x and y positions of sampled tuples on the 2DWorld environment colored by the best performing method trained on the behavior dataset corresponding to π_{b_2} The evaluation policies used include π_{b_2} (A), π_{e_1} (B), π_{e_2} (C), π_{e_3} (D).

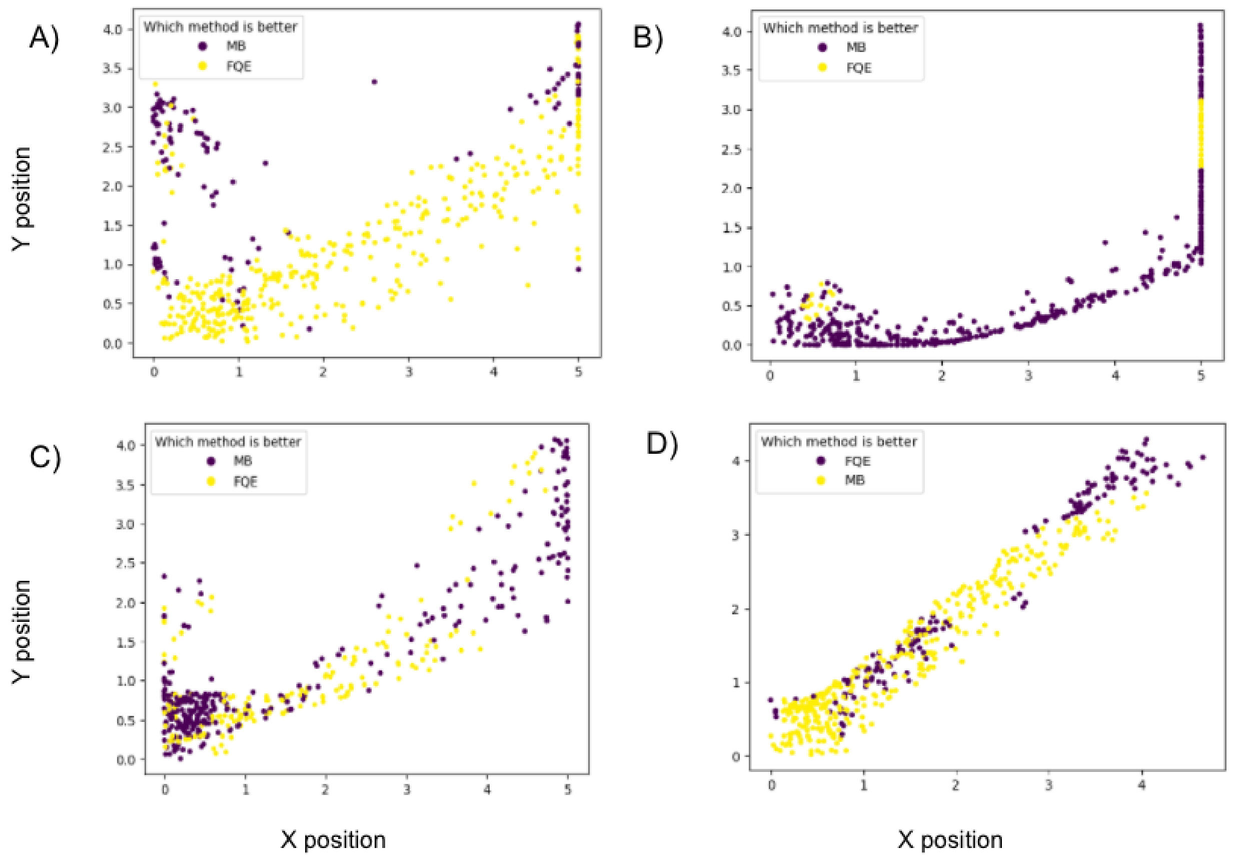
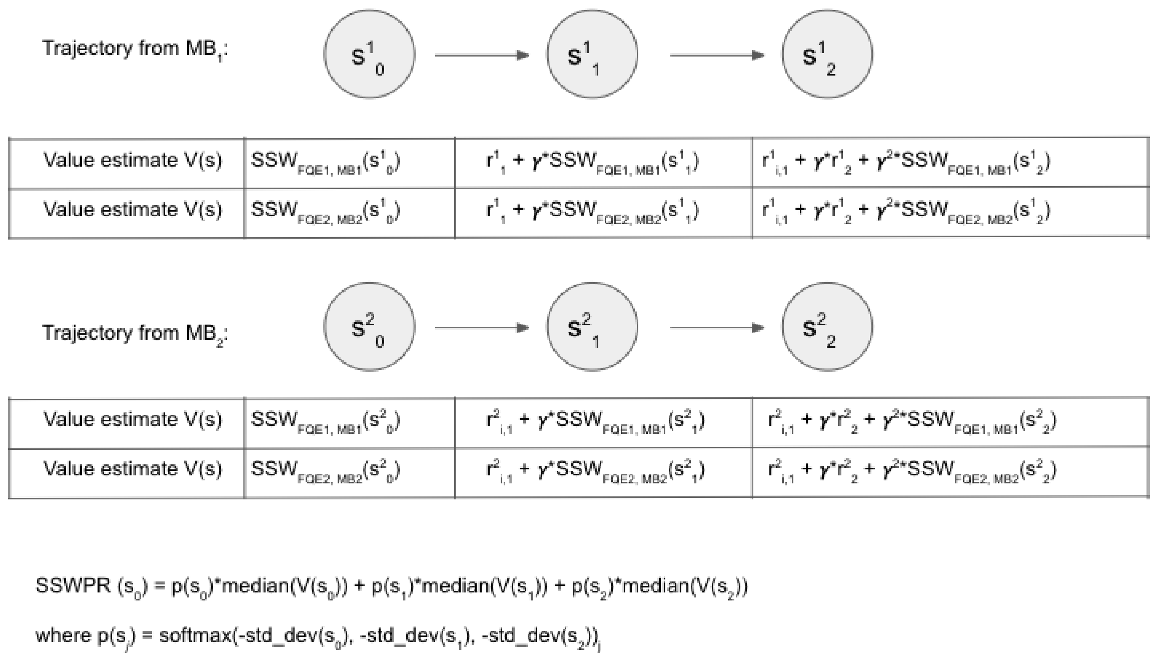


Figure 5.8: A depiction of the SSWPR method with ensembles of size two for FQE and the model-based method and a maximum partial horizon of 3 where std_ dev represents the standard deviation, median($V(s_t)$) represents the median of all value estimates at time t, s_j^i represents the simulated state from dynamics model i at time step j, and r_j^i represents the reward given from dynamics model i at time step j.



Value estimate V(s)	$SSW_{FQE1, MB1}(s^1_0)$	$r^1_1 + \gamma * SSW_{FQE1, MB1}(s^1_1)$	$r^1_{i,1} + \gamma * r^1_2 + \gamma^2 * SSW_{FQE1, MB1}(s^1_2)$
Value estimate V(s)	$SSW_{FQE2, MB2}(s^1_0)$	$r^1_1 + \gamma * SSW_{FQE2, MB2}(s^1_1)$	$r^1_{i,1} + \gamma * r^1_2 + \gamma^2 * SSW_{FQE2, MB2}(s^1_2)$

Value estimate V(s)	$SSW_{FQE1, MB1}(s^2_0)$	$r^2_{i,1} + \gamma * SSW_{FQE1, MB1}(s^2_1)$	$r^2_{i,1} + \gamma * r^2_2 + \gamma^2 * SSW_{FQE1, MB1}(s^2_2)$
Value estimate V(s)	$SSW_{FQE2, MB2}(s^2_0)$	$r^2_{i,1} + \gamma * SSW_{FQE2, MB2}(s^2_1)$	$r^2_{i,1} + \gamma * r^2_2 + \gamma^2 * SSW_{FQE2, MB2}(s^2_2)$

$SSWPR\ (s_0) = p(s_0) * median(V(s_0)) + p(s_1) * median(V(s_1)) + p(s_2) * median(V(s_2))$

where $p(s_j) = softmax(-std\_dev(s_0), -std\_dev(s_1), -std\_dev(s_2))_j$

Figure 5.9: Scatterplots of the predictions and target values for evaluating π_{b_1} on the logged data deriving from π_{b_1} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

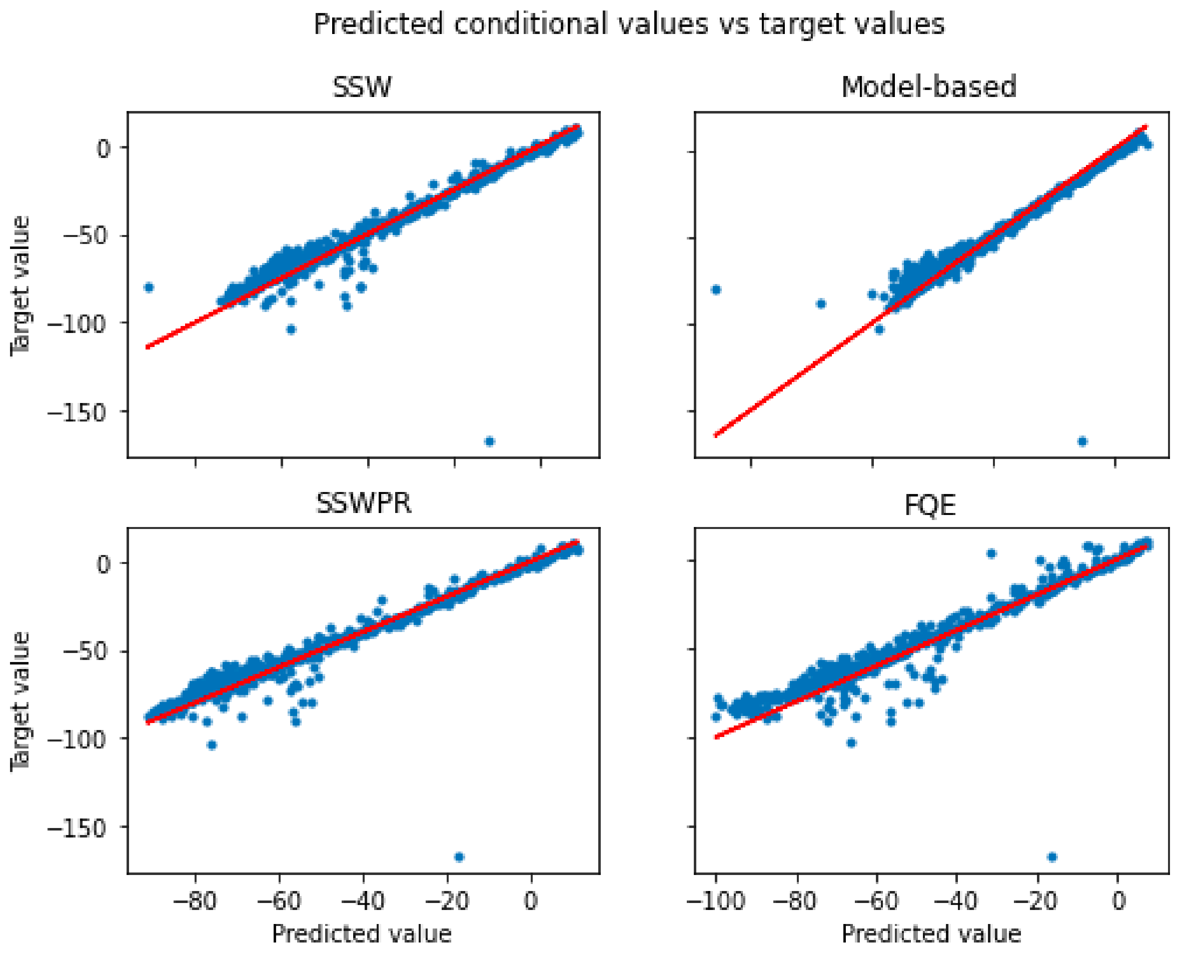


Figure 5.10: Scatterplots of the predictions and target values for evaluating π_{e_1} on the logged data deriving from π_{b_1} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

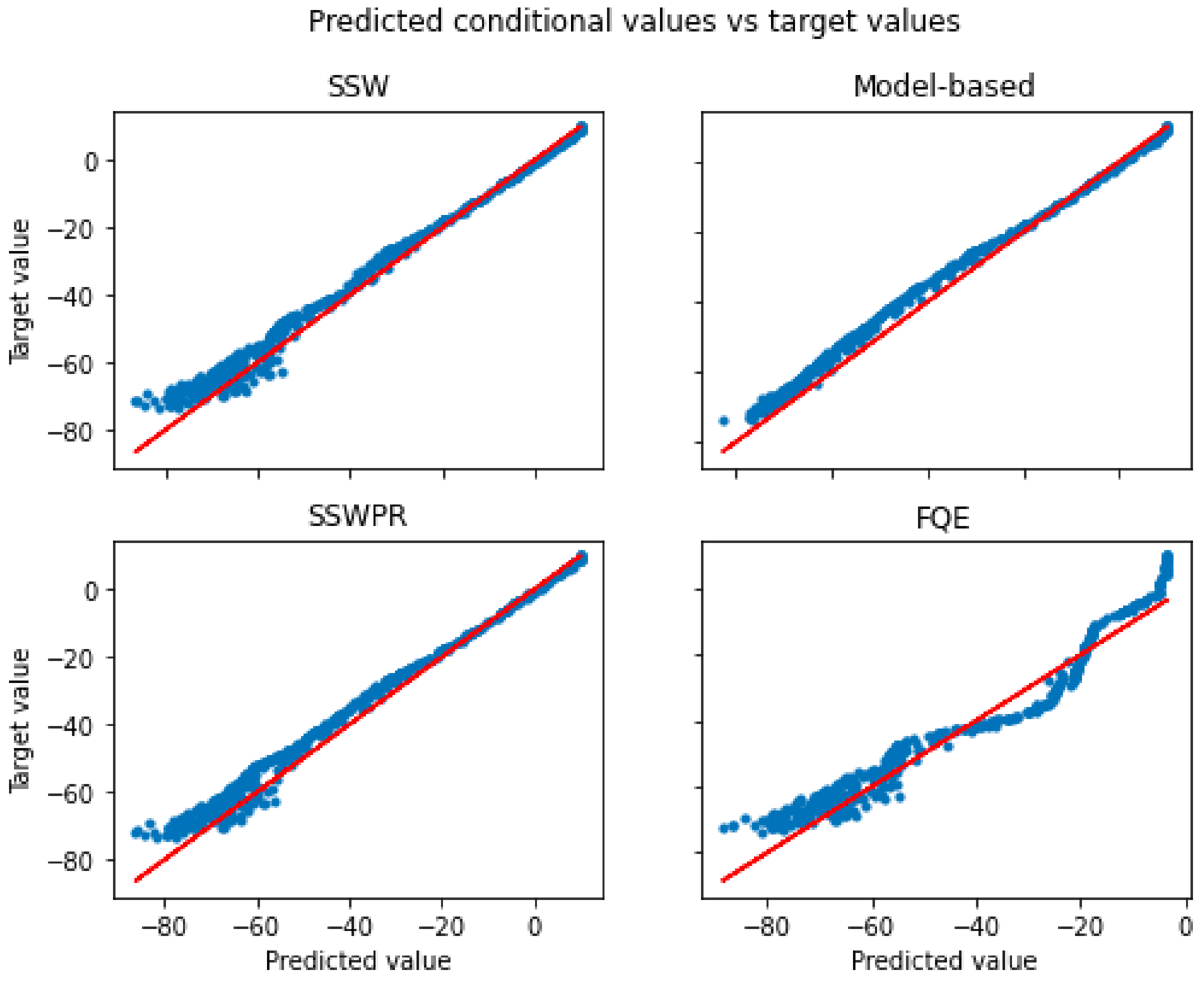


Figure 5.11: Scatterplots of the predictions and target values for evaluating π_{e_2} on the logged data deriving from π_{b_1} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

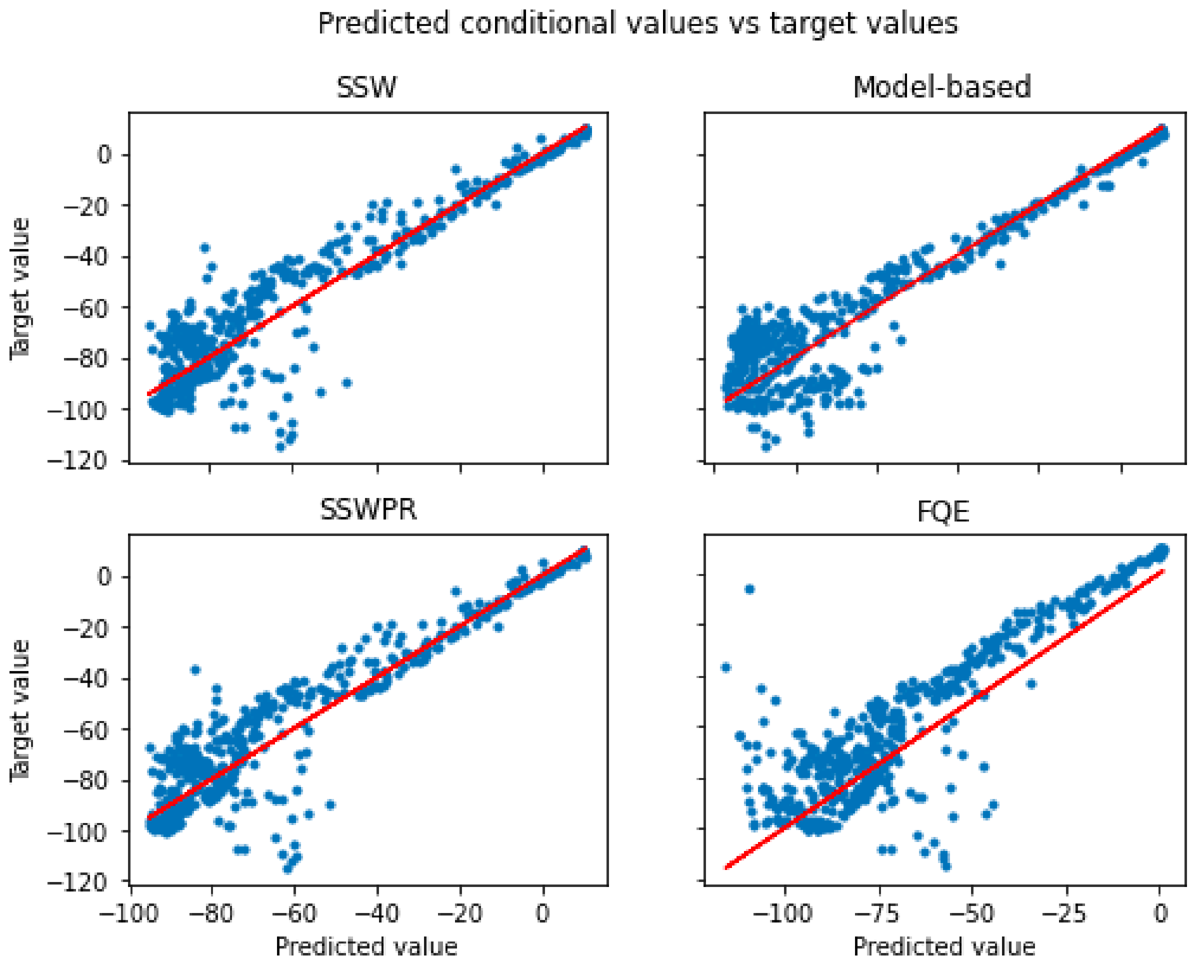


Figure 5.12: Scatterplots of the predictions and target values for evaluating π_{e_3} on the logged data deriving from π_{b_1} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

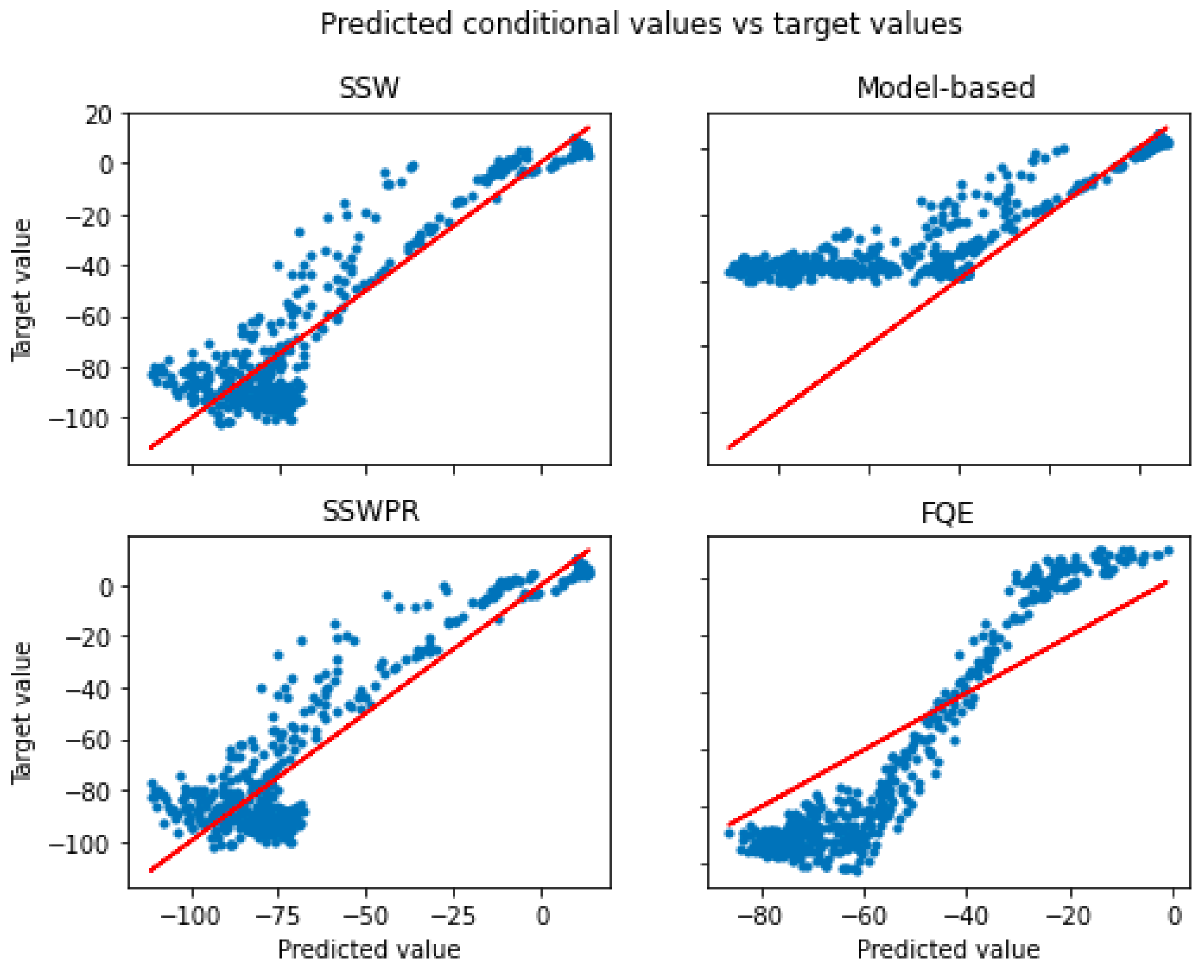


Figure 5.13: Scatterplots of the predictions and target values for evaluating π_{b_2} on the logged data deriving from π_{b_2} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

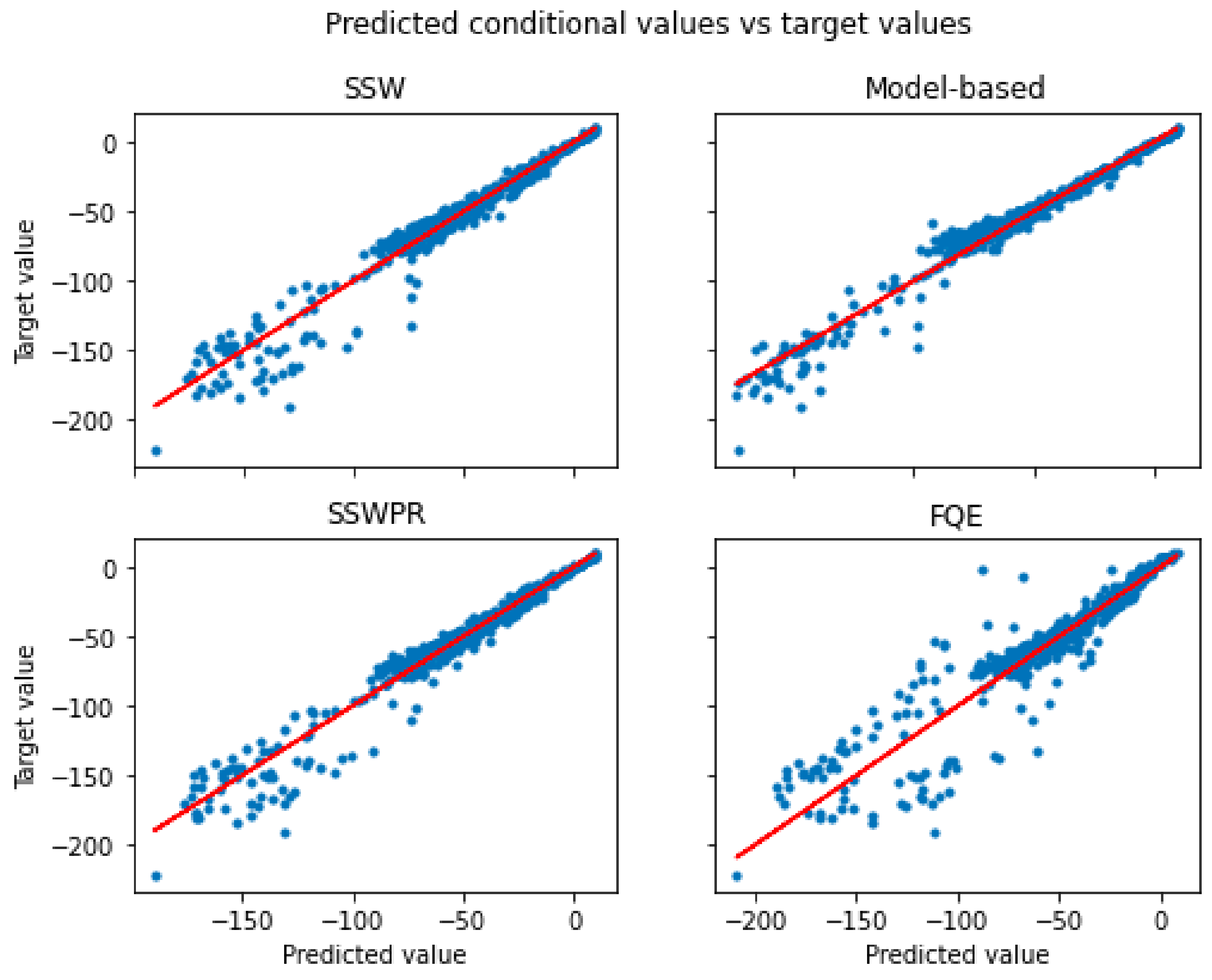


Figure 5.14: Scatterplots of the predictions and target values for evaluating π_{e_1} on the logged data deriving from π_{b_2} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

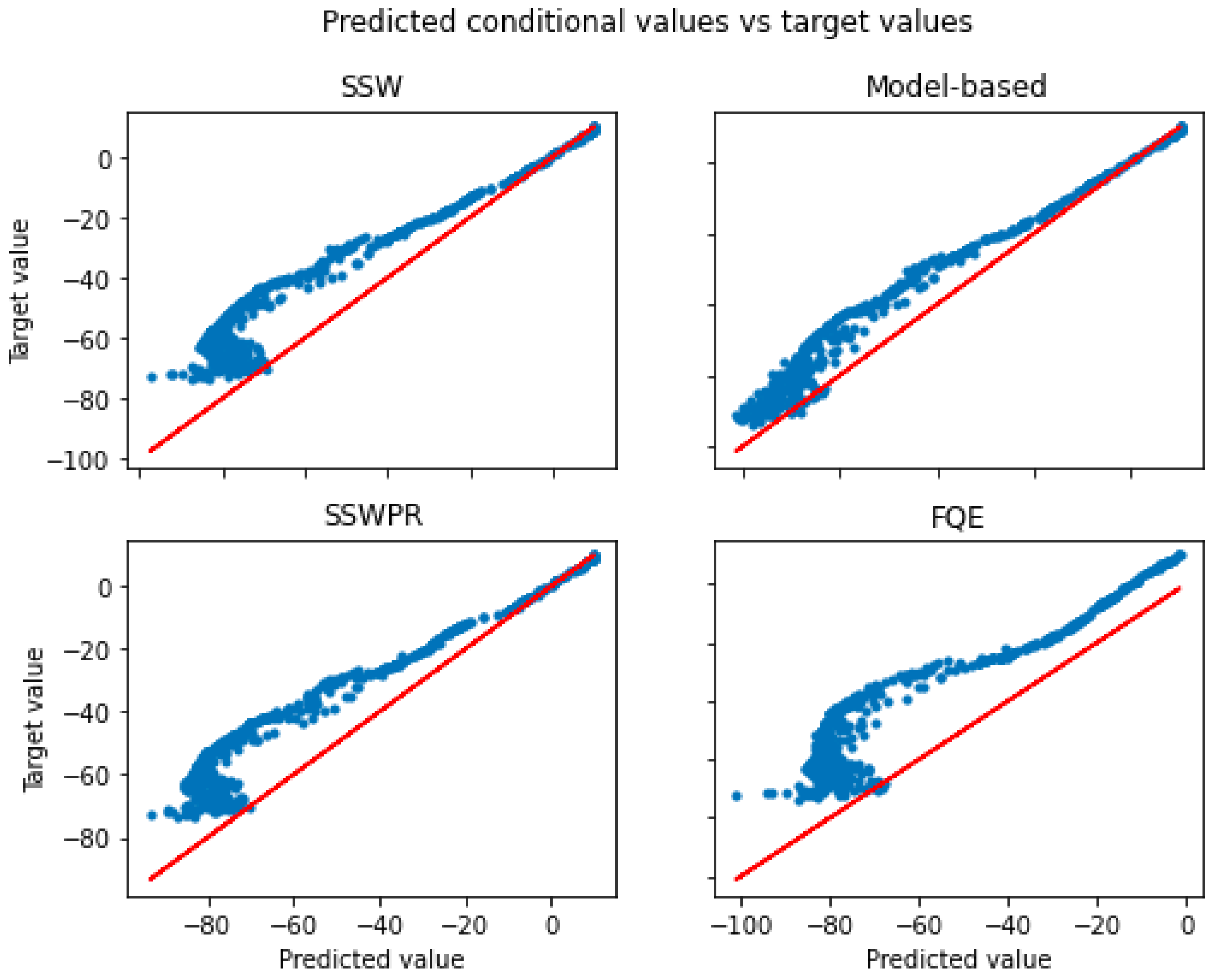


Figure 5.15: Scatterplots of the predictions and target values for evaluating π_{e_2} on the logged data deriving from π_{b_2} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

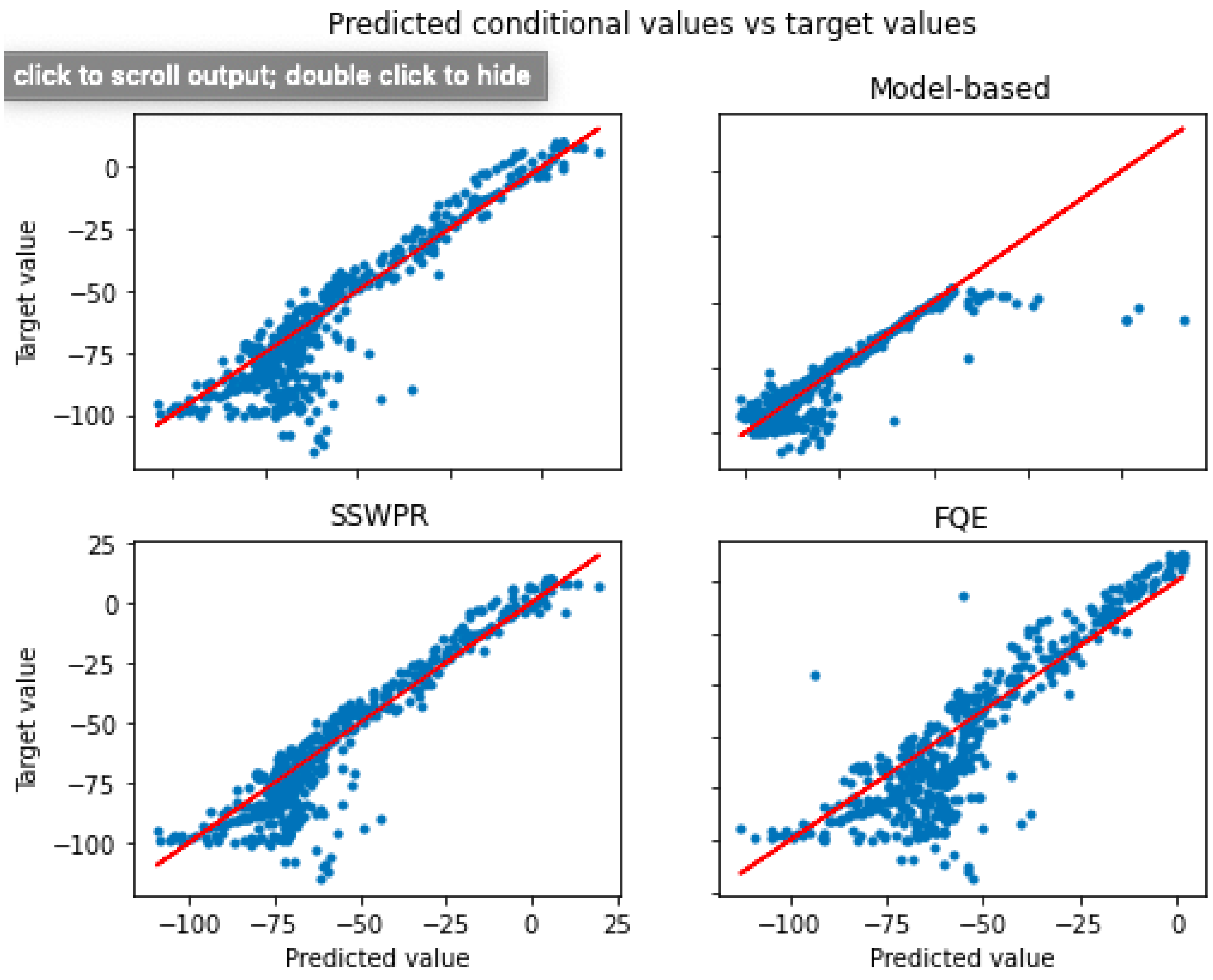


Figure 5.16: Scatterplots of the predictions and target values for evaluating π_{e_3} on the logged data deriving from π_{b_2} on the 2DWorld environment using the model-based method, FQE, SSW, and SSWPR.

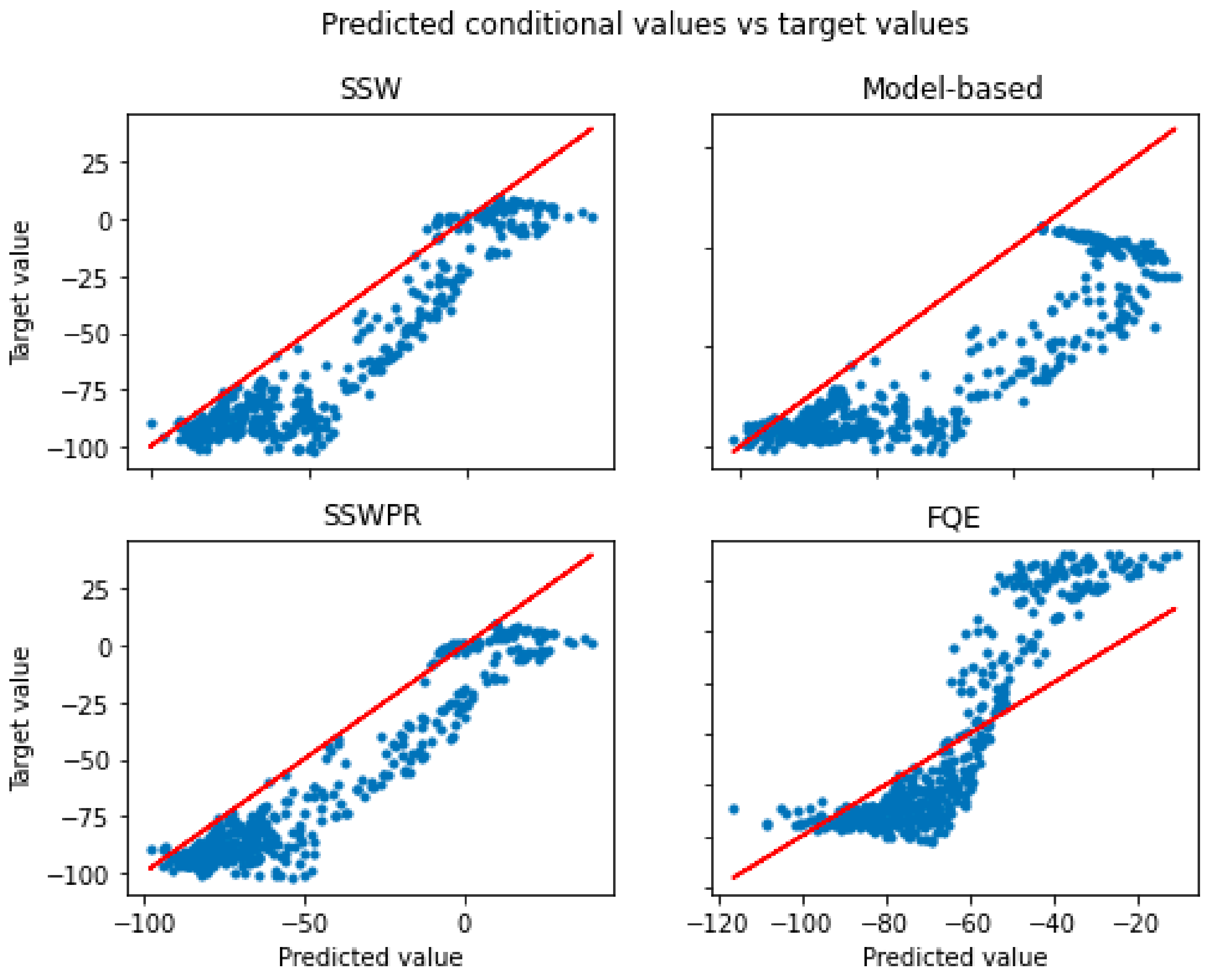


Figure 5.17: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{b_1} on the logged data deriving from π_{b_1} on the 2DWorld environment

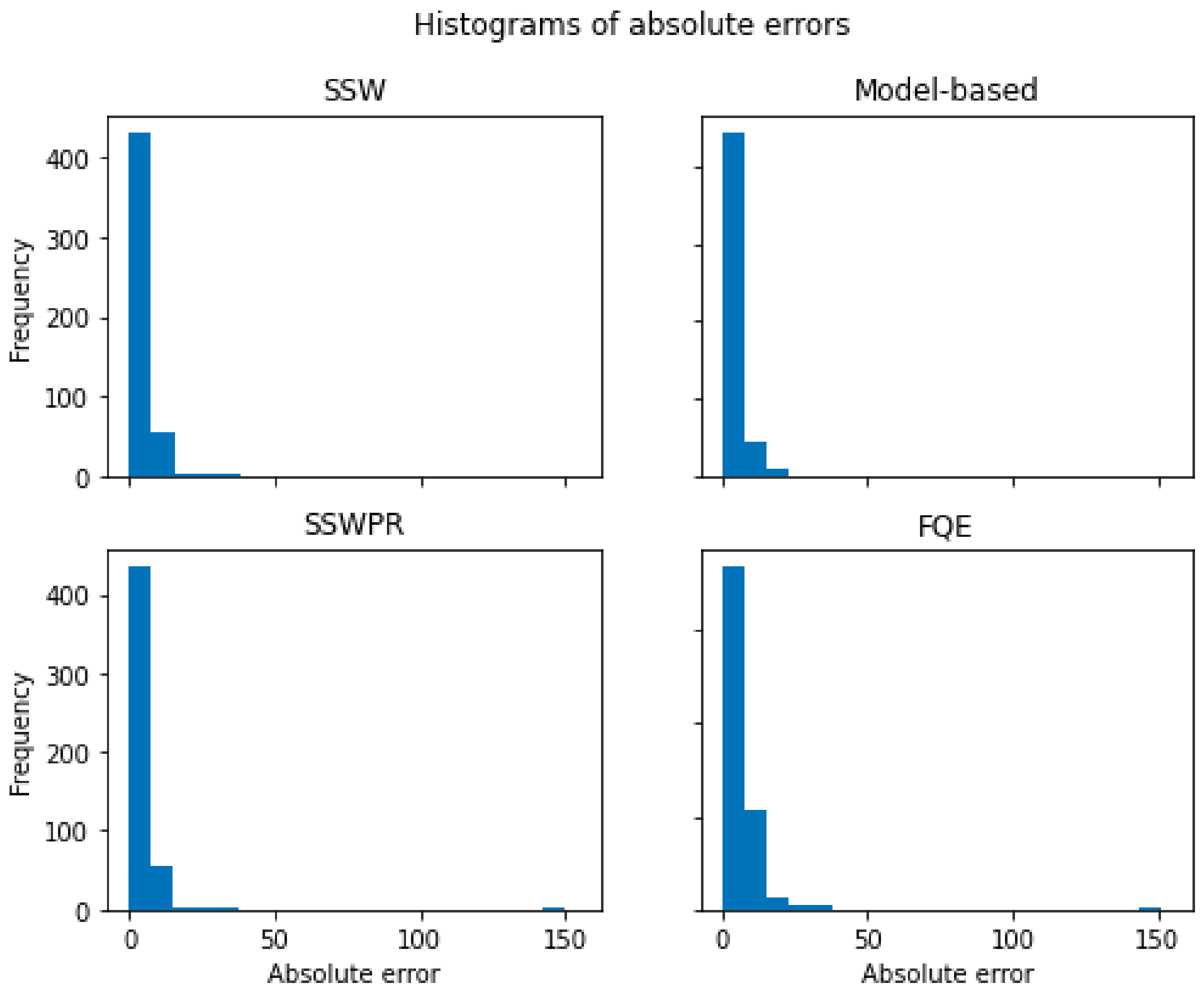


Figure 5.18: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_1} on the logged data deriving from π_{b_1} on the 2DWorld environment

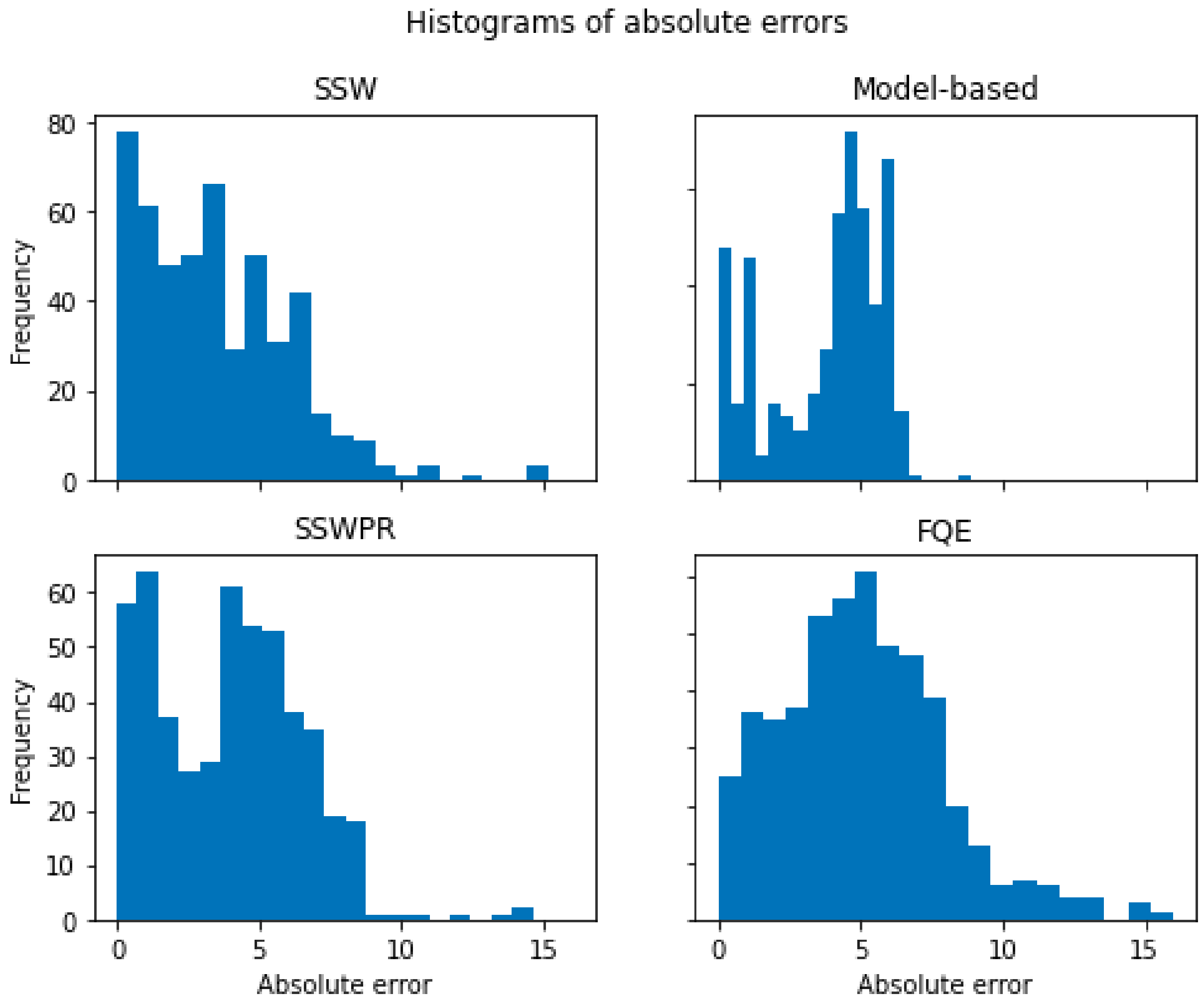


Figure 5.19: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_2} on the logged data deriving from π_{b_1} on the 2DWorld environment

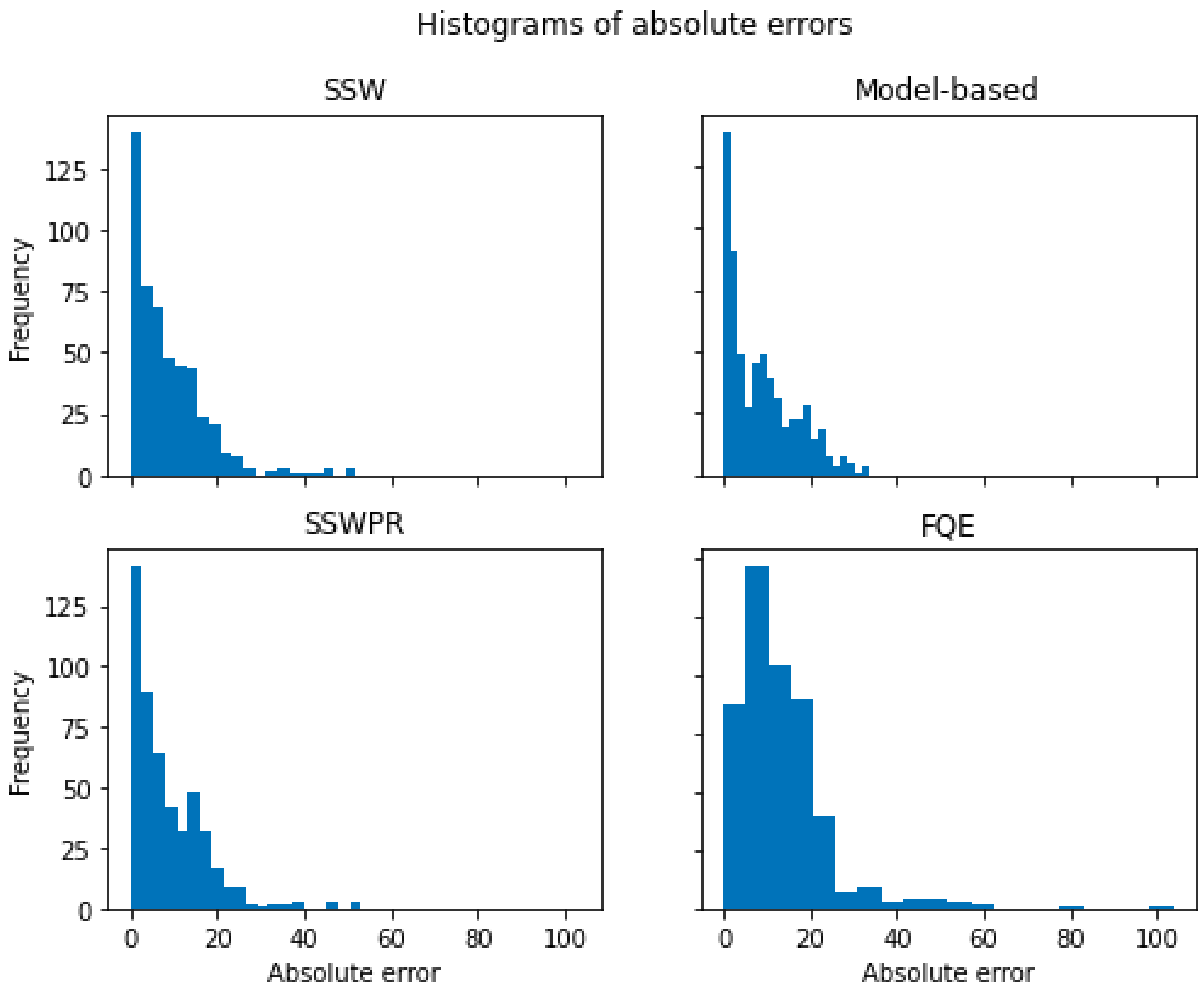


Figure 5.20: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_3} on the logged data deriving from π_{b_1} on the 2DWorld environment

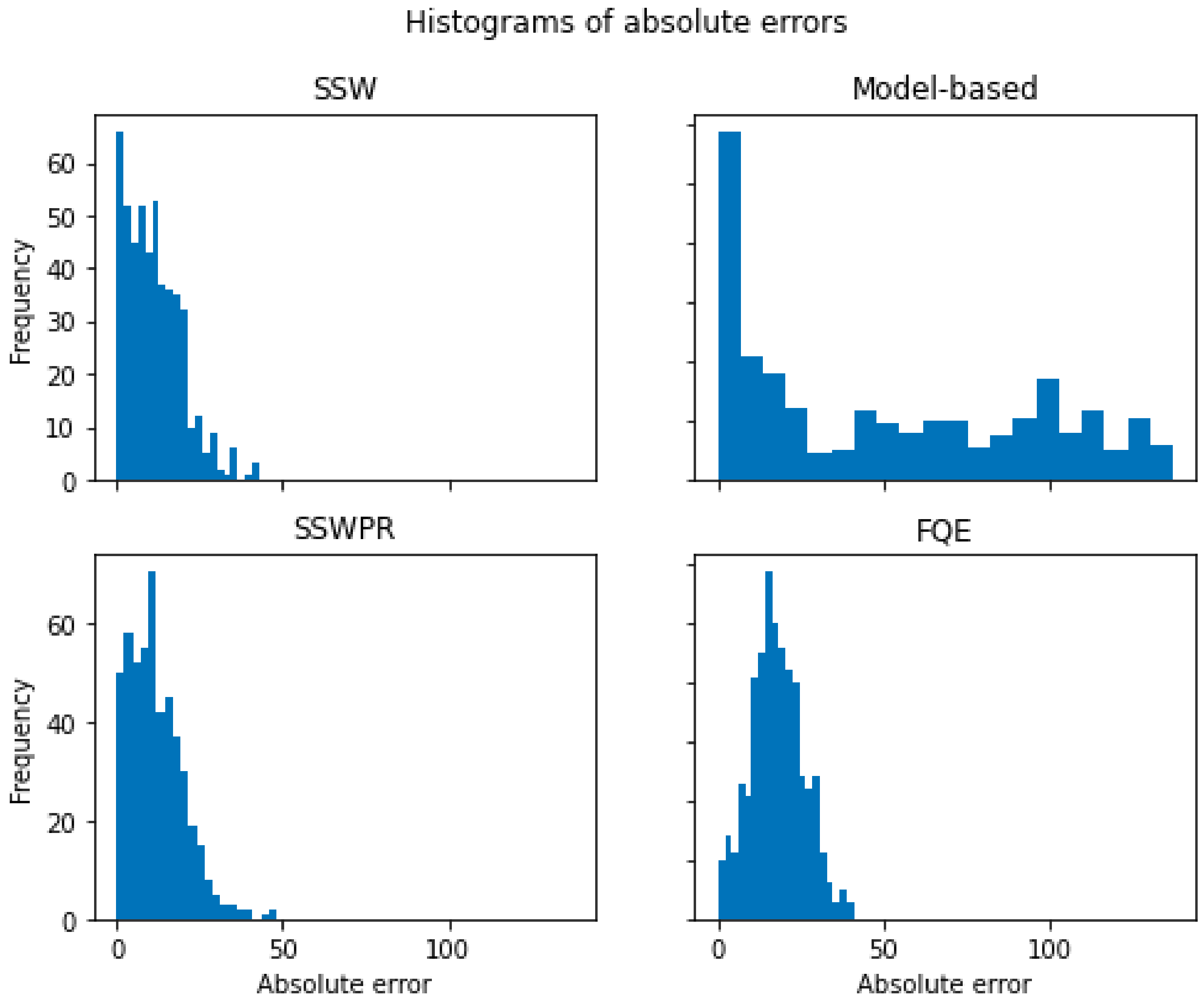


Figure 5.21: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{b_2} on the logged data deriving from π_{b_2} on the 2DWorld environment

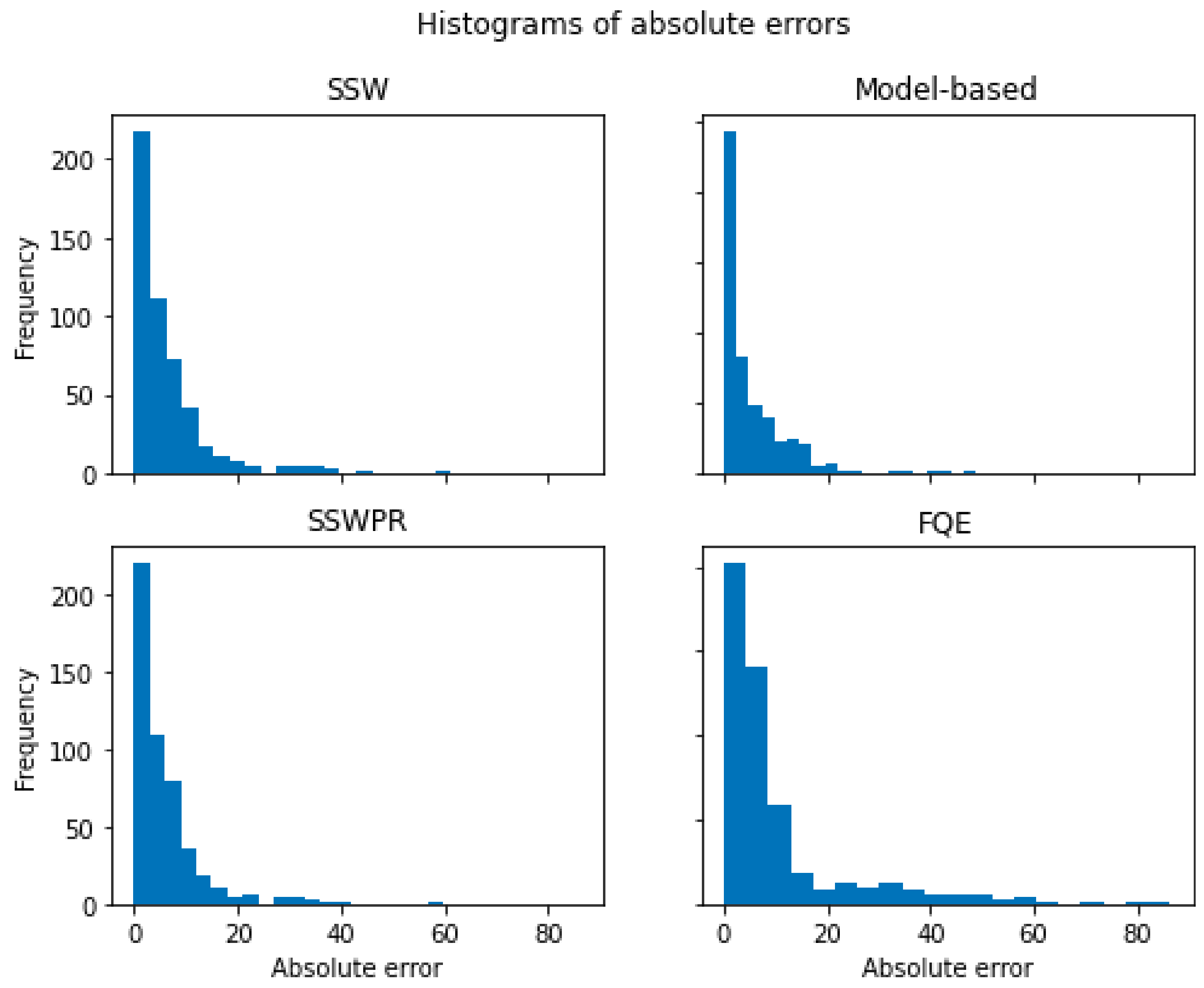


Figure 5.22: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_1} on the logged data deriving from π_{b_2} on the 2DWorld environment

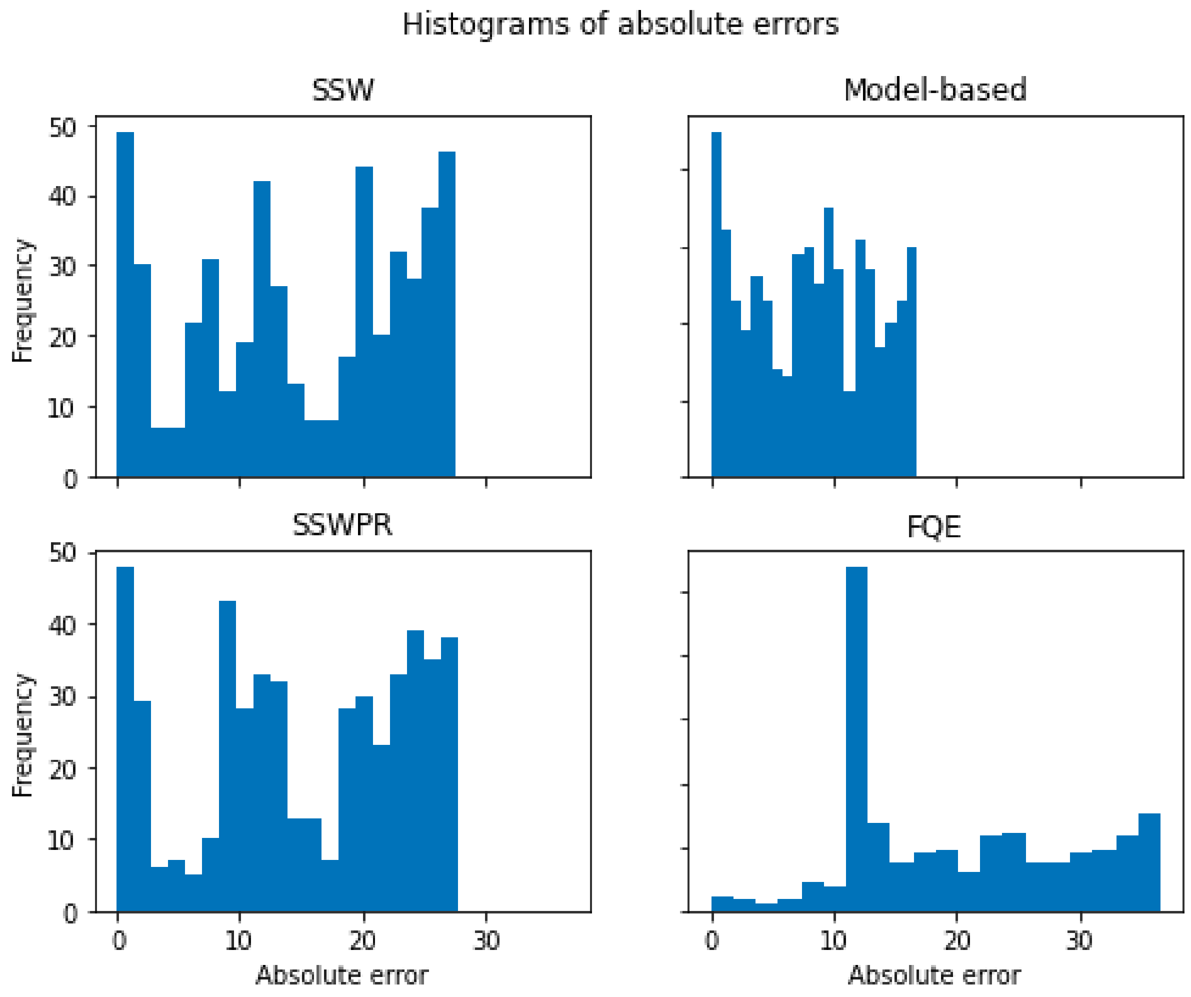


Figure 5.23: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_2} on the logged data deriving from π_{b_2} on the 2DWorld environment

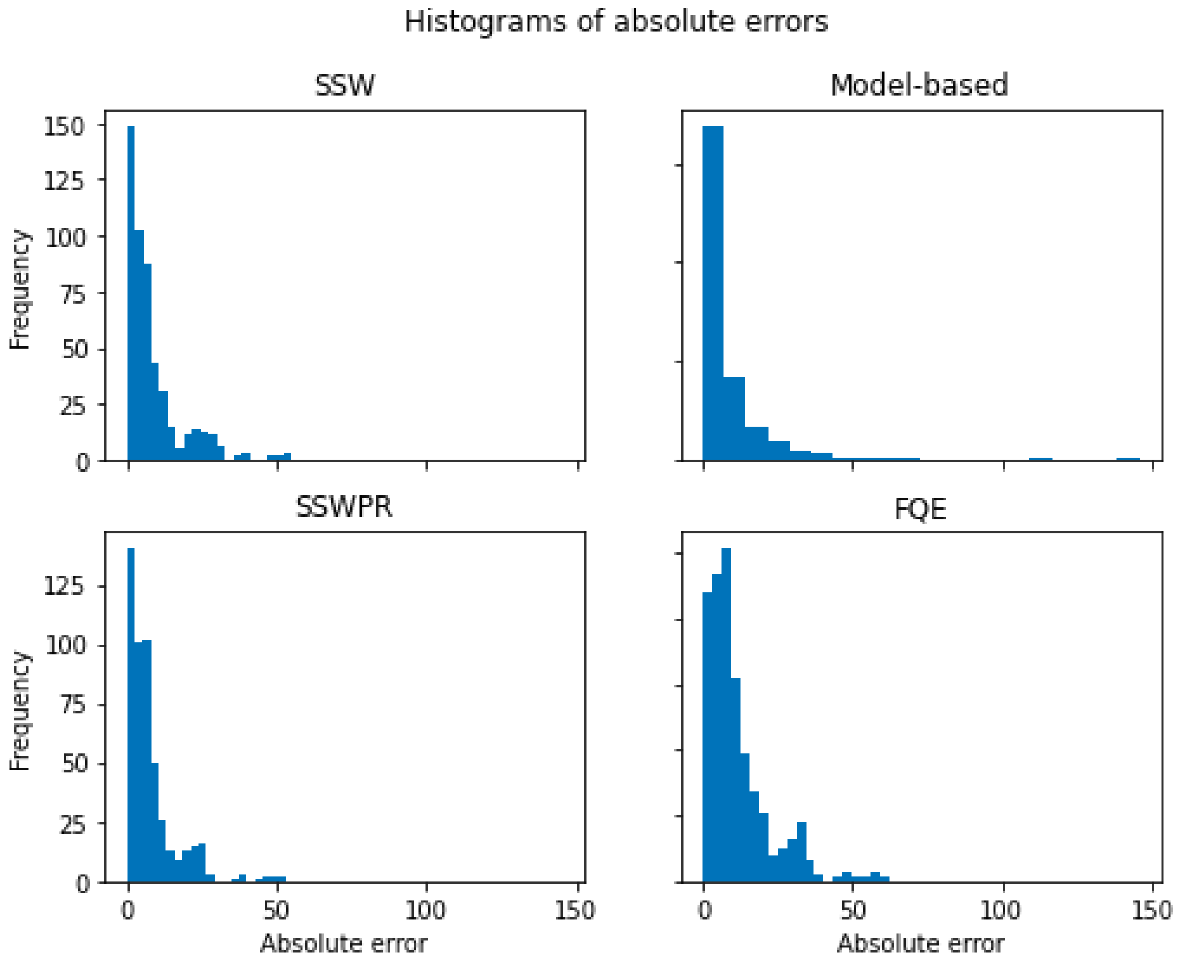


Figure 5.24: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_3} on the logged data deriving from π_{b_2} on the 2DWorld environment

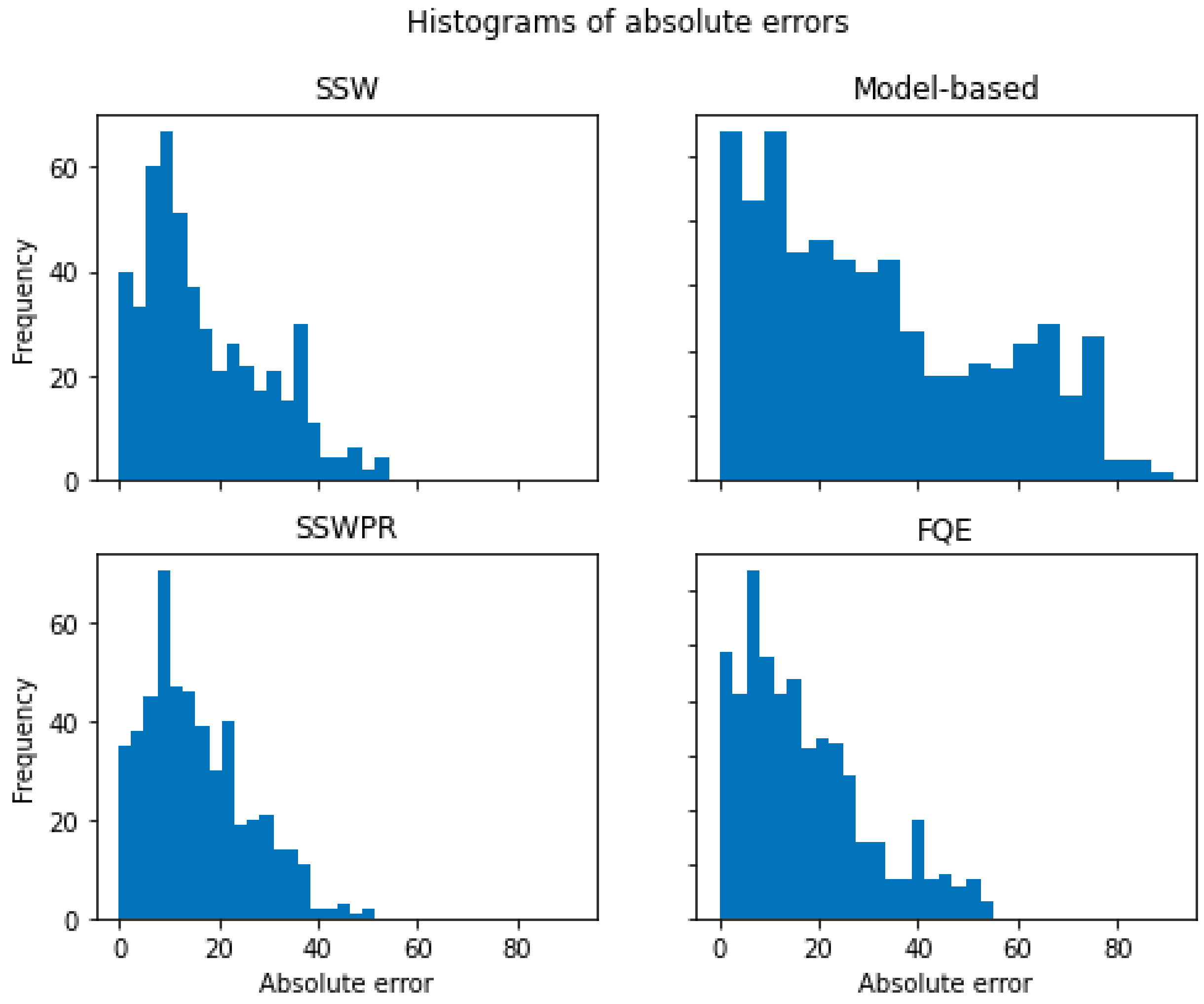


Figure 5.25: Scatterplots of the predictions of SSW vs SSWPR for evaluating A) π_{b_1}, B) π_{e_1}, C) π_{e_2}, D) π_{e_3} on the logged data deriving from π_{b_1} on the 2DWorld environment

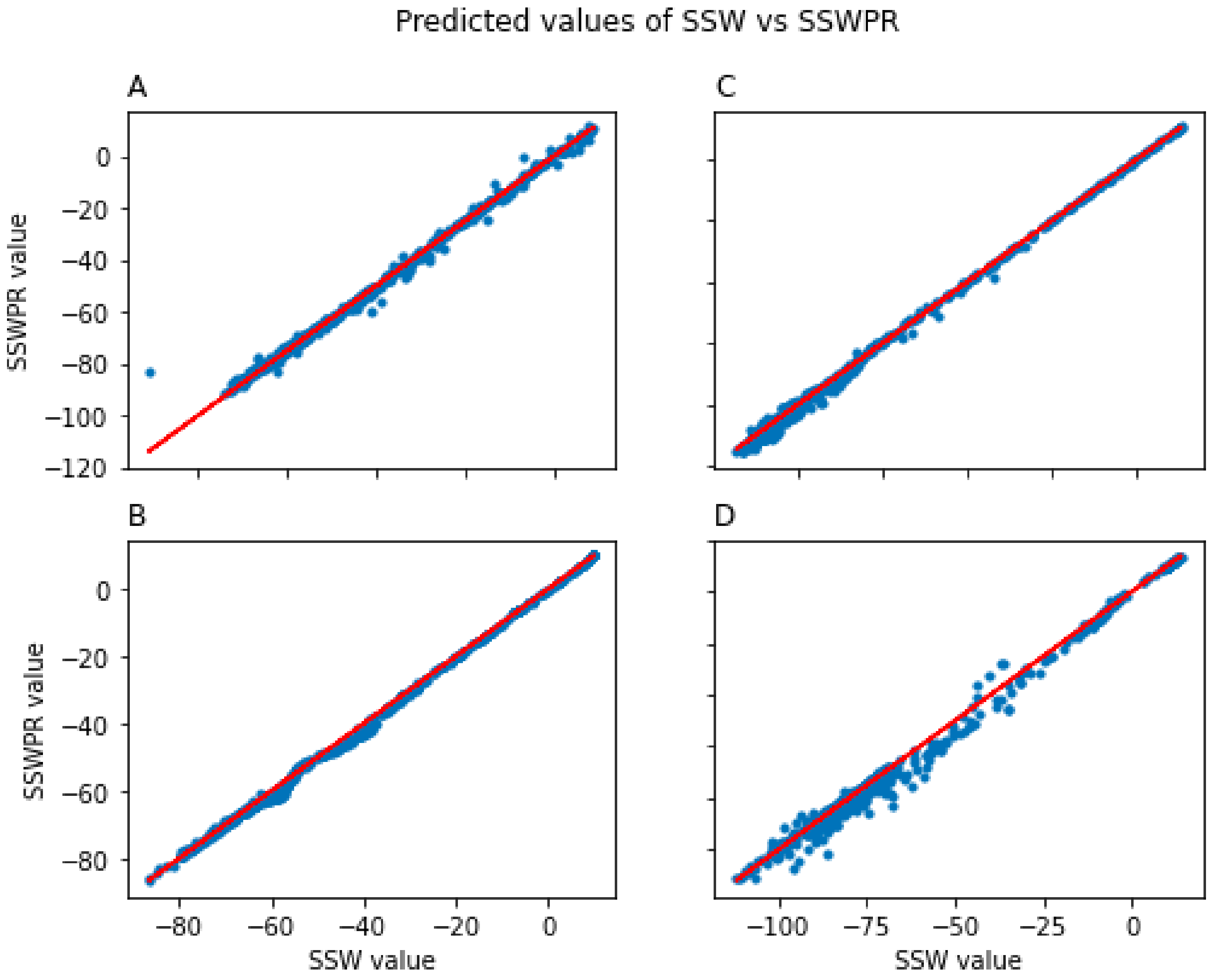


Figure 5.26: Scatterplots of the predictions of SSW vs SSWPR for evaluating A) π_{b_2}, B) π_{e_1}, C) π_{e_2}, D) π_{e_3} on the logged data deriving from π_{b_2} on the 2DWorld environment

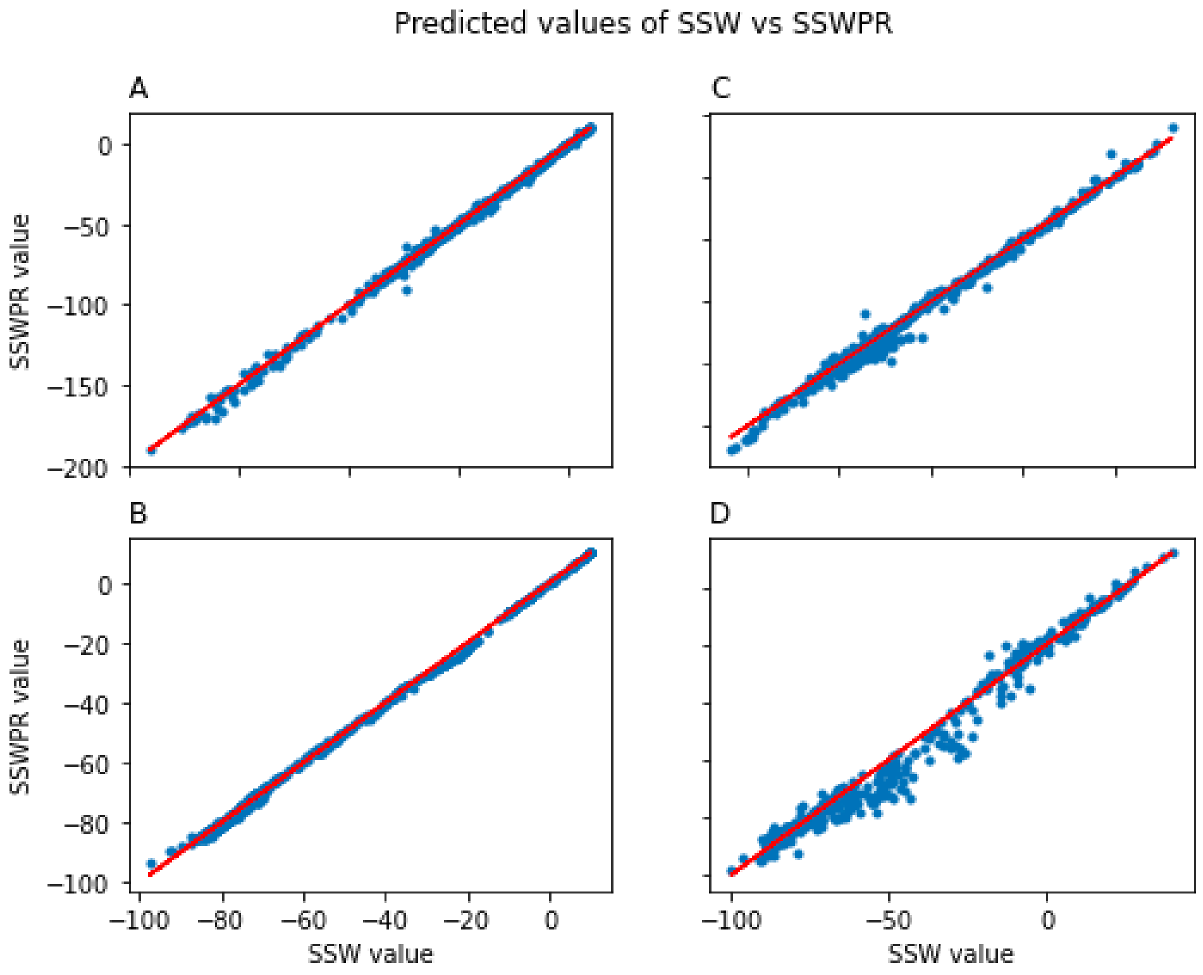


Figure 5.27: Scatterplots of the predictions and target values for evaluating π_{b_1} on the logged data deriving from π_{b_1} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

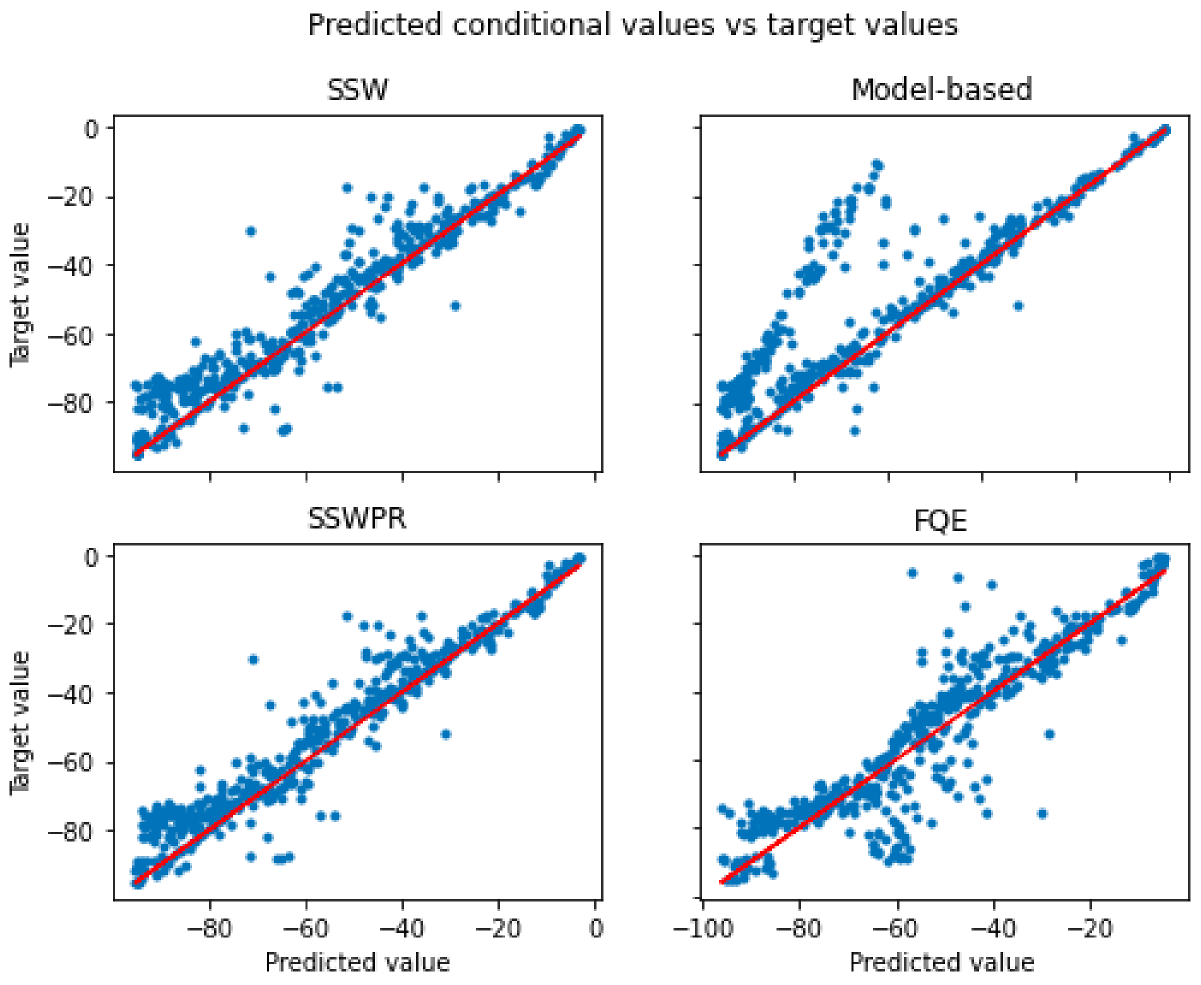


Figure 5.28: Scatterplots of the predictions and target values for evaluating π_{e_1} on the logged data deriving from π_{b_1} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

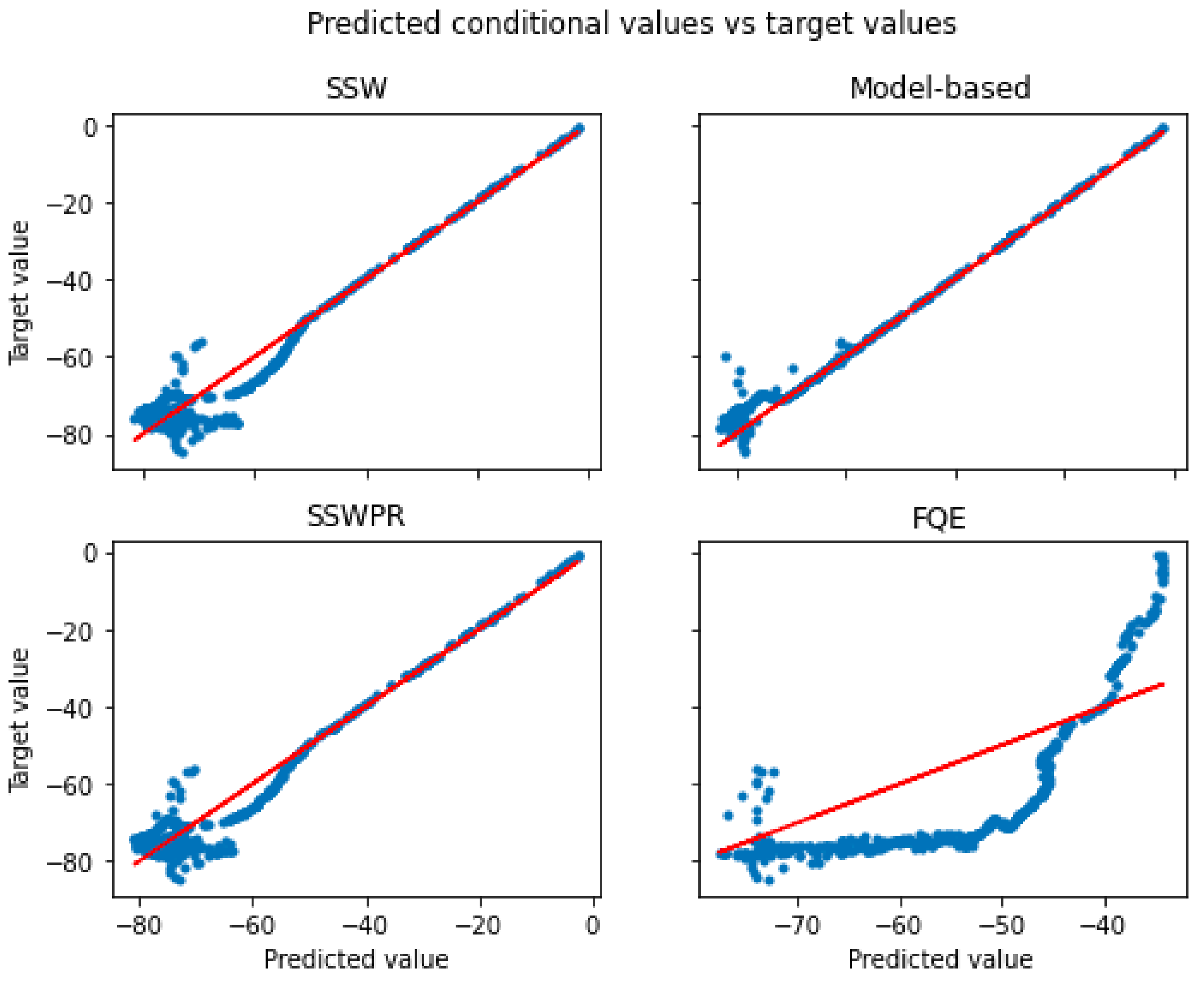


Figure 5.29: Scatterplots of the predictions and target values for evaluating π_{e_2} on the logged data deriving from π_{b_1} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

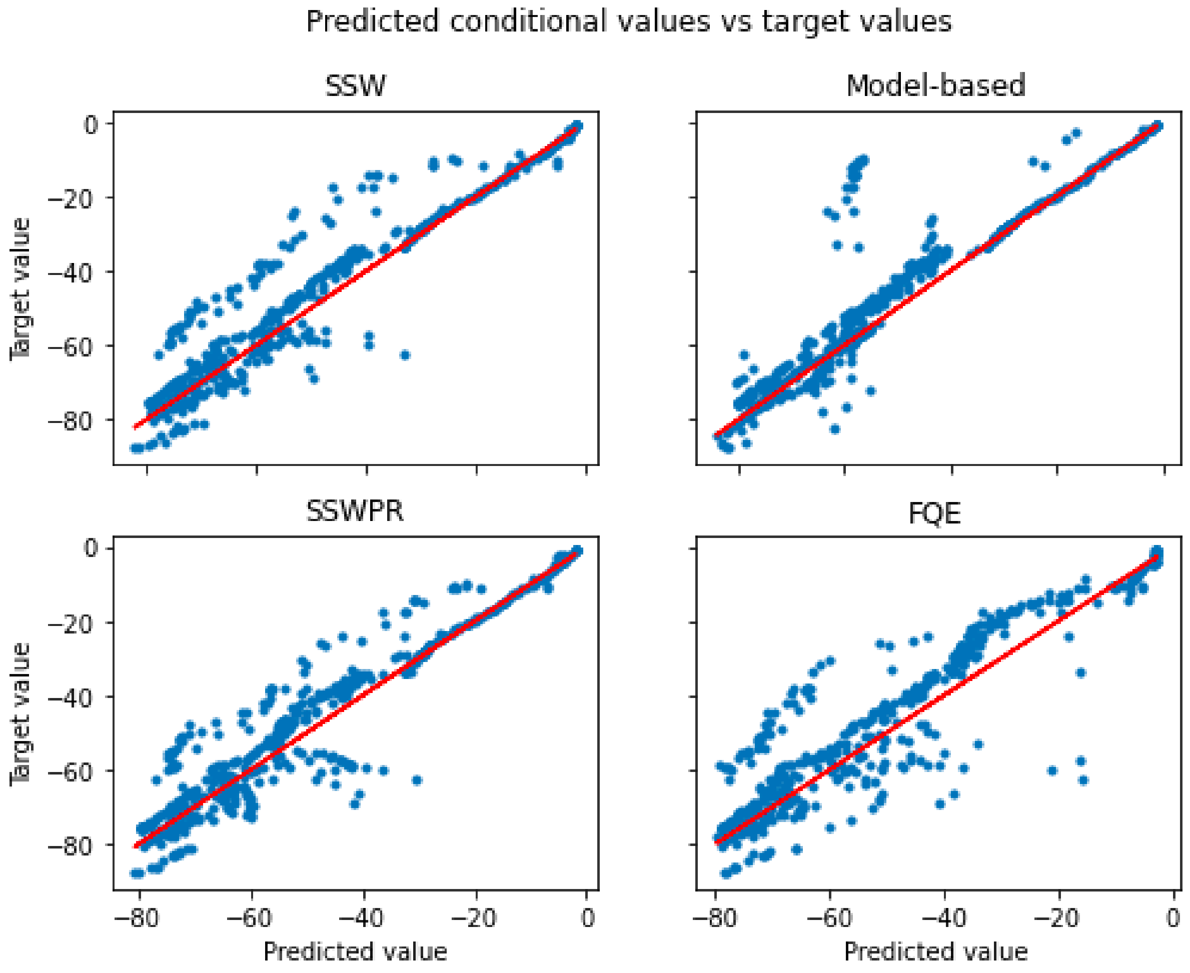


Figure 5.30: Scatterplots of the predictions and target values for evaluating π_{e_3} on the logged data deriving from π_{b_1} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

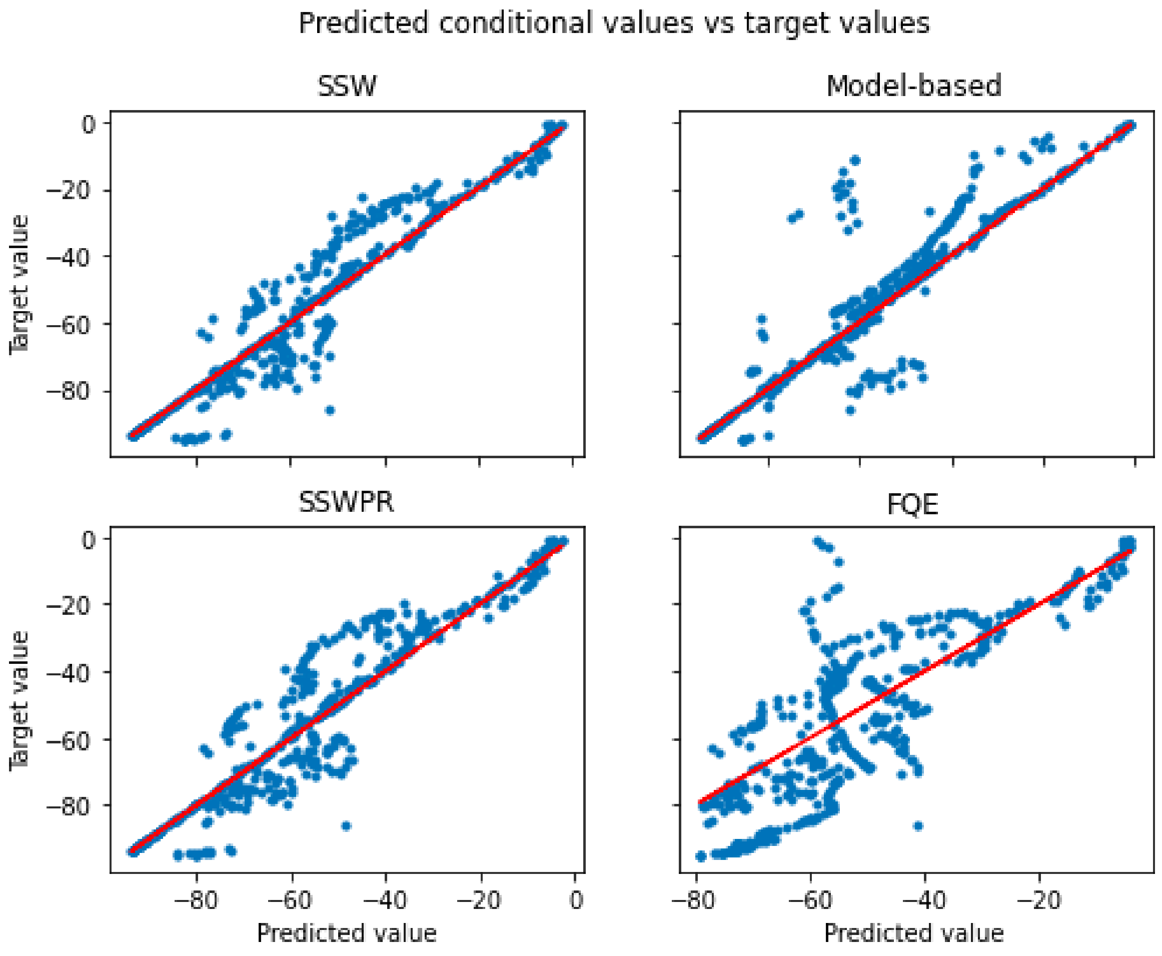


Figure 5.31: Scatterplots of the predictions and target values for evaluating π_{b_2} on the logged data deriving from π_{b_2} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

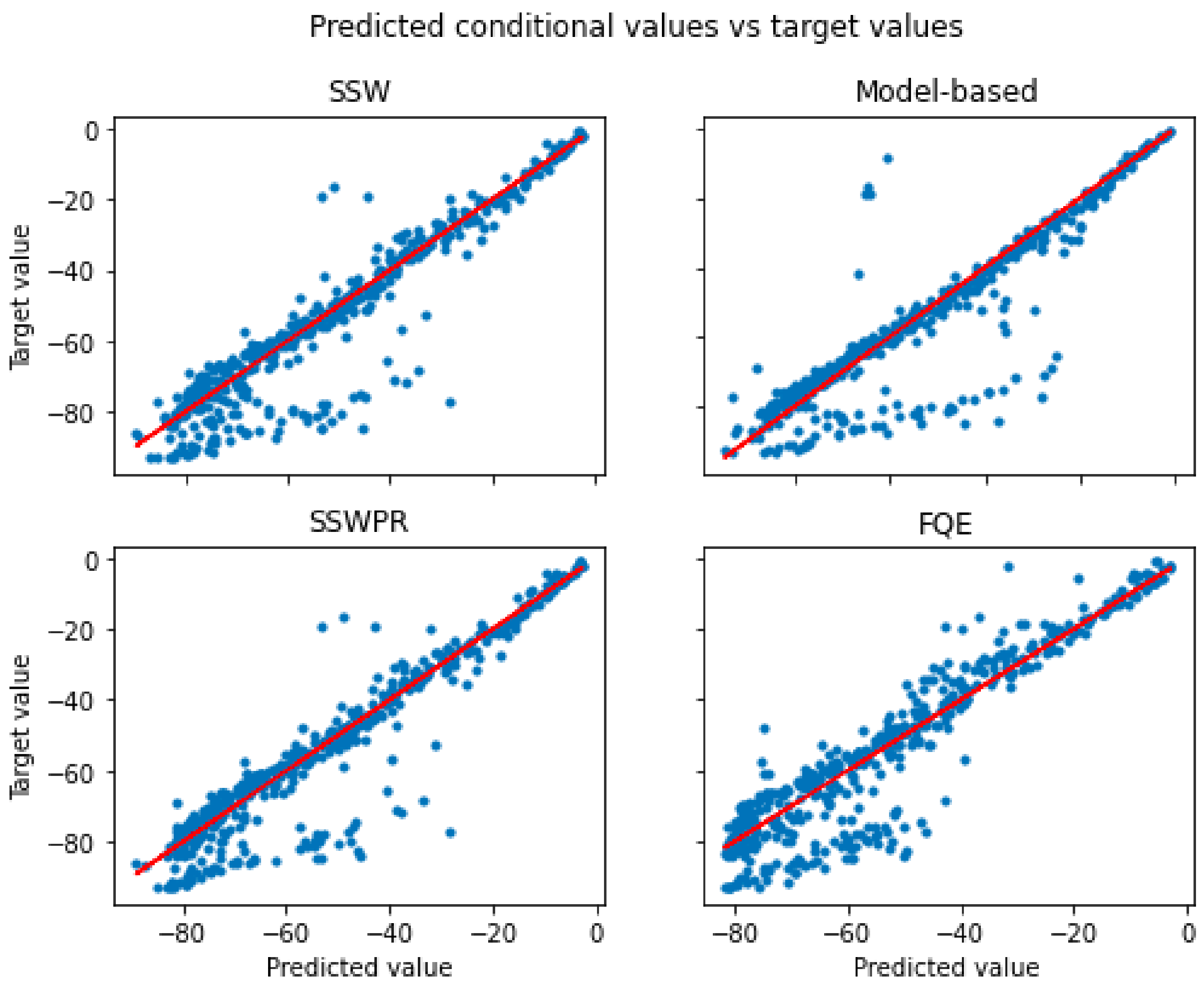


Figure 5.32: Scatterplots of the predictions and target values for evaluating π_{e_1} on the logged data deriving from π_{b_2} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

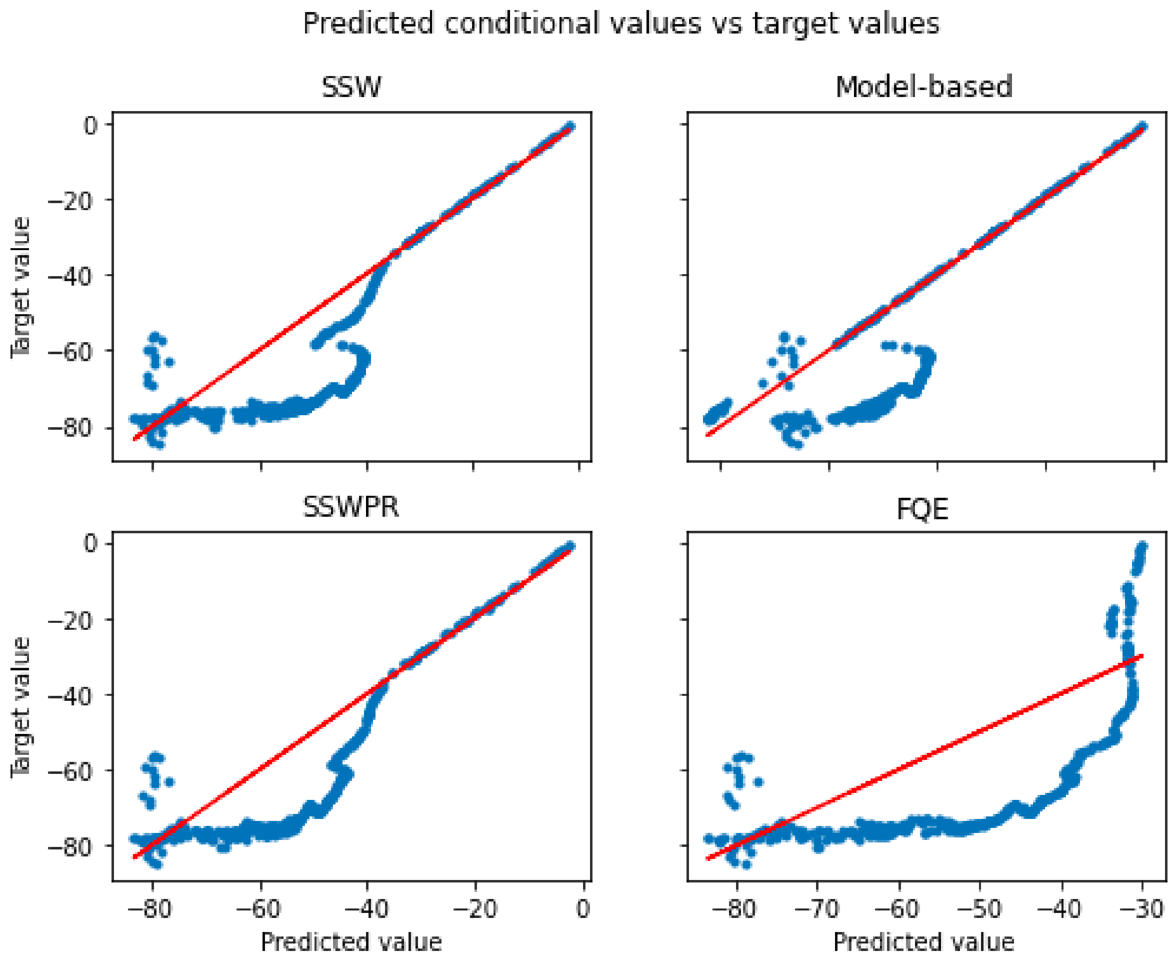


Figure 5.33: Scatterplots of the predictions and target values for evaluating π_{e_2} on the logged data deriving from π_{b_2} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

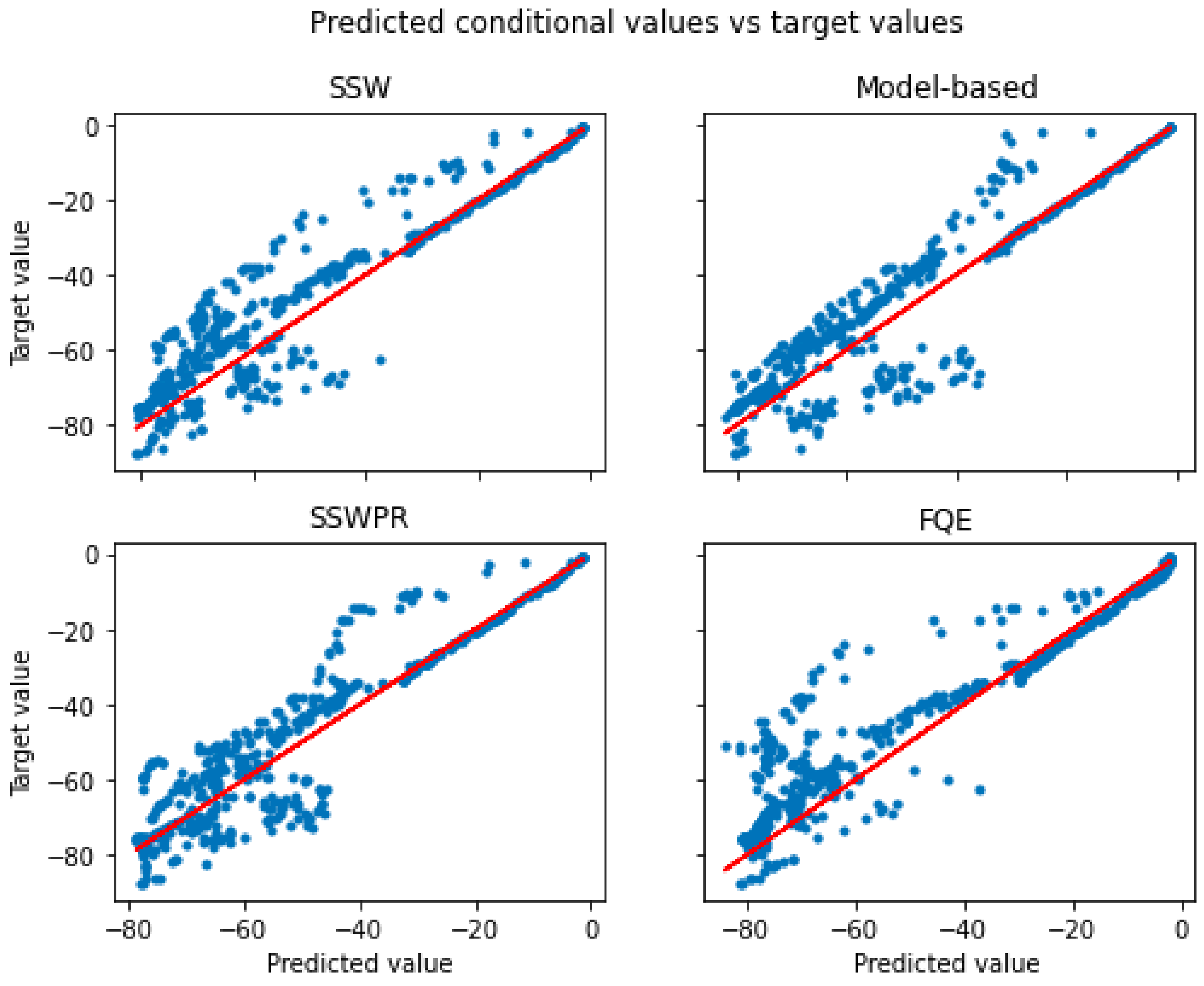


Figure 5.34: Scatterplots of the predictions and target values for evaluating π_{e_3} on the logged data deriving from π_{b_2} on the mountain car environment using the model-based method, FQE, SSW, and SSWPR.

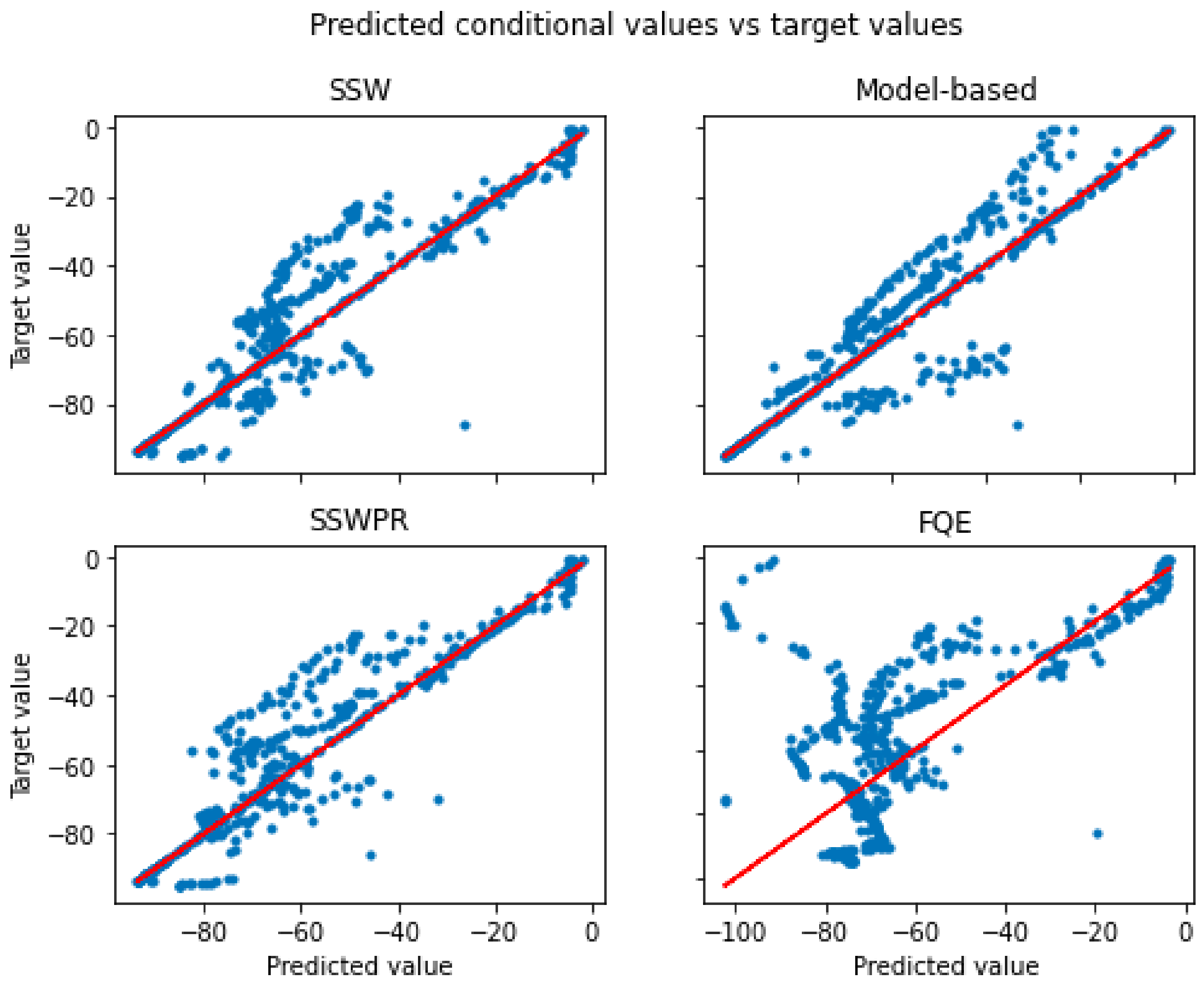


Figure 5.35: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{b_1} on the logged data deriving from π_{b_1} on the mountain car environment

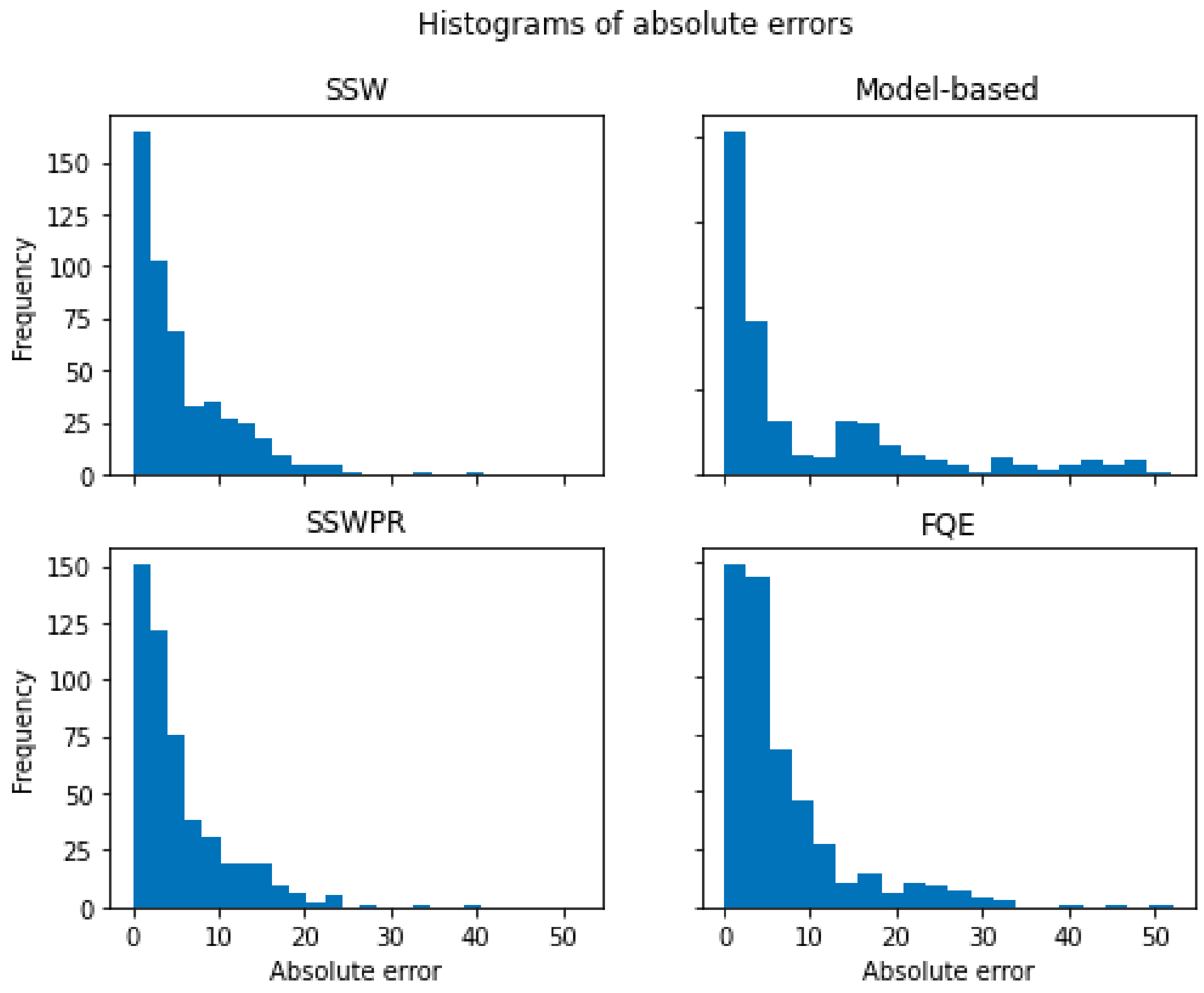


Figure 5.36: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_1} on the logged data deriving from π_{b_1} on the mountain car environment

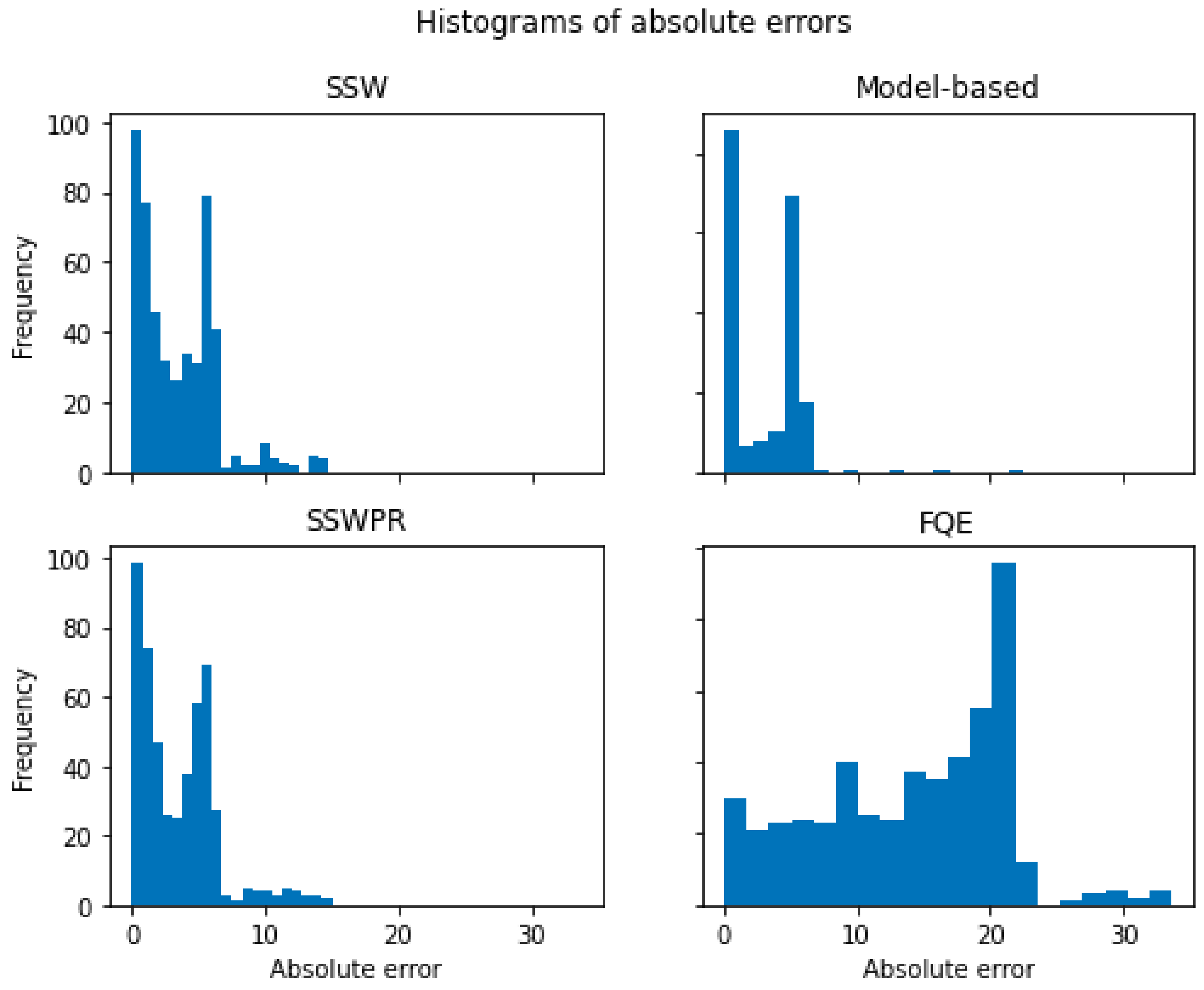


Figure 5.37: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_2} on the logged data deriving from π_{b_1} on the mountain car environment

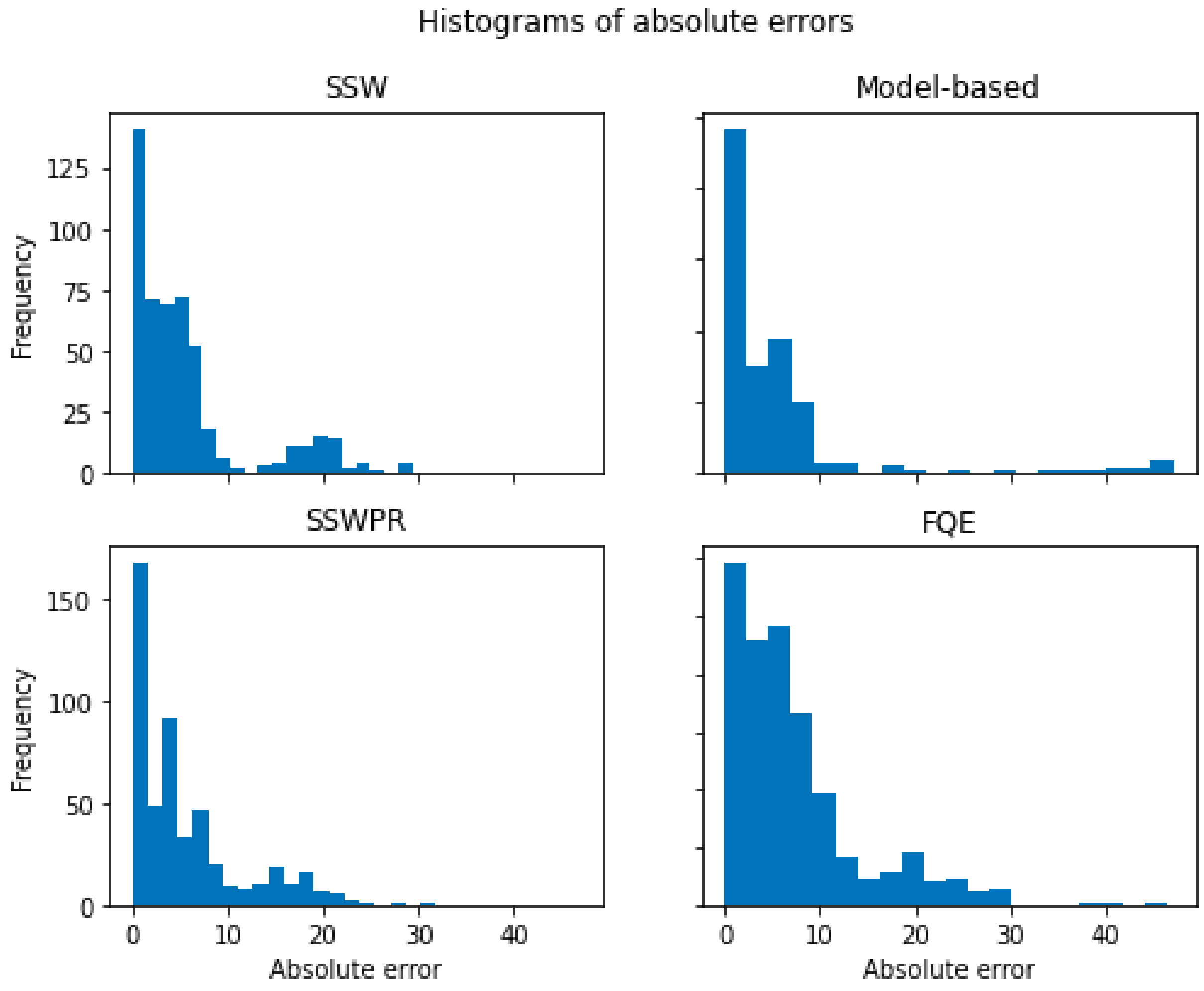


Figure 5.38: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_3} on the logged data deriving from π_{b_1} on the mountain car environment

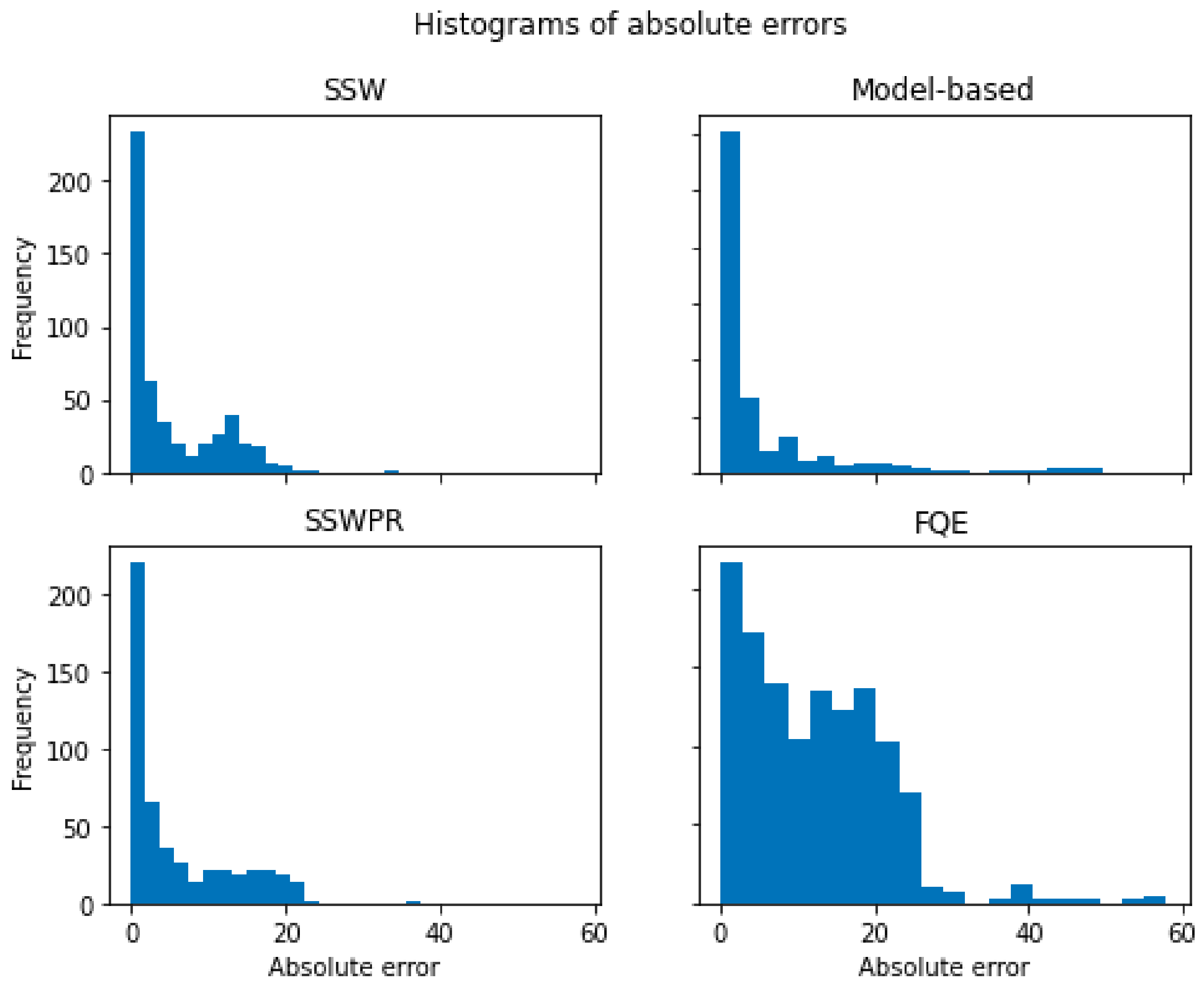


Figure 5.39: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{b_2} on the logged data deriving from π_{b_2} on the mountain car environment

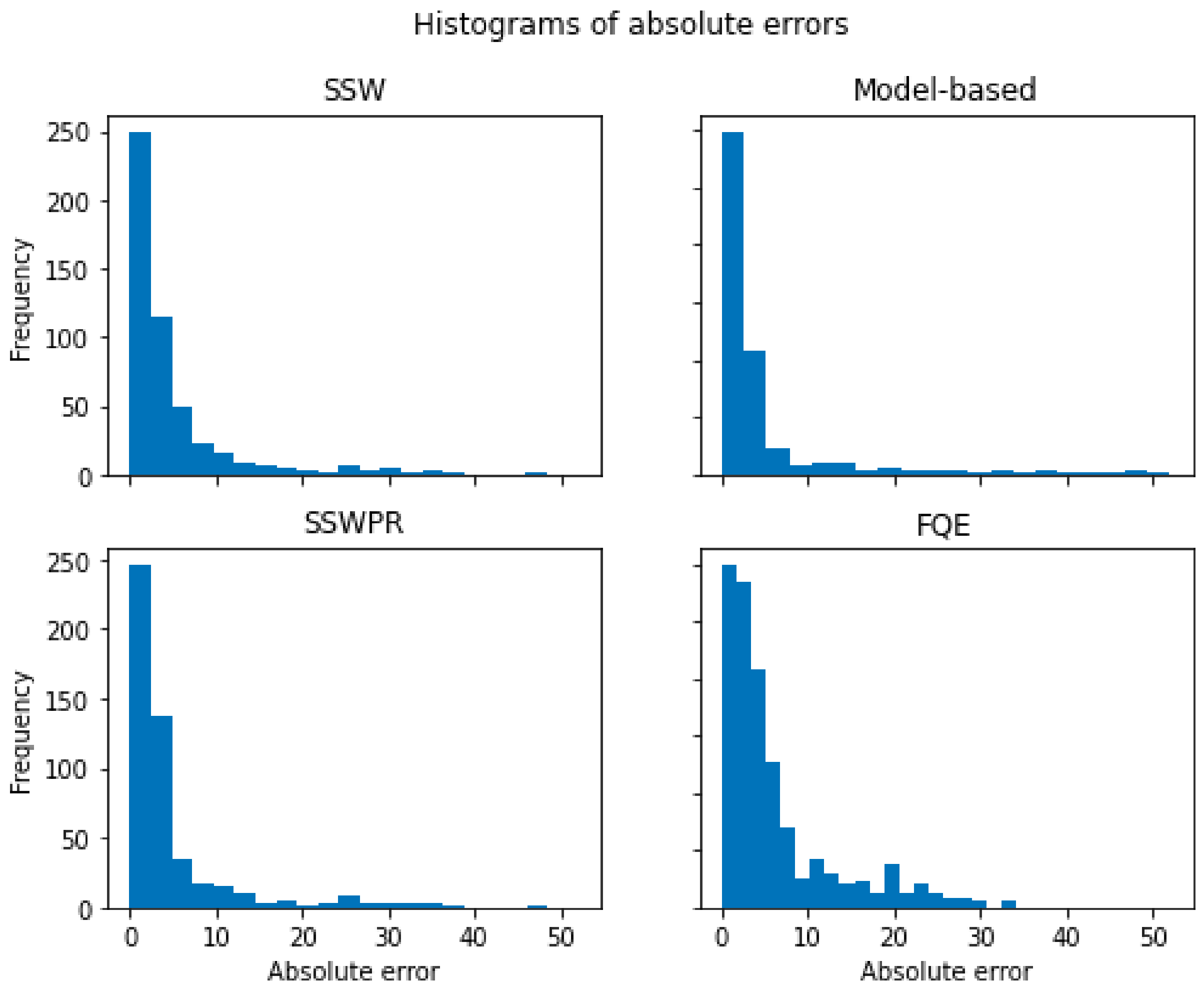


Figure 5.40: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_1} on the logged data deriving from π_{b_2} on the mountain car environment

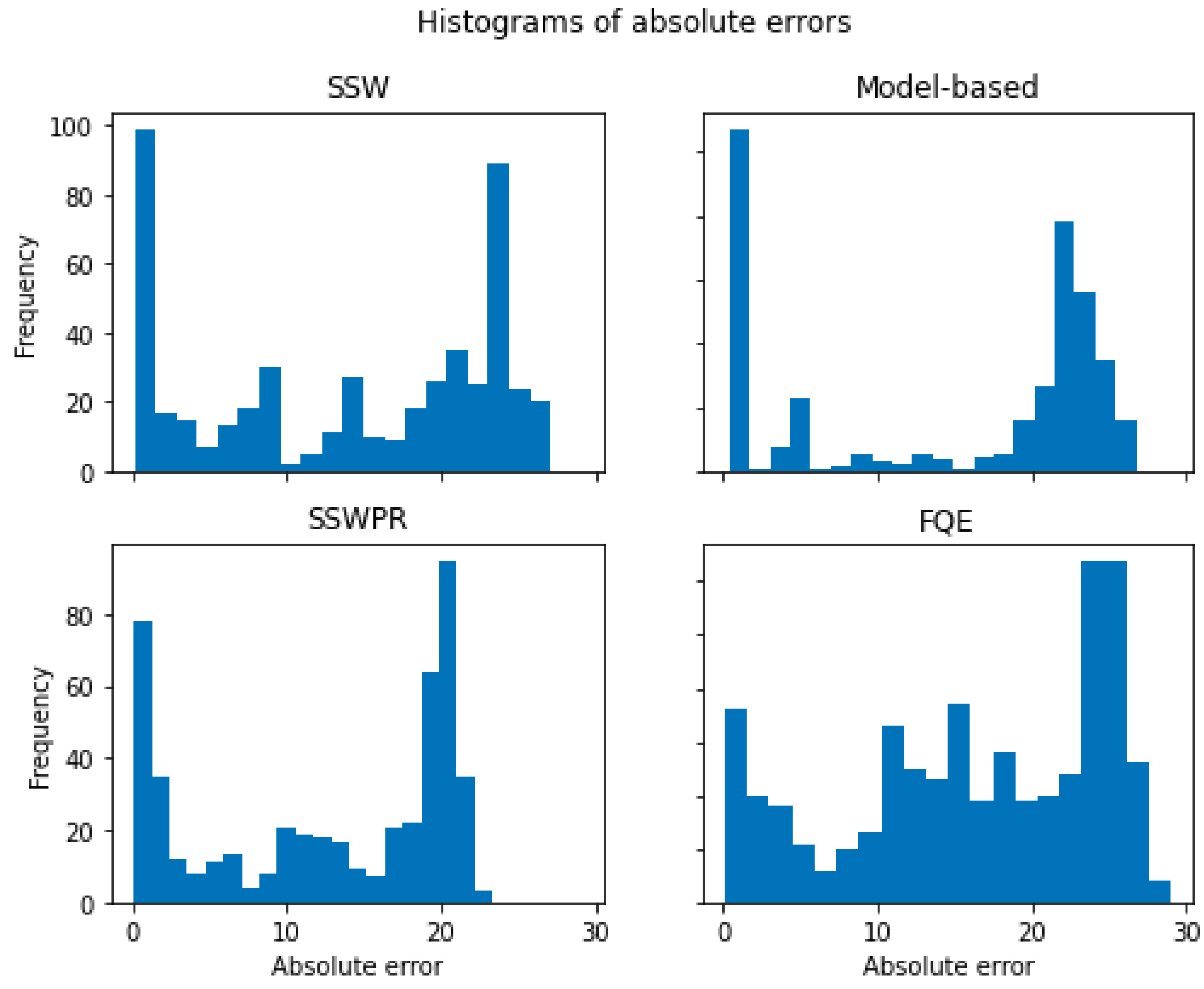


Figure 5.41: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_2} on the logged data deriving from π_{b_2} on the mountain car environment

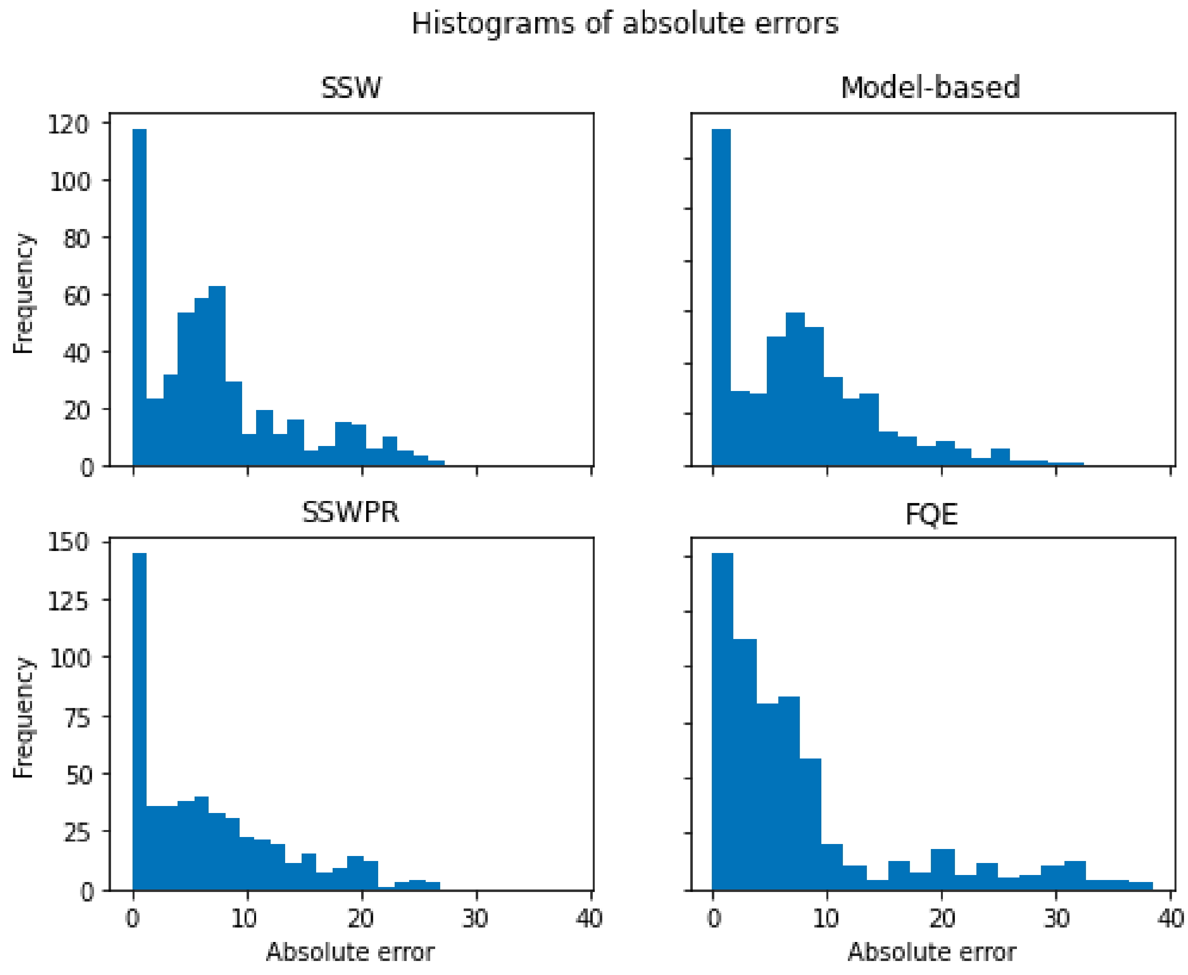


Figure 5.42: Histogram of absolute errors of SSW, SSWPR, FQE, and the model-based method for evaluating π_{e_3} on the logged data deriving from π_{b_2} on the mountain car environment

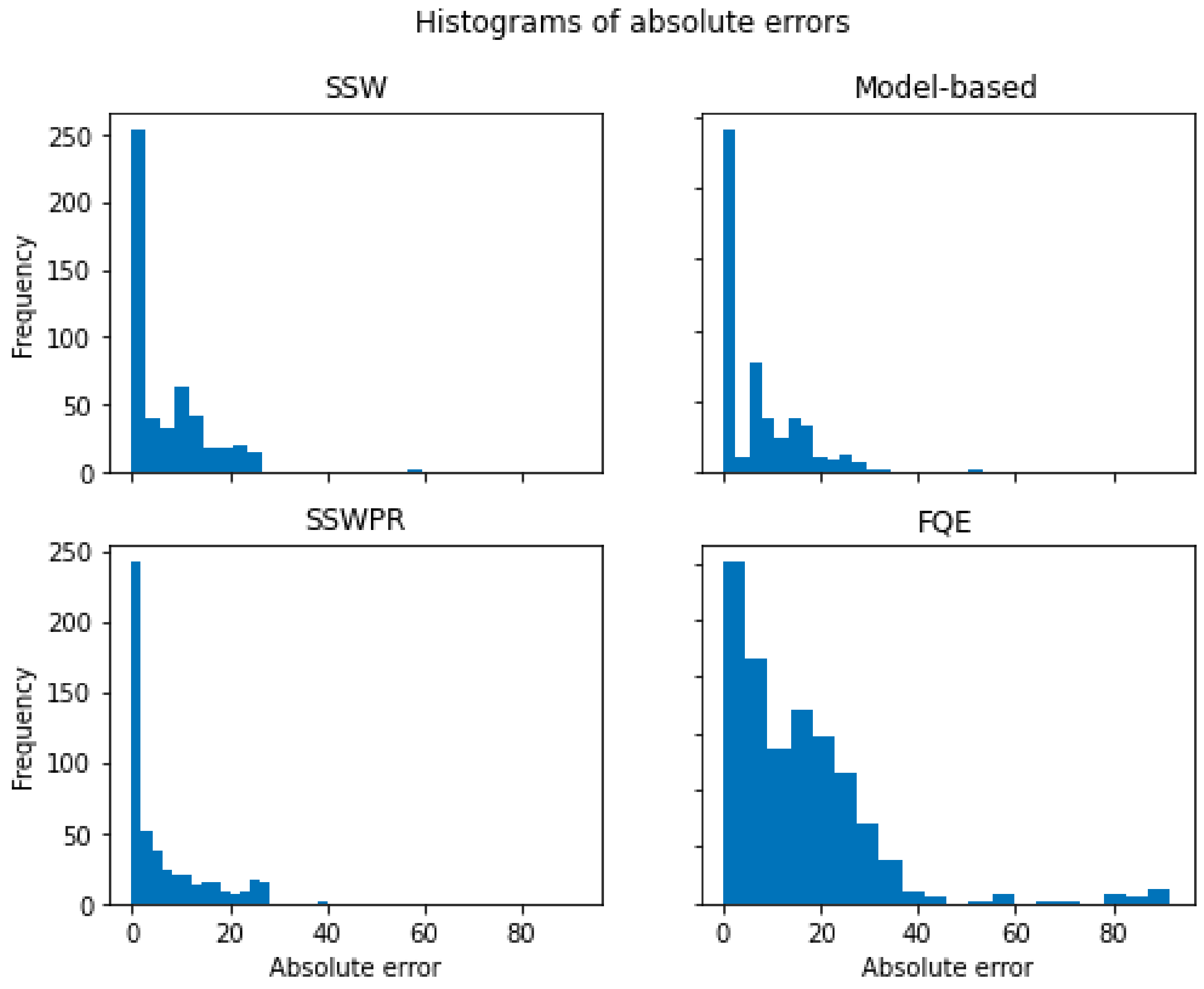


Figure 5.43: Scatterplots of the predictions of SSW vs SSWPR for evaluating A) π_{b_1}, B) π_{e_1}, C) π_{e_2}, D) π_{e_3} on the logged data deriving from π_{b_1} on the mountain car environment

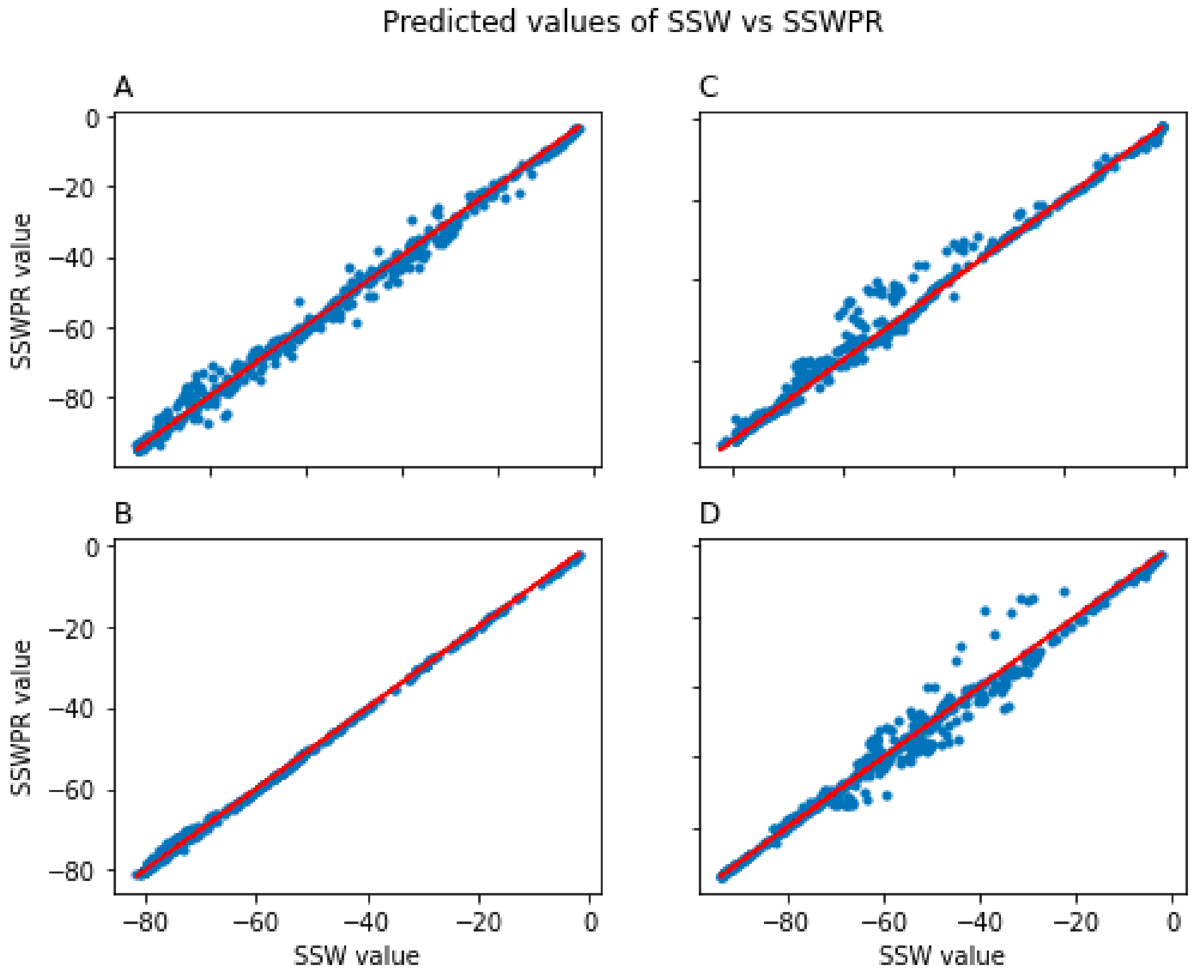
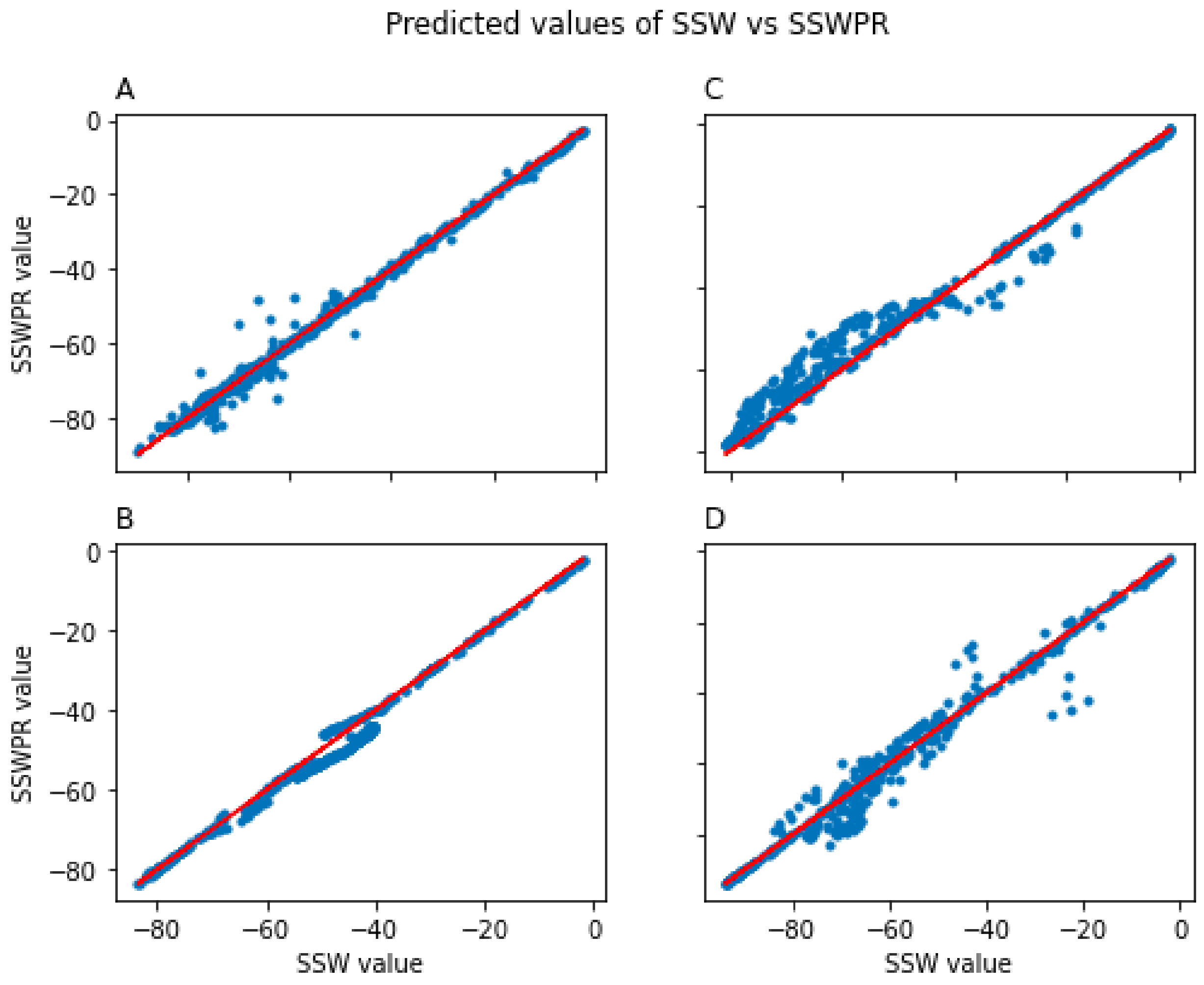


Figure 5.44: Scatterplots of the predictions of SSW vs SSWPR for evaluating A) π_{b_2}, B) π_{e_1}, C) π_{e_2}, D) π_{e_3} on the logged data deriving from π_{b_2} on the mountain car environment


Predicted values of SSW vs SSWPR
A
B
C
D
SSWPR value
SSWPR value
SSW value
SSW value
0
−20
−40
−60
−80

Bibliography

[1] Rios A and Kavuluлu R. “Few-shot and zero-shot multi-label learning for structured label spaces”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing.* 2018, pp. 3132–3142.

[2] Mohammed Alawad et al. “Automatic extraction of cancer registry reportable information from free-text pathology reports using multitask convolutional neural networks”. In: *Journal of the American Medical Informatics Association* 27.1 (Nov. 2019), pp. 89–98. ISSN: 1527-974X. DOI: `10.1093/jamia/ocz153`. eprint: `https://academic.oup.com/jamia/article-pdf/27/1/89/33744947/ocz153.pdf`. URL: `https://doi.org/10.1093/jamia/ocz153`.

[3] Mohammed Alawad et al. “Automatic extraction of cancer registry reportable information from free-text pathology reports using multitask convolutional neural networks”. In: *Journal of the American Medical Informatics Association* 27.1 (2020), pp. 89–98.

[4] Christoph Alt, Marc Hübner, and Leonhard Hennig. *Improving Relation Extraction by Pre-trained Language Representations.* June 2019.

[5] Nicholas Altieri et al. “Curating a COVID-19 Data Repository and Forecasting County-Level Death Counts in the United States”. In: *Harvard Data Science Review* NaN.Special Issue 1 (Feb. 8, 2021). https://hdsr.mitpress.mit.edu/pub/p6isyf0g. DOI: `10.1162/99608f92.1d4e0dae`. URL: `https://hdsr.mitpress.mit.edu/pub/p6isyf0g`.

[6] Nicholas Altieri et al. “Supervised line attention for tumor attribute classification from pathology reports: Higher performance with less data”. In: *Journal of Biomedical Informatics* 122 (2021), p. 103872. ISSN: 1532-0464. DOI: `https://doi.org/10.1016/j.jbi.2021.103872`. URL: `https://www.sciencedirect.com/science/article/pii/S153204642100201X`.

[7] *Aspect-augmented Adversarial Networks for Domain Adaptation | Transactions of the Association for Computational Linguistics | MIT Press Journals.* URL: `https://www.mitpressjournals.org/doi/abs/10.1162/tacl_a_00077` (visited on 02/24/2020).

[8] James Bergstra and Yoshua Bengio. “Random Search for Hyper-Parameter Optimization”. In: *Journal of Machine Learning Research* 13.Feb (2012), pp. 281–305. ISSN: ISSN 1533-7928. URL: `http://www.jmlr.org/papers/v13/bergstra12a.html?source=post_page---------------------------` (visited on 02/24/2020).

[9] Gerard Burger et al. “Natural language processing in pathology: a scoping review”. eng. In: *Journal of Clinical Pathology* (July 2016). ISSN: 1472-4146. DOI: `10.1136/jclinpath-2016-203872`.

[10] *Cancer Protocol Templates*. en-us. Library Catalog: www.cap.org. URL: `https://www.cap.org/protocols-and-guidelines/cancer-reporting-tools/cancer-protocol-templates` (visited on 05/06/2020).

[11] Nicolò Cesa-Bianchi et al. “On-line Prediction and Conversion Strategies”. In: *Machine Learning* 25 (2005), pp. 71–110.

[12] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. en-us. In: Oct. 2014, pp. 1724–1734. DOI: `10.3115/v1/D14-1179`. URL: `https://www.aclweb.org/anthology/D14-1179` (visited on 02/24/2020).

[13] Kurtland Chua et al. “Deep Reinforcement Learning in a Handful of Trials Using Probabilistic Dynamics Models”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS’18. Montréal, Canada: Curran Associates Inc., 2018, pp. 4759–4770.

[14] Anni Coden et al. “Automatically extracting cancer disease characteristics from pathology reports into a Disease Knowledge Representation Model”. en. In: *Journal of Biomedical Informatics* 42.5 (Oct. 2009), pp. 937–949. ISSN: 15320464. DOI: `10.1016/j.jbi.2008.12.005`. URL: `http://linkinghub.elsevier.com/retrieve/pii/S1532046408001585` (visited on 01/27/2018).

[15] Morris H. Degroot and Stephen E. Fienberg. “The Comparison and Evaluation of Forecasters”. en. In: *Journal of the Royal Statistical Society: Series D (The Statistician)* 32.1-2 (1983), pp. 12–22. ISSN: 1467-9884. DOI: `10.2307/2987588`. URL: `https://rss.onlinelibrary.wiley.com/doi/abs/10.2307/2987588` (visited on 01/18/2020).

[16] Louise Deleger et al. “Building Gold Standard Corpora for Medical Natural Language Processing Tasks”. In: *AMIA Annual Symposium Proceedings* 2012 (Nov. 2012), pp. 144–153. ISSN: 1942-597X. URL: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3540456/` (visited on 01/25/2018).

[17] Leon Derczynski. “Complementarity, F-score, and NLP Evaluation”. en. In: (), p. 6.

[18] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. en. In: (Oct. 2018). URL: `https://arxiv.org/abs/1810.04805v2` (visited on 06/21/2020).

[19] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[20] Yaqi Duan, Zeyu Jia, and Mengdi Wang. "Minimax-Optimal off-Policy Evaluation with Linear Function Approximation". In: *Proceedings of the 37th International Conference on Machine Learning*. ICML'20. JMLR.org, 2020.

[21] Glenn A. Edwards. "Expert systems for clinical pathology reporting". eng. In: *The Clinical Biochemist. Reviews* 29 Suppl 1 (Aug. 2008), S105–109. ISSN: 0159-8090.

[22] Allan Fong et al. "Assessment of Automating Safety Surveillance From Electronic Health Records: Analysis for the Quality and Safety Review System". In: *Journal of patient safety* (2016).

[23] Justin Fu et al. "Benchmarks for Deep Off-Policy Evaluation". In: *ArXiv* abs/2103.16596 (2021).

[24] Scott Fujimoto, David Meger, and Doina Precup. "Off-Policy Deep Reinforcement Learning without Exploration". In: *ICML*. 2019.

[25] Shang Gao et al. "Hierarchical attention networks for information extraction from cancer pathology reports". en. In: *Journal of the American Medical Informatics Association* 25.3 (Mar. 2018), pp. 321–330. ISSN: 1067-5027, 1527-974X. DOI: `10.1093/jamia/ocx131`. URL: `https://academic.oup.com/jamia/article/25/3/321/4636780` (visited on 10/09/2019).

[26] Shang Gao et al. "Hierarchical attention networks for information extraction from cancer pathology reports". en. In: *Journal of the American Medical Informatics Association* 25.3 (Mar. 2018), pp. 321–330. ISSN: 1067-5027. DOI: `10.1093/jamia/ocx131`. URL: `https://academic.oup.com/jamia/article/25/3/321/4636780` (visited on 02/24/2020).

[27] Alexander P. Glaser et al. "Automated Extraction of Grade, Stage, and Quality Information From Transurethral Resection of Bladder Tumor Pathology Reports Using Natural Language Processing". In: *JCO Clinical Cancer Informatics* 2 (2018), pp. 1–8. ISSN: 2473-4276. DOI: `10.1200/CCI.17.00128`. URL: `http://ascopubs.org/doi/10.1200/CCI.17.00128`.

[28] Botao Hao et al. "Bootstrapping Statistical Inference for Off-Policy Evaluation". In: *ArXiv* abs/2102.03607 (2021).

[29] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". en. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 328–339. DOI: `10.18653/v1/P18-1031`. URL: `http://aclweb.org/anthology/P18-1031` (visited on 05/22/2020).

[30] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. "ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission". en. In: (Apr. 2019). URL: `https://arxiv.org/abs/1904.05342v2` (visited on 06/21/2020).

[31] Sarthak Jain and Byron C. Wallace. "Attention is not Explanation". In: *arXiv:1902.10186 [cs]* (May 2019). arXiv: 1902.10186. URL: `http://arxiv.org/abs/1902.10186` (visited on 04/23/2020).

[32] Natasha Jaques et al. "Way Off-Policy Batch Deep Reinforcement Learning of Implicit Human Preferences in Dialog". In: *ArXiv* abs/1907.00456 (2019).

[33] Nan Jiang and Lihong Li. "Doubly Robust Off-Policy Value Evaluation for Reinforcement Learning". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, 2016, pp. 652–661.

[34] Alistair EW Johnson et al. "MIMIC-III, a freely accessible critical care database". In: *Scientific data* 3.1 (2016), pp. 1–9.

[35] V. Jouhet et al. "Automated classification of free-text pathology reports for registration of incident cases of cancer". eng. In: *Methods of Information in Medicine* 51.3 (2012), pp. 242–251. ISSN: 2511-705X. DOI: `10.3414/ME11-01-0005`.

[36] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. 2nd ed. Prentice Hall series in artificial intelligence. Pearson Prentice Hall, 2009. ISBN: 978-0-13-187321-6.

[37] Sham Kakade. "A Natural Policy Gradient". In: *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*. NIPS'01. Vancouver, British Columbia, Canada: MIT Press, 2001, pp. 1531–1538.

[38] Hidetaka Kamigaito et al. "Supervised Attention for Sequence-to-Sequence Constituency Parsing". en-us. In: Nov. 2017, pp. 7–12. URL: `https://www.aclweb.org/anthology/I17-2002` (visited on 02/24/2020).

[39] Rahul Kidambi et al. "MOReL: Model-Based Offline Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 21810–21823. URL: `https://proceedings.neurips.cc/paper/2020/file/f7efa4f864ae9b88d43527f4b14f750f-Paper.pdf`.

[40] Rahul Kidambi et al. "Morel: Model-based offline reinforcement learning". In: *Advances in neural information processing systems* 33 (2020), pp. 21810–21823.

[41] Diederik Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations* (Dec. 2014).

[42] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". en. In: (Dec. 2014). URL: `https://arxiv.org/abs/1412.6980v9` (visited on 02/24/2020).

[43] Bangalore Kiran et al. "Deep Reinforcement Learning for Autonomous Driving: A Survey". In: *IEEE Transactions on Intelligent Transportation Systems* PP (Feb. 2021), pp. 1–18. DOI: `10.1109/TITS.2021.3054625`.

[44] Ilya Kostrikov and Ofir Nachum. "Statistical Bootstrapping for Uncertainty Estimation in Off-Policy Evaluation". In: *ArXiv* abs/2007.13609 (2020).

[45] Kory Kreimeyer et al. "Natural language processing systems for capturing and standardizing unstructured clinical information: A systematic review". In: *Journal of Biomedical Informatics* 73 (Sept. 2017), pp. 14–29. DOI: `10.1016/j.jbi.2017.07.012`. URL: `http://dx.doi.org/10.1016/j.jbi.2017.07.012`.

[46] Aviral Kumar et al. "Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[47] Hoang Le, Cameron Voloshin, and Yisong Yue. "Batch Policy Learning under Constraints". In: (Mar. 2019).

[48] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". en. In: (Jan. 2019). DOI: `10.1093/bioinformatics/btz682`. URL: `https://arxiv.org/abs/1901.08746v4` (visited on 06/21/2020).

[49] Sergey Levine and Vladlen Koltun. "Guided Policy Search". In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1–9. URL: `https://proceedings.mlr.press/v28/levine13.html`.

[50] Sergey Levine et al. "Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems". In: *ArXiv* abs/2005.01643 (2020).

[51] Yue Li and David Martinez. "Information Extraction of Multiple Categories from Pathology Reports". In: *Proceedings of the Australasian Language Technology Association Workshop 2010*. Melbourne, Australia, Dec. 2010, pp. 41–48. URL: `https://www.aclweb.org/anthology/U10-1008` (visited on 12/16/2019).

[52] Lemao Liu et al. "Neural Machine Translation with Supervised Attention". en-us. In: Dec. 2016, pp. 3093–3102. URL: `https://www.aclweb.org/anthology/C16-1291` (visited on 02/24/2020).

[53] Siqi Liu et al. "Reinforcement Learning for Clinical Decision Support in Critical Care: Comprehensive Review". In: *Journal of Medical Internet Research* 22 (2020).

[54] Jueqing Lu et al. "Multi-label few/zero-shot learning with knowledge aggregated from multiple label graphs". In: *arXiv preprint arXiv:2010.07459* (2020).

[55] David Martinez and Yue Li. "Information extraction from pathology reports in a hospital setting". In: *Proceedings of the 20th ACM international conference on Information and knowledge management*. 2011, pp. 1877–1882.

[56] Mausam and Andrey Kolobov. *Planning with Markov Decision Processes: An AI Perspective*. Morgan amp; Claypool Publishers, 2012. ISBN: 1608458865.

[57] Volodymyr Mnih et al. "Playing Atari with Deep Reinforcement Learning". In: (Dec. 2013).

[58] Giulio Napolitano et al. "Pattern-based information extraction from pathology reports for cancer registration". en. In: *Cancer Causes & Control* 21.11 (Nov. 2010), pp. 1887–1894. ISSN: 0957-5243, 1573-7225. DOI: 10.1007/s10552-010-9616-4. URL: http://link.springer.com/10.1007/s10552-010-9616-4 (visited on 07/22/2019).

[59] Anthony N Nguyen et al. "Symbolic rule-based classification of lung cancer stages from free-text pathology reports". en. In: *Journal of the American Medical Informatics Association* 17.4 (July 2010), pp. 440–445. ISSN: 1527-974X, 1067-5027. DOI: 10.1136/jamia.2010.003707. URL: https://academic.oup.com/jamia/article-lookup/doi/10.1136/jamia.2010.003707 (visited on 07/22/2019).

[60] Anobel Odisho* et al. "PD58-09 EXTRACTING STRUCTURED INFORMATION FROM PATHOLOGY REPORTS USING NATURAL LANGUAGE PROCESSING AND MACHINE LEARNING". en. In: *The Journal of Urology* (Apr. 2019). URL: https://www.auajournals.org/doi/abs/10.1097/01.JU.0000557177.97226.63 (visited on 02/24/2020).

[61] Philip V. Ogren et al. "Building and Evaluating Annotated Corpora for Medical NLP Systems". In: *AMIA Annual Symposium Proceedings* 2006 (2006), p. 1050. ISSN: 1942-597X. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1839264/ (visited on 01/27/2018).

[62] Tomasz Oliwa et al. "Obtaining Knowledge in Pathology Reports Through a Natural Language Processing Approach With Classification, Named-Entity Recognition, and Relation-Extraction Heuristics". In: *JCO Clinical Cancer Informatics* 3 (July 2019). Publisher: American Society of Clinical Oncology, pp. 1–8. DOI: 10.1200/CCI.19.00008. URL: https://ascopubs.org/doi/full/10.1200/CCI.19.00008 (visited on 04/18/2020).

[63] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[64] Matthew E. Peters et al. "Deep contextualized word representations". en. In: (Feb. 2018). URL: https://arxiv.org/abs/1802.05365v2 (visited on 06/21/2020).

[65] Athanasios Polydoros and Lazaros Nalpantidis. "Survey of Model-Based Reinforcement Learning: Applications on Robotics". In: *Journal of Intelligent Robotic Systems* 86 (May 2017), pp. 153–. DOI: 10.1007/s10846-017-0468-y.

[66] Doina Precup, Richard S. Sutton, and Satinder P. Singh. "Eligibility Traces for Off-Policy Policy Evaluation". In: *Proceedings of the Seventeenth International Conference on Machine Learning*. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 759–766. ISBN: 1558607072.

[67] Rafael Prudencio, Marcos Maximo, and Esther Colombini. "A Survey on Offline Reinforcement Learning: Taxonomy, Review, and Open Problems". In: (Mar. 2022).

[68] Sampo Pyysalo et al. *Distributional Semantics Resources for Biomedical Text Processing*. en. Library Catalog: www.semanticscholar.org. 2013. (Visited on 06/21/2020).

[69] John X. Qiu et al. “Deep Learning for Automated Extraction of Primary Sites From Cancer Pathology Reports”. eng. In: *IEEE journal of biomedical and health informatics* 22.1 (2018), pp. 244–251. ISSN: 2168-2208. DOI: 10.1109/JBHI.2017.2700722.

[70] Angus Roberts et al. “The CLEF Corpus: Semantic Annotation of Clinical Text”. In: *AMIA Annual Symposium Proceedings* 2007 (2007), pp. 625–629. ISSN: 1942-597X. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2655900/ (visited on 01/27/2018).

[71] Florian R Schroeck et al. “Development of a Natural Language Processing Engine to Generate Bladder Cancer Pathology Data for Health Services Research”. In: *URL* 110 (Dec. 2017), pp. 84–91. DOI: 10.1016/j.urology.2017.07.056. URL: https://doi.org/10.1016/j.urology.2017.07.056.

[72] Yanyao Shen et al. “Deep Active Learning for Named Entity Recognition”. en. In: (July 2017). URL: https://arxiv.org/abs/1707.05928v3 (visited on 05/22/2020).

[73] Yuqi Si and Kirk Roberts. “A Frame-Based NLP System for Cancer-Related Information Extraction”. In: *AMIA Annual Symposium Proceedings* 2018 (Dec. 2018), pp. 1524–1533. ISSN: 1942-597X. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6371330/ (visited on 04/18/2020).

[74] Rebecca L. Siegel, Kimberly D. Miller, and Ahmedin Jemal. “Cancer statistics, 2020”. en. In: *CA: A Cancer Journal for Clinicians* 70.1 (Jan. 2020), pp. 7–30. ISSN: 0007-9235, 1542-4863. DOI: 10.3322/caac.21590. URL: https://onlinelibrary.wiley.com/doi/abs/10.3322/caac.21590 (visited on 04/15/2020).

[75] David Silver et al. “Mastering the game of Go without human knowledge”. In: *Nature* 550 (2017), pp. 354–359.

[76] Bharat Singh, Rajesh Kumar, and Vinay Singh. “Reinforcement learning in robotic applications: a comprehensive survey”. In: *Artificial Intelligence Review* 55 (Feb. 2022). DOI: 10.1007/s10462-021-09997-9.

[77] Brett R South et al. “Developing a manually annotated clinical document corpus to identify phenotypic information for inflammatory bowel disease”. en. In: *BMC Bioinformatics* 10.Suppl 9 (2009), S12. ISSN: 1471-2105. DOI: 10.1186/1471-2105-10-S9-S12. URL: http://www.biomedcentral.com/1471-2105/10/S9/S12 (visited on 01/27/2018).

[78] Amber Stubbs. “MAE and MAI: Lightweight Annotation and Adjudication Tools”. en-us. In: June 2011, pp. 129–133. URL: https://www.aclweb.org/anthology/W11-0416 (visited on 02/24/2020).

[79] Richard Sutton. “Learning to Predict by the Method of Temporal Differences”. In: *Machine Learning* 3 (Aug. 1988), pp. 9–44. DOI: 10.1007/BF00115009.

[80] Richard S Sutton, Hamid Maei, and Csaba Szepesvári. "A Convergent O(n) Temporal-difference Algorithm for Off-policy Learning with Linear Function Approximation". In: *Advances in Neural Information Processing Systems*. Ed. by D. Koller et al. Vol. 21. Curran Associates, Inc., 2008. URL: `https://proceedings.neurips.cc/paper/2008/file/e0c641195b27425bb056ac56f8953d24-Paper.pdf`.

[81] Richard S Sutton et al. "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in Neural Information Processing Systems*. Ed. by S. Solla, T. Leen, and K. Müller. Vol. 12. MIT Press, 1999. URL: `https://proceedings.neurips.cc/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf`.

[82] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. URL: `http://incompleteideas.net/book/the-book-2nd.html`.

[83] Richard S. Sutton et al. "Fast Gradient-Descent Methods for Temporal-Difference Learning with Linear Function Approximation". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 993–1000. ISBN: 9781605585161. DOI: `10.1145/1553374.1553501`. URL: `https://doi.org/10.1145/1553374.1553501`.

[84] Gerald Tesauro. "TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play". In: *Neural Computation* 6.2 (1994), pp. 215–219. DOI: `10.1162/neco.1994.6.2.215`.

[85] Philip S. Thomas and Emma Brunskill. "Data-Efficient off-Policy Policy Evaluation for Reinforcement Learning". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, 2016, pp. 2139–2148.

[86] Cameron Voloshin, Hoang Minh Le, and Yisong Yue. "Empirical Analysis of Off-Policy Policy Evaluation for Reinforcement Learning". In: 2019.

[87] Yanshan Wang et al. "Clinical information extraction applications: a literature review". In: *Journal of biomedical informatics* 77 (2018), pp. 34–49.

[88] Christopher Watkins and Peter Dayan. "Technical Note: Q-Learning". In: *Machine Learning* 8 (May 1992), pp. 279–292. DOI: `10.1007/BF00992698`.

[89] Yifan Wu, George Tucker, and Ofir Nachum. "Behavior Regularized Offline Reinforcement Learning". In: (Nov. 2019).

[90] Yongqin Xian et al. "Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly". In: *IEEE transactions on pattern analysis and machine intelligence* 41.9 (2018), pp. 2251–2265.

[91] Jun Xu et al. “Applying a deep learning-based sequence labeling approach to detect attributes of medical concepts in clinical text”. en. In: *BMC Medical Informatics and Decision Making* 19.5 (Dec. 2019), p. 236. ISSN: 1472-6947. DOI: `10.1186/s12911-019-0937-2`. URL: `https://doi.org/10.1186/s12911-019-0937-2` (visited on 04/18/2020).

[92] Adam Yala et al. “Using machine learning to parse breast pathology reports”. eng. In: *Breast Cancer Research and Treatment* 161.2 (2017), pp. 203–211. ISSN: 1573-7217. DOI: `10.1007/s10549-016-4035-1`.

[93] Wen-wai Yim et al. “Natural Language Processing in Oncology: A Review”. en. In: *JAMA Oncology* 2.6 (June 2016), p. 797. ISSN: 2374-2437. DOI: `10.1001/jamaoncol.2016.0213`. URL: `http://oncology.jamanetwork.com/article.aspx?doi=10.1001/jamaoncol.2016.0213` (visited on 07/22/2019).

[94] Bin Yu and Karl Kumbier. “Veridical data science”. In: *Proceedings of the National Academy of Sciences* 117.8 (2020), pp. 3920–3929. DOI: `10.1073/pnas.1901326117`.

[95] Bianca Zadrozny and Charles Elkan. *Transforming Classifier Scores into Accurate Multiclass Probability Estimates.* 2002.

[96] Ruiqi Zhang et al. “Off-Policy Fitted Q-Evaluation with Differentiable Function Approximators: Z-Estimation and Inference Theory”. In: (Feb. 2022).

Appendix A

Appendix

SUPPLEMENTAL NOTES

Corpus Statistics

The prostate cancer reports have a mean length of 912 words and 22382 unique words in the vocabulary. After replacing rare words with the special <UNK> token, there are 6803 unique words in the vocabulary. We include the percentage of reports containg each data element in supplemental table 6.

Text Preprocessing Methods

We removed commas, backslashes, semi-colons, tildes, periods, and the word "null" from each report in the corpus. For colons, forward slashes, parentheses, plus, and equal signs we added a space before and after the character, because these frequently had semantic value important for the classification task, for example colons typically occur in the synoptic comment where most of the relevant is contained, so if an n-gram has a colon in it, then this is indicative that it may contain important information. Each report was preprocessed so that every token in the report had one space preceding and one space following it. We then created a vocabulary of words using only reports in the traing corpus. If a token occurred fewer than 10 times in the traing corpus vocabulary, it was replaced with a "<UNK>" token for every report.

Pathologic Stage Information

We applied the token extractor methods rather than the classifiers to the pathologic stages, even though the number of classes for each stage type is small. The reason is the way the pathologic stages are encoded in the reports. For example, each pathologic stage is encoded in a single token, such as *pt2an0m0.* The letters *t, n,* and *m* denote the pathologic stage type, while *2a* and *0s* that follow immediately denote the class for each stage type. Classifier methods are not well suited for this task, because the number of possible token encodings is large (equal to the number of classes for the t-stage multiplied by the number of classes for the n and m-stages). It may even be the case that a certain encoding may show up in the test set and does not show up in the train set if a certain combination of the stages is new. For this reason, we first extract the stage token (eg *pt2an0m0*) and then determine the values for the *t, n,* and *m* stages using a regular expression rule-based method.

Model hyperparameters

We sampled regularization values from -6 to 6 in logspace for logistic regression models. For the random forest, we varied parameters such as bootstrapping or not, the max depth of each decision tree from 10 to 50 in increments of 10, the number of trees from 200 to 1000 in increments of 200, and the minimum number of samples per leaf from 2 to 64 by factors of two. For the support vector machines method, we sampled the regularization parameter from -6 to 6 in logspace, the kernel from either linear or radial basis kernel, and the gamma parameter from -3 to 1 in logspace. For the boosting classifier we sampled the learning rate from -4 to 1 in logspace using the SAMME.R algorithm in scikit-learn.

For the convolutional neural network, we sampled the learning rate from -6 to -1 in logspace, the number of convolutional filters from 50 to 400 in increments of 50, drop out parameter from 0, 0.125, 0.25, and 0.5, batch sizes from 16 to 32, and filter sizes from 3 to 6. For the LSTM, we sampled the learning rate from -6 to -1 in logspace, the drop out from 0, 0.125, 0.25, and 0.5, the hidden dimension size from 50 to 300 in increments of 50.

SUPPLEMENTAL TABLES

Supplemental Table 1. Macro F1 scores for classification fields across on full traing data sample (n = 2,066)

Data Element	**LSTM**	**Logistic regression**	**Adaboost classifier**	**SVM**	**Random Forest**
Gleason Grade - Primary	0.329	0.864	0.865	0.751	0.624
Gleason Grade - Secondary	0.300	0.680	0.561	0.649	0.536
Gleason Grade - Tertiary	0.268	0.692	0.771	0.492	0.385
Tumor histology	0.498	0.496	0.499	0.498	0.499
Cribriform pattern	0.583	0.493	0.807	0.593	0.493
Treatment effect	0.492	0.620	0.496	0.496	0.496
Tumor margin status	0.553	0.932	0.953	0.902	0.864
Benign margin status	0.489	0.593	0.701	0.496	0.972
Perineural invasion	0.600	0.942	0.978	0.927	0.936
Seminal vesicle invasion	0.550	0.888	0.946	0.884	0.876
Extraprostatic extension	0.555	0.898	0.953	0.856	0.687
Lymph node status	0.550	0.657	0.636	0.650	0.657
Mean Macro F1 across classification data elements	**0.481**	**0.730**	**0.764**	**0.683**	**0.669**

LSTM: Long Short-Term Memory Neural Network
CNN: Convolutional Neural Network
SVM: Support Vector Machine

Supplemental Table 2. Micro F1 scores for classification fields across on full traing data sample (n = 2,066)

Data Element	**LSTM**	**CNN**	**Logistic regression**	**Adaboost classifier**	**SVM**	**Random Forest**
Gleason Grade - Primary	0.604	0.981	0.978	0.972	0.935	0.947
Gleason Grade - Secondary	0.577	0.969	0.959	0.947	0.913	0.919
Gleason Grade - Tertiary	0.750	0.935	0.925	0.935	0.907	0.876
Tumor histology	0.993	0.997	0.984	0.997	0.993	0.996
Cribriform pattern	0.972	0.975	0.975	0.981	0.975	0.975
Treatment effect	0.972	0.981	0.981	0.984	0.987	0.987
Tumor margin status	0.645	0.951	0.941	0.953	0.919	0.891
Benign margin status	0.959	0.981	0.975	0.966	0.987	0.969
Perineural invasion	0.614	0.972	0.944	0.978	0.929	0.938
Seminal vesicle invasion	0.787	0.975	0.941	0.975	0.941	0.941
Extraprostatic extension	0.753	0.960	0.953	0.953	0.941	0.901
Lymph node status	0.824	0.988	0.984	0.953	0.975	0.984
Mean Micro F1 across classification data elements	0.788	0.972	0.962	0.966	0.950	0.944

LSTM: Long Short-Term Memory Neural Network
CNN: Convolutional Neural Network
SVM: Support Vector Machine

Supplemental Table 3. Mean macro F1 scores and standard deviations across columns for classification models on varying number of reports (N) across 5 runs

Model	**N = 16**	**N = 32**	**N = 64**	**N = 128**	**N = 256**
Logistic	0.434 ± 0.121	0.495 ± 0.116	0.532 ± 0.129	0.586 ± 0.154	0.630 ± 0.170
AdaBoost	0.475 ± 0.133	0.545 ± 0.155	0.590 ± 0.169	0.620 ± 0.182	0.658 ± 0.183
Random forest	0.445 ± 0.148	0.476 ± 0.135	0.508 ± 0.143	0.529 ± 0.134	0.562 ± 0.140
SVM	0.420 ± 0.141	0.444 ± 0.147	0.460 ± 0.148	0.503 ± 0.131	0.531 ± 0.161
CNN	0.386 ± 0.159	0.416 ± 0.162	0.460 ± 0.173	0.504 ± 0.171	0.577 ± 0.174
LSTM	0.387 ± 0.135	0.408 ± 0.128	0.405 ± 0.131	0.417 ± 0.126	0.430 ± 0.127

Supplemental Table 4. Mean micro F1 scores and standard deviations across columns for classification models on varying number of reports (N) across 5 runs

Model	**N = 16**	**N = 32**	**N = 64**	**N = 128**	**N = 256**
Logistic	0.808 ± 0.164	0.851 ± 0.123	0.880 ± 0.089	0.913 ± 0.060	0.937 ± 0.041
AdaBoost	0.845 ± 0.131	0.883 ± 0.098	0.907 ± 0.067	0.931 ± 0.050	0.949 ± 0.034
Random forest	0.828 ± 0.148	0.860 ± 0.117	0.888 ± 0.088	0.901 ± 0.077	0.913 ± 0.065
SVM	0.764 ± 0.202	0.774 ± 0.205	0.798 ± 0.179	0.853 ± 0.106	0.868 ± 0.138
CNN	0.722 ± 0.216	0.763 ± 0.180	0.786 ± 0.183	0.852 ± 0.122	0.901 ± 0.077
LSTM	0.691 ± 0.208	0.740 ± 0.186	0.740 ± 0.207	0.768 ± 0.182	0.778 ± 0.166

Supplemental Table 5. Counts for each outcome of error analysis for each field from 10 randomly sampled errors (or all errors if fewer than 10 errors were made)

Field	**Annotation Error**	**Model error**	**Report anomaly**	**Evaluation Error**	**Not reported in text**	**Total**	**Best model**
Gleason Grade - Primary	3	6	1	0	0	10	Cnn
Gleason Grade - Secondary	4	5	1	0	0	10	Cnn
Gleason Grade - Tertiary	4	6	0	0	0	10	Cnn
Tumor histology	0	3	0	0	0	3	Cnn
Cribriform pattern	3	1	0	0	0	4	Cnn
Treatment effect	3	3	1	0	0	7	Cnn
Tumor margin status	1	1	0	0	0	2	Boost
Benign margin status	0	1	0	0	0	1	Svm
Perineural invasion	3	6	1	0	0	10	Boost
Seminal vesicle invasion	1	8	1	0	0	10	Cnn
Extraprostatic extension	4	5	1	0	0	10	Cnn
Lymph node status	2	8	0	0	0	10	Cnn
Prostate weight	3	3	0	2	2	10	Boost
Estimated volume of tumor	2	1	0	3	4	10	Random forest
Pathologic T-stage	3	0	1	0	6	10	Logistic

Supplemental Table 6. Number of annotated data elements by number of reports used in the traing set

Field	**Percent of reports containg field**

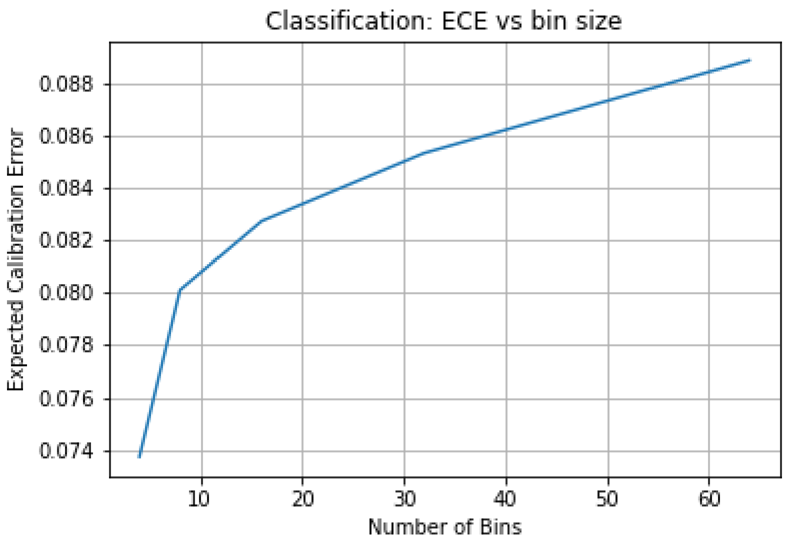
continued on next page

continued from previous page

Gleason Grade - Primary	0.99
Gleason Grade - Secondary	0.99
Gleason Grade - Tertiary	0.15
Tumor histology	1.00
Cribriform pattern	1.00
Treatment effect	1.00
Tumor margin status	1.00
Benign margin status	1.00
Perineural invasion	1.00
Seminal vesicle invasion	1.00
Extraprostatic extension	1.00
Lymph node status	0.99
Prostate weight	0.98
Estimated volume of tumor	0.98
Pathologic T-stage	0.95
Pathologic M-stage	0.94
Pathologic N-stage	0.38

SUPPLEMENTAL FIGURES

Supplemental Figure 1. Expected Calibration Error as a function of the bin size for boosting classification model averaged across fields



Supplemental Figure 2. Expected Calibration Error as a function of bin size for random forest extractor models averaged across fields

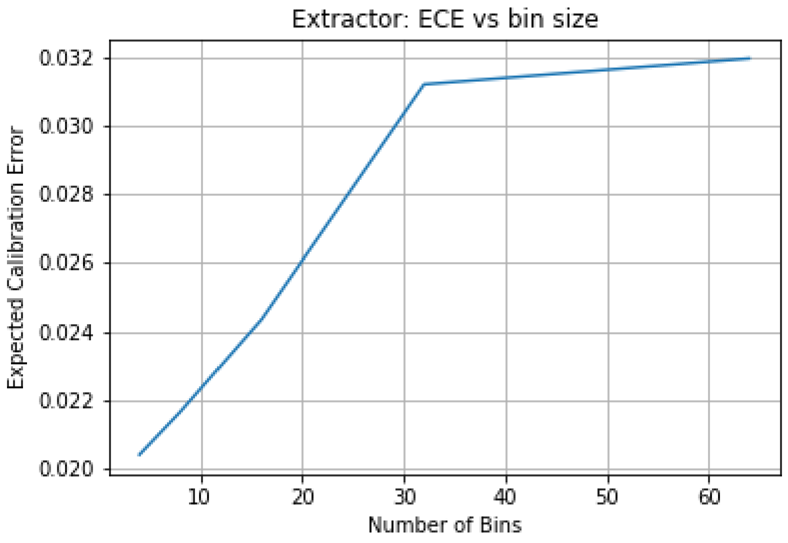


Figure 1A: Average pre-calibration and post-calibration ECE estimates across transfer learning variations of the two-stage method as a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports. The results presented include the mean performance across 10 random splits of the data and 95% confidence intervals. The two-stage method with transfer learning for both the line and final classifiers consistently outperforms other variations of the two-stage method including no transfer. Calibrating the models via isotonic regression is shown to reduce ECE scores across all methods.

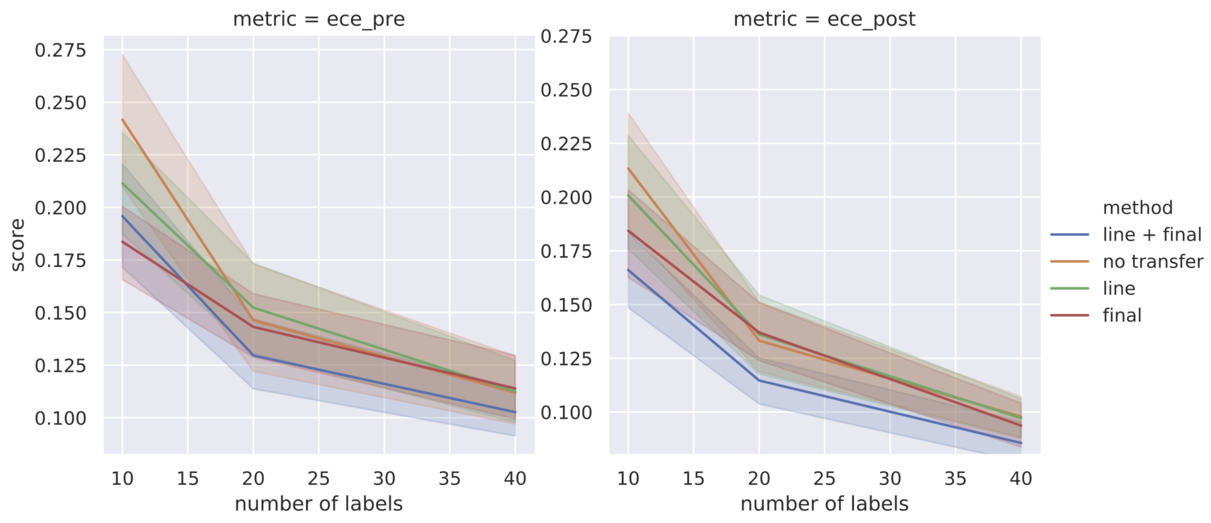


Figure 2A: Average pre-calibration and post-calibration ECE estimates across baseline methods as a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports. The results presented include the mean performance across 10 random splits of the data and 95% confidence intervals for the shared field and shared labels case. Compared to the transfer learning variations of the two-stage method, the baselines are less calibrated.

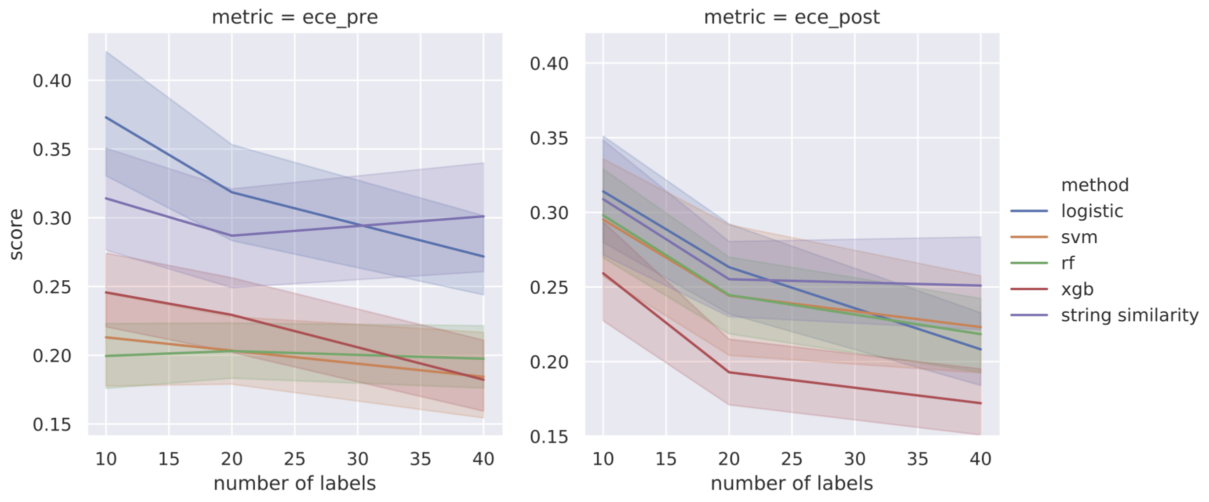
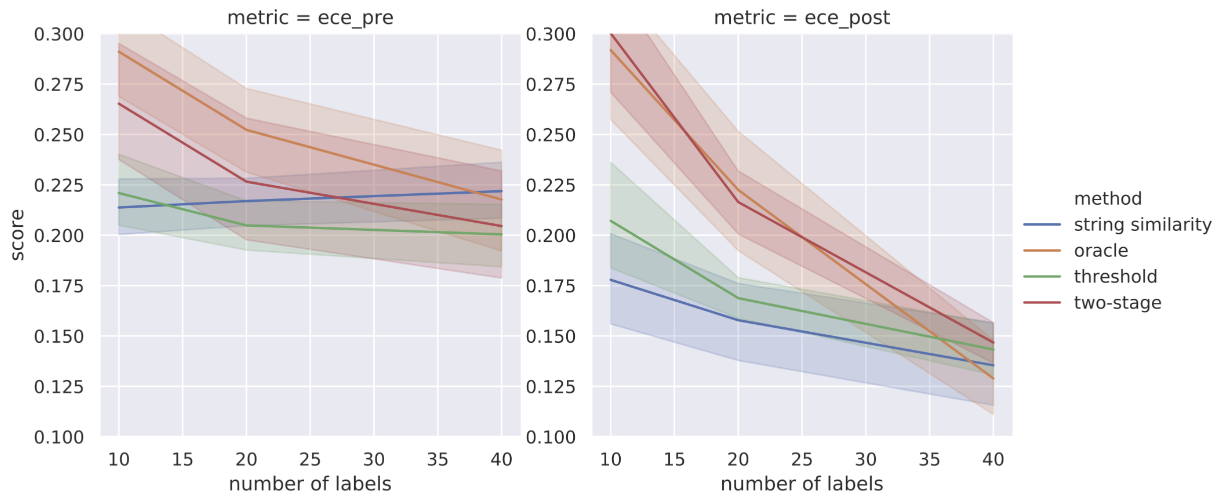


Figure 3A: Average pre-calibration and post-calibration ECE estimates across string similarity variations of the two-stage method as a function of 10, 20, and 40 labeled examples on colon, kidney, and lung cancer pathology reports. The results presented include the mean performance across 10 random splits of the data and 95% confidence intervals for the shared field and unique labels case. The string similarity variations consistently outperform the two-stage method using the ECE metric. Surprisingly the string similarity method in isolation has lower ECE after calibration than the other string similarity variations of the two-stage approach.


metric = ece_pre
metric = ece_post
score
number of labels
method
string similarity
oracle
threshold
two-stage