Software package for regression algorithms based on Gaussian Conditional Random Fields

Tijana Markovic*, Vladan Devedzic†, Fang Zhou‡, Zoran Obradovic§
*School of Innovation, Design and Engineering, Malardalen University, Vasteras, Sweden
†Faculty of Organizational Sciences University of Belgrade, Belgrade, Serbia
‡School of Data Science and Engineering, East China Normal University, Shanghai, China
§Center for Data nalytics and Biomedical Informatics Temple University, Philadelphia, US
tijana.markovic@mdu.se, devedzic.vladan@fon.bg.ac.rs, fzhou@dase.ecnu.edu.cn, zoran.obradovic@temple.edu

***bstract*—The Gaussian Conditional Random Fields (GCRF) algorithm and its extensions are used for machine learning regression problems in which the attributes of objects and the correlation between objects should be considered when making predictions. These algorithms can be applied in different domains where problems can be seen as graphs, but their implementation requires complex calculations and good programming skills. This paper presents an open source software package that includes a tool with graphical user interface (GCRFs tool) and Java library (GCRFs library). GCRFs tool is software that integrates various GCRF-based algorithms and supports training and testing of those algorithms on real-world datasets. The main goal of GCRFs tool is to provide a straightforward and user-friendly graphical user interface that will simplify the use of GCRF-based algorithms. GCRFs Java library contains basic classes for GCRF concepts and can be used by researchers who have experience in Java programming. lso, this paper presents the results of a pilot usability evaluation of the GCRFs tool, where the software was evaluated with expert and non-expert users. This evaluation gave us detailed insight into the experiences and opinions of the users and helped us outline priorities for future development.**

Keywords: Gaussian Conditional Random Fields, Regression lgorithms, Graphs, Software, Library

I. Introduction

Machine Learning (ML) is a sub-area of artificial intelligence, where algorithms have the ability to learn from experience [1]. Regression ML algorithms investigate the relationship between independent variables (attributes) and a dependent variable (output) with goal to learn how to predict the output which is a continuous value. Structured regression ML algorithms are designed to use relationships between objects to predict output variables. In other words, structured regression algorithms consider the attributes of objects and the dependencies between objects to make predictions as accurately as possible. Traditional ML algorithms, such as neural networks, use only information contained in attributes ($\mathbf{x}$) to predict the output variable (y), whereas structured regression algorithms use dependencies between outputs to improve final predictions. Problems that can be solved using structured regression can be seen as graphs, where nodes correspond to objects with attributes ($\mathbf{x}$) and outputs (y), while relationships between nodes are application-specific and defined in advance, either by domain knowledge or by assumptions. For example, relationships between hospitals can be based on similarity of their specialization [2], relationships between pairs of scientific papers can be presented as the similarity of sequences of citation [3], relationships between people can be quantified based on strength of their friendship [4], etc.

The Gaussian Conditional Random Fields (GCRF) [5] is a structured regression algorithm that incorporates the outputs of traditional ML algorithms (unstructured predictor) and the correlation between the output variables to achieve a higher prediction accuracy. It was first applied in computer vision [6], but since then it has been applied in different areas and extended for various purposes. The number of projects that use GCRF-based algorithms could be significantly higher, but implementing them is not easy, and there is no library that can facilitate the implementation process.

In this paper, we present the open-source software package[1] that includes a tool with graphical user interface (GCRFs tool) and a Java library (GCRFs library). GCRFs tool is software that integrates various GCRF-based algorithms and supports training and testing of those algorithms on real-world data from different domains without writing code. The main goal of this tool is to provide a simple and user-friendly Graphical User Interface (GUI) for GCRF-based algorithms and to simplify the use of those algorithms for non-expert users. Users can easily import their datasets and apply different algorithms. In addition, researchers in the field of machine learning can use this software to easily run GCRF-based algorithms on their datasets to make comparisons in terms of accuracy and execution time. GCRFs library can be used by researchers who have experience in Java programming. This library contains basic classes for GCRF concepts and implementation of some concrete algorithms. It has an intuitive and simple programming interface (PI) and is very flexible and extensible.

The remainder of the paper is organized as follows. Section II introduces the GCRF algorithm and various GCRF extensions that are integrated into the GCRF tool. Section III presents the functional and technical description of the GCRFs tool, and the analysis of time consumption. In addition, this section contains a short description of the pilot usability evaluation study and its results. Section IV introduces the Java GCRFs library, and Section V concludes the paper.

[1] https://gcrfs-tool.com/

978-1-6654-6283-9/22/$31.00 ©2022 IEEE
DOI 10.1109/ICMLA55696.2022.00184

II. B CKGROUND

. *GCRF algorithm*

The Gaussian Conditional Random Fields (GCRF) [5] [6] algorithm incorporates the outputs of unstructured predictors and the correlation between the output variables to achieve higher prediction accuracy. main assumption is that if two objects are closely related, they should be more similar to each other and they should have similar values of the output variable.

In a Conditional Random Field (CRF) algorithm [7], the observable attributes $\mathbf{x}$ interact with each of the outputs y_i directly and independently of each other. The CRF probability function can be represented by the Eq. (1), where and I are real valued functions that are known in CRF literature as association (Eq. (2)) and interaction (Eq. (3)) potential. K unstructured predictors are used to predict a single output y_i and L similarity functions are used to represent different types of dependencies between nodes. The larger the value of , the more y_i is related to the attributes $\mathbf{x}$. The larger the value of I, the more y_i is related to y_j.

$$P\ \mathbf{y}|\mathbf{x}) = \frac{1}{Z\ \mathbf{x},\alpha,\beta)} \exp\ \sum_{i=1}^{N} \quad \alpha,y_i,\mathbf{x}) + \sum_{j\sim i} I\ \beta,y_i,y_j)\big), \quad (1)$$

$$\alpha,y_i,\mathbf{x}) = \quad \sum_{k=1}^{K} \alpha_k\ y_i \quad R_{i,k}\ \mathbf{x}))^2, \quad (2)$$

$$I\ \beta,y_i,y_j) = \quad \sum_{l=1}^{L} \beta_l S^l_{ij}\ y_i \quad y_j)^2, \quad (3)$$

The GCRF is a CRF model with both quadratic feature and quadratic interaction functions that can be transposed directly onto a Gaussian multivariate probability distribution (Eq. (4))

$$P\ \mathbf{y}|\mathbf{x}) = \frac{1}{2\pi|\ |^{1/2}} exp \quad \frac{1}{2}\ \mathbf{y} \quad \mu)^T Q\ \mathbf{y} \quad \mu)) \quad (4)$$

When setting these two conditional probability models equal to each other, we get a precision matrix (Q) defined in terms of the confidence of the input predictors and the pairwise interaction structure, measured by α and β respectively. The learning task is to choose the parameters α and β to maximize the conditional log-likelihood of the set of training examples. The precision matrix is calculated by the Eq. (5), where L_j is the Laplacian matrix of j^{th} similarity matrix (S).

$$Q = \sum_k \alpha_k I + \sum_j \beta_j L_j, \quad (5)$$

Representing the input predictions as a matrix R, the formula for the final prediction can be concisely written as in Eq. (6).

$$\mu = Q\ ^1 R\alpha \quad (6)$$

GCRF has been used in a broad set of applications: climate [5], [8], [9], energy forecasting [10], [11], healthcare [12], [2], speech recognition [13], computer vision [14], [15], etc.

B. GCRF extensions

GCRF is extended for various purposes, and following extensions of GCRF are integrated in GCRFs tool:

- **Directed GCRF (DirGCRF)** [4] extends the GCRF algorithm by considering asymmetric similarity, which means that it can be applied on directed graphs. In many real-world networks, objects are asymmetrically linked, and standard GCRF could not be directly applied on these problems since this method requires a symmetric similarity matrix. In [4] DirGCRF is applied on the Teenagers [16] dataset that consists of three temporal observations of 50 teenagers. The friendship network (up to 12 best friends) and teenager's alcohol consumption (ranging from 1 to 5) are provided for each time point. The goal was to predict alcohol consumption at observation time point 3, based on two previous observations. The results showed that the DirGCRF has 17% higher accuracy than standard GCRF, and 4% higher accuracy than neural network.
- **Unimodal GCRF (UmGCRF)** [17] algorithm extends the GCRF parameter space to facilitate joint modeling of positive and negative influences. GCRF is restricted to positive weights, and UmGCRF expands the parameter search space to allow for negative links and negative influence of unstructured predictors. UmGCRF was evaluated on the problem of predicting monthly hospital admissions for 189 classes of diseases in California [18]. Relationships between diseases were based on their similarity (on a scale of zero to one) created using the disease-symptom similarity network built in [19]. UmGCRF had 17% and 12% improvement in accuracy over the input baselines. lso, the new mathematical formulation caused a huge increase in speed, far exceeding the speed and scalability of standard GCRF.
- **Marginalized GCRF (m-GCRF)** [20] deals with missing labels in partially observed temporal attributed graphs. common problem in a real-life applications is that the large fraction of observations is often missing. This method extends GCRF to naturally handle missing labels, rather than expecting the missing data to be treated in a preprocessing stage. The benefits of the m-GCRF are demonstrated on a challenging application for predicting precipitation based on partial observations of climate variables in a temporal graph that spans the entire continental US [21]. Each temporal graph has 1218 nodes (meteorological stations). The spatial information is used for calculating similarities (correlations) between stations, and for each station participation and 6 more attributes are provided. There are no missing values in input variables, but about 5% of the dependent variables (precipitations) are missing. Experiments on this data provided an evidence that m-GCRF brings accuracy improvement (5% versus neural network).
- **Uncertainty Propagation GCRF (up-GCRF)** [22] is an algorithm for propagating uncertainty in temporal graphs

by modeling noisy inputs. It is aimed to support structured regression for long-term decision making, which has been of interest in many high impact applications. The up-GCRF method takes into account uncertainty that comes from the data when estimating uncertainty of the predictions. The up-GCRF was evaluted on the California HCUP data [18]. Problem considered in this study is long-term prediction of admission and mortality rate based on inpatient discharge data. In all experiments the up-GCRF outperformed baselines in terms of both accuracy and uncertainty propagation.

- **Representation Learning based Structured Regression (RLSR)** [23] simultaneously learns hidden representation of objects and relationships among outputs. The objective of the algorithm is to improve the representational power of the standard GCRF by introducing hidden variables that are nonlinear functions of the input variables. Such a method can learn more informative representations and structural dependencies simultaneously. One of RLSR applications is prediction of the daily solar energy income at 98 Oklahoma Mesonet sites. On this dataset, the RLSR outperformed all baselines by at least 50%.

III. GCRFs TOOL

. *Overview of the system*

The GCRFs tool[2] can be used to train and test various GCRF-based structured regression algorithms on datasets from different domains. The common procedure for solving GCRF-based structure regression problems includes the following steps:

1) prepare data for training and testing
2) select and train an unstructured predictor
3) train one of the GCRF-based algorithms
4) test selected algorithm
5) obtain predicted values
6) calculate accuracy

Users provide a dataset for their specific problem at step 1) and GCRFs tool will finish all remaining structure regression steps. Fig. 1 presents an overview of the system architecture of the GCRFs tool.

The GCRFs tool provides dataset samples, but users can also add their own dataset. The software trains the parameters of the selected algorithm on the given dataset, and the parameter values are stored on the local file system and used to test the algorithm. The training process includes the training of unstructured predictors. fter the testing process, users can get predicted values, accuracy, and execution time.

To calculate the regression accuracy of all methods, we used coefficient of determination (R^2) that measures how closely the output of the model matches the actual value of the data. score of 0 indicates a poor match, while a score of 1 indicates a perfect match. Some very bad predictors can be characterized by a negative coefficient of determination.

[2]https://gcrfs-tool.com/about

Different unstructured predictors can be incorporated in the GCRF algorithm, and this software integrates following:

- **Neural Network (NN)** - Neurons in an artificial NN are grouped in layers: input layer, output layer, and one or more hidden layers. In the GCRFs tool, the number of neurons in the input layer is the same as the number of attributes in the chosen dataset, while the number of neurons in the output layer is 1 since we have only one predicted variable. There is one hidden layer, and the user is asked to insert the number of neurons in it.
- **Linear Regression (LR) or Multivariate Linear Regression (MLR)** - LR is an approach for modeling the relationship between a dependent variable y and one or more explanatory variables $\mathbf{x}$. This relationship is modeled using linear predictor functions whose parameters are estimated from the data. LR has only one explanatory variable, while MLR incorporates multiple explanatory variables. In the GCRFs tool, LR or MLR is selected based on the number of attributes in the dataset.

B. Functional description

To use the GCRFs tool, users should download a zipped file from the website[3] and extract it to the desired location. The user manual is included in the folder, but it can also be downloaded from the website. ll required libraries are included in the folder and there is no need to install any dependencies. The standard GCRF and DirCGRF can be used without M TL B, but if the user wants to use the remaining four algorithms, M TL B should be installed as well. When the tool is run for the first time, the *Configuration* panel will be displayed. In this panel, main parameters for all algorithms can be configured, and an optional connection with M TL B can be established. Most of the parameters have default values.

This software provides 7 dataset samples. lso, users can add their own datasets using the *dd dataset* option in the *Datasets* menu item. Data for GCRF-based methods are naturally modeled as graphs, where objects are represented as nodes, and relations are represented as edges between the nodes. The similarity matrix that quantifies the relationships among the nodes is denoted with S. Each node is characterized by one or more attributes ($\mathbf{x}$) and has one output variable (y). For each dataset, the user should provide *txt* files with edges, attributes, and outputs. The required formats of these files are described in the user manual. Once the new dataset is added, it will be stored in the Datasets folder and the files will be denoted as follows:

- *x txt* - attributes
- *y txt* - desired output
- *s txt* - edges (graph that presents relationships between objects).

Since some of the algorithms have the ability to learn similarity between nodes, a file with edges is not mandatory and "Learn similarity" option can be selected instead. In that case, only those algorithms can be applied to the specific dataset.

[3]http://gcrfs-tool.com/installation-and-user-manual/

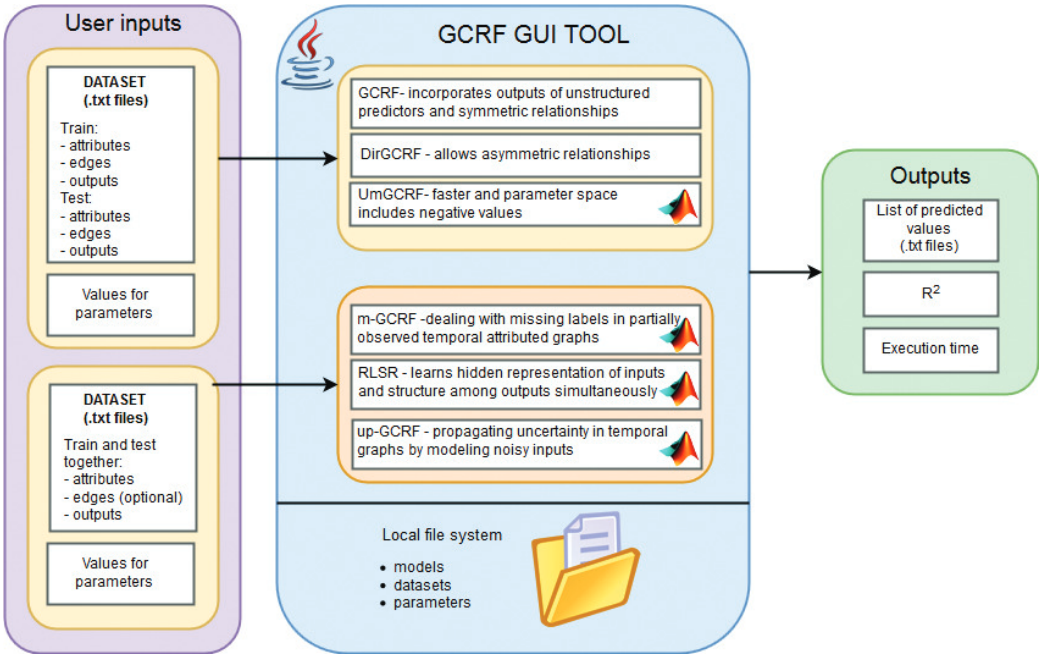

Fig. 1: System architecture of the GCRFs tool

ll datasets are listed in the *Manage datasets* option in the *Datasets* menu item.

In the *Help* menu item, users can find general information about the software, datasets and methods. For all datasets, *Help* provides an explanation of the nodes in that dataset, the connections between the nodes, the number of attributes each node has, what they represent, what should be predicted, etc. The *Methods* option in the *Help* menu item provides a basic description of each method (algorithm).

The main software functionalities can be presented throughout the two case studies:

- **Case study 1:** The Geostep [24] dataset consists of 50 treasure hunt games. Each game can have a maximum of 10 clues and each clue belongs to one of 4 categories (business, social, travel, and irrelevant). Each game has six attributes: the number of clues in each category, game privacy scope, and game duration. The training data contains 25 games that are randomly chosen, while the rest of the games are in the test data. The file with edges contains similarity between games, which is calculated based on domain knowledge. Two versions of this dataset are provided within the GCRFs tool, "Geostep symmetric" with asymmetric similarities, and "Geostep Symmetric" with symmetric similarities. The goal is to predict the probability that a game can be used for tourist purposes. In this case study, the "Train on networks" option is used to train the DirGCRF method on the "Geostep symmetric" dataset. Since this dataset has an asymmetric similarity matrix, DirGCRF is the only method that can be applied. NN is used as an unstructured predictor. dditionally, an option " pply standard GCRF" is selected, and the similarity matrix will be automatically converted from asymmetric to symmetric in order to apply standard GCRF and obtain its accuracy for comparison. fter clicking the "TR IN and TEST" button, the training and testing process will start, and the execution time and R^2 in the training and testing data will be displayed. The predicted values of the test data will be exported to a *txt* file. n example of this setup and training and testing results is presented in Fig. 2.
- **Case study 2:** The Energy[4] dataset consists of solar energy forecasting in Oklahoma Mesonet sites. The original dataset contains 15 attributes for 98 sites measured in 1600 time points. To reduce the dataset that will be included in the GCRFs tool, we randomly selected 10 sites, and for each site we extracted only one attribute for all 1600 time points. The file with edges is not provided in this dataset, which means that the selected method should be able to learn similarities between sites. The goal is to predict the total daily incoming solar energy at these sites. This problem can be modeled as a temporal graph, but the structural dependencies (edges) should be learned simultaneously with the training phase. In this case study, the "Train and test on temporal networks" option is used to train and test the RLSR method on the "Energy" dataset. It is required to split the dataset on training, validation and testing subset, and in this example we are using 1000-300-300 time points, respectively. For all other parameters, default values are used. Since this datasets does not contain similarity information, the

[4]https://www.kaggle.com/c/ams-2014-solar-energy-prediction-contest

"Learn similarity" option is automatically checked. When the "TR IN and TEST" button is clicked, M TL B is automatically called since the RLSR method is implemented in M TL B. When M TL B completes all calculations, the results from the training and testing phase will be provided, and the predicted values for the test data will be exported to a *txt* file. n example of this setup is presented in Fig. 3.

C. Technical description

The GCRFs tool was implemented in Java, using the Eclipse integrated development environment (IDE)[5]. This is an open-source project, and the code is publicly available on GitHub[6]. GUI was implemented using components from the Swing GUI toolkit. The tool is installed and run locally, so its performance is not influenced by the number of users accessing the tool.

The following main functions are implemented in Java:

- gradient descent algorithm that is used to train GCRF-based algorithms
- unstructured predictors (neural network, linear regression and multivariate linear regression)
- random graph generator
- two algorithms (GCRF and DirGCRF)

The remaining four algorithms (UmGCRF, m-GCRF, RLSR, up-GCRF) are implemented in M TL B. These are called from Java, the input arguments for different functions are automatically passed, and the output values are automatically collected. Each time the user chooses any of the M TL B methods, GCRFs tool launches and then controls a M TL B session without any user intervention. When the process is completed, the session is automatically closed.

The Java libraries that have been used include: library for matrix calculations (Oj lgo[7]), library for the implementation of neural networks (Neuroph[8]), and libary for calling M T-L B from Java (M TL Bcontrol[9]).

D. Time consumption

The scalability of the tool and the running behavior of different methods were assessed on different datasets with varying numbers of nodes: 100, 500, 1000 and 5000. Time consumption varies depending on the methods chosen. ll experiments were run on Windows with 16GB R M memory and a 3.4GHz CPU. The time consumption is presented after 50 iterations, and the results are shown in Table I. We can see that models implemented in Java take more time due to Java's object-oriented nature, which requires more memory and more time to handle large matrix computations. On the other hand, M TL B has much more support for high-level mathematical operations, as well as built-in matrix operations, so it runs faster than Java. The advantage of using Java is that it can be downloaded for free, and almost all users already have Java installed on their machines, while M TL B is expensive, and including it on a computer can be very costly for users.

E. Usability evaluation

Usability [25] is a quality attribute that evaluates the ease with which user interfaces are used. The main usability attributes [26] are usefulness, efficiency, effectiveness, learnability, and satisfaction. The goal of usability testing is to collect empirical data while observing users using the software to perform realistic tasks and to utilize those data to improve the product and make it more useful and usable. There are various testing methodologies, and this chapter briefly describes the methodology used for GCRFs tool usability evaluation and the results of a pilot usability study.

In the first stage, GCRFs tool was tested by a group of internal experts, trying to identify design flaws, bugs, or any other problems that may occur during the use of the software. Some issues were identified, and the tool was updated and optimized. In the second stage, an evaluation was conducted with different types of end-user. Since the GCRFs tool should be intuitive and easy to use for both experts and beginners in the ML field, it was decided to test the software with two groups of users: experts and students. ll participants were asked to complete four tasks using this tool. Before getting the assignments, the participants received a short description of the system, without any instructions on the specific tasks. The technical requirement for all participants was to have a computer with Java 8 installed, and M TL B was optional.

To get a detailed insight into the experiences and opinions of users, the authors have created a questionnaire for evaluation[10] that participants were supposed to fill out once they had completed all tasks. The main purpose of the questionnaire is to collect information from participants to clarify and deepen understanding of the strengths and weaknesses of the software [26]. The questionnaire is designed to be easy for both the authors (to facilitate the analysis of the responses) and the participants (to minimize their time filling out the questionnaire). Participants are asked to check boxes, circle answers, or score statements on a scale of 1 to 5. The questionnaire contains only one open-ended question, which is not mandatory. These questions can help authors gather information about users' opinions and feelings about ease of use and ease of learning, as well as reveal their satisfaction with the software. The questionnaire has five parts, and those parts are briefly described together with the results of the pilot usability evaluation study:

1) **User profile** - This part of the questionnaire is used to collect basic information that will help the authors understand the backgrounds of users. During the pilot usability evaluation study, GCRFs tool was tested with 34 users, 12 experts, and 22 undergraduate and graduate students. Most of the participants were males (82%), from 20 to 30 years old (71%). When it comes to the education level, the majority of participants were

[5]https://eclipse.org/

[6]https://github.com/vujicictijana/GCRF_GUI_TOOL

[7]http://ojalgo.org/

[8]http://neuroph.sourceforge.net/

[9]https://code.google.com/archive/p/M TL Bcontrol/

[10]https://goo.gl/forms/1kVVMLc1lVe1eYU32/

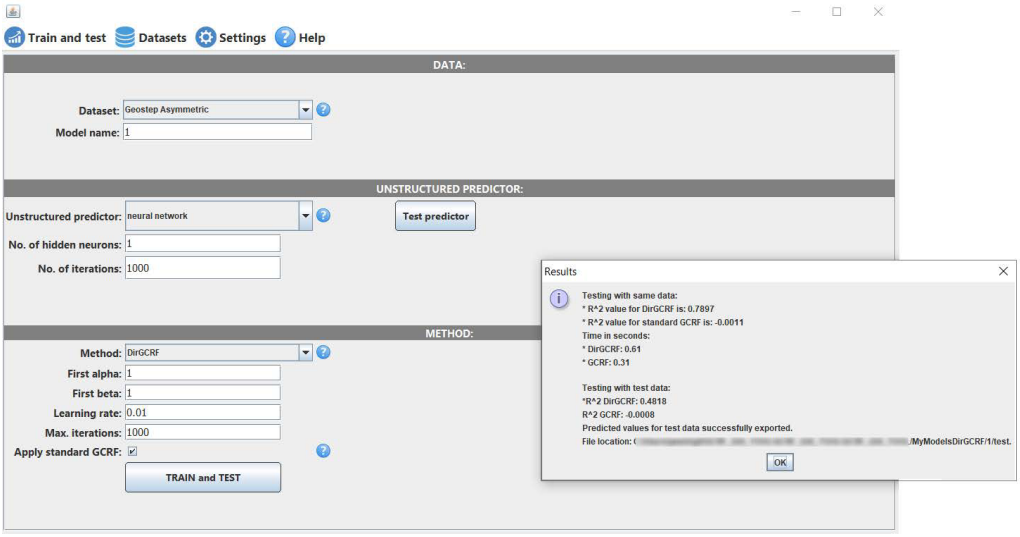

Fig. 2: Example of the use of GCRFs tool for Case study 1

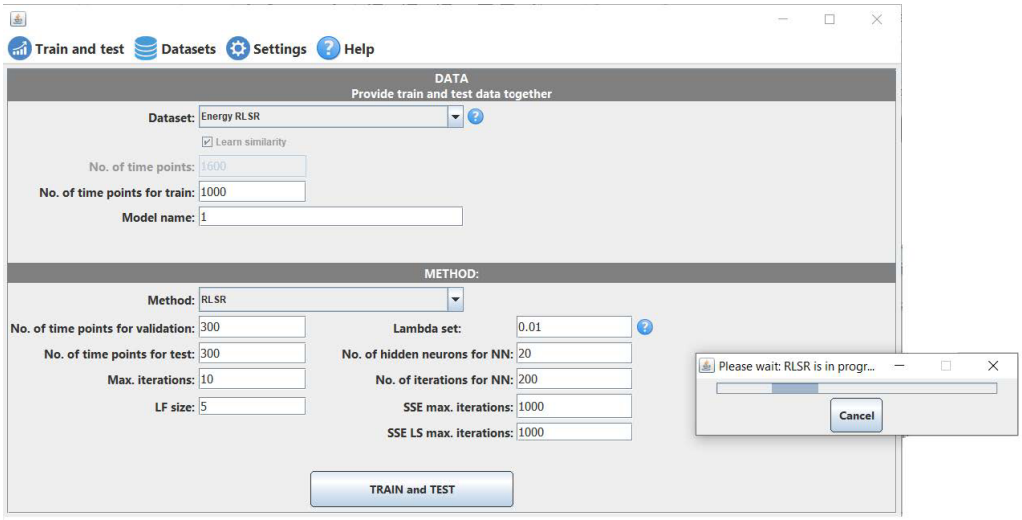

Fig. 3: Example of the use of GCRFs tool for Case study 2

T BLE I: Execution time of GCRFs tool for different methods with varying numbers of nodes in a graph

No. of nodes	No. of edges	GCRF	DirGCRF	UmGCRF	m-GCRF	up-GCRF	RLSR
100	5,094	0.27 s	0.17 s	6.62 s	8.49 s	26.84 s	69.25 s
500	127,540	16.98 s	9.49 s	7.45 s	17 s	6.58 min	8.25 min
1000	509,376	129.4 s	69.57 s	8 s	53.15 s	27.6 min	1h 18 min
5000	12,749,518	4h 45 min	2h 12 min	34.48 s	65 min	N/	N/

PhD students (56%), but we also had master (26%) and undergraduate students (18%), which means that different user profiles were included in this evaluation. We had participants with different pre-knowledge levels, 30% of the participants had no experience in ML, but 64% of participants were not familiar with structured regression. The participants have used different operating systems to test the software, and the majority of them had M TL B installed on their computer (76%).

2) **Level of tasks** - The main goal of this part of the questionnaire is to determine whether specific tasks are easy or difficult for different types of users. During the pilot usability evaluation study, all tasks had an average score of 4.4/5 or higher. It is important to mention that 80-94% of users have seen the tasks as "Very easy" or "Easy", while for 3-8% of them, the tasks have been "Very difficult" or "Difficult".
3) **Terminology and system information** - This part of the questionnaire assesses what users think about GUI and its components. During the pilot usability evaluation study, all statements had average scores greater than 4/5, which means that users are generally satisfied with the clarity of the terminology used. However, some of them would like better help, more detailed error messages and clearer input prompts.
4) **System Usability Scale** - This part of the questionnaire uses the System Usability Scale (SUS), which has proven to be a valuable evaluation tool and a reliable measure of system usability [27]. The results of the pilot usability evaluation study showed that 76% of the participants would like to use GCRFs tool, and that 82% think the software is easy to use. On the other hand, 12% of the participants think that the software is too complex and 6% think it is very cumbersome to use. 18% of participants would need the support of a technical person to be able to use this software, while 9% think that most people would not learn to use it very quickly.
5) **Comments/suggestions** - This part contains an open-ended question for users' opinions and suggestions. From the comments that the participants gave during the pilot usability evaluation study, we concluded that they found the first-time experience encouraging and that the majority of them think that the GCRFs tool is very simple and easy to use. The primary user feedback remarks were regarding the help and explanations in the software, so the first goal is to improve that aspect.

fter the pilot usability evaluation study, this questionnaire is published on the project website. t the time of submission of the paper the total number of completed surveys was 76 and the average grades were 4.2/5 for "Level of tasks" and 4.3/5 for "Terminology and system information", while the SUS grade was " " (82.3/100). The summary results of the questionnaire are updated in real time and publicly available on the website[11].

[11]http://gcrfs-tool.com/results/

IV. GCRFs J V LIBR RY

The GCRFs Java library is an open-source library and its code[12] and jar file[13] can be downloaded from GitHub and project website. Basic classes[14] provide a general structure and logic components that are common for GCRF-based algorithms:

- BasicCalcs - class that contains static methods for required mathematical calculations, such as vector and matrix multiplication, matrix trace, inverse matrix, etc.
- Calculations - interface that specifies a list of methods that should be implemented by all GCRF-based algorithms regarding calculation rules.
- CalculationsGCRF - class that implements Calculations interface using the rules of standard GCRF. Other GCRF-based algorithms should extend this class and override calculation rules that are different for the specific algorithm.
- Learning lgorithm - interface that specifies a list of methods to be implemented by the learning algorithm.
- Gradient scent - class that implements the Learning lgorithm using the rules of the gradient ascent.
- Parameters - class that specifies all parameters that are required by the gradient ascent algorithm.
- lgorithm - interface that specifies a list of methods that should be implemented by all GCRF-based algorithms.
- Basic - basic class for GCRF algorithm (implements lgorithm interface). ll GCRF-based algorithms should extend this class and specify its own calculation rules (class that implements Calculations interface) and learning algorithm (class that implements Learning lgorithm interface).

Different GCRF-based algorithms are implemented by extending basic classes and by overriding existing or adding additional methods. The library can be easily extended with new GCRF-based algorithms. The full list of packages, classes, and methods can be found in GCRFs library PI documentation[15].

V. CONCLUSIONS

This paper presents GCRFs tool, open-source software that integrates various GCRF-based algorithms and supports training and testing of those algorithms on real-world data from different domains. The main functionalities of software were briefly described and demonstrated through two case studies. lso, this paper presents open-source Java library than can be used to include existing GCRF-based algorithms in Java code and that can be easily extended with new GCRF-based algorithms. Since a very important aspect of the GCRFs tool is that it should be intuitive and easy to use, evaluation with experts and non-expert users has been conducted. Based on the overall evaluation results, we can conclude that both groups of

[12]https://github.com/vujicictijana/GCRFs_Library
[13]http://gcrfs-tool.com/use/
[14]http://gcrfs-tool.com/class-diagram/
[15]http://gcrfs-tool.com/api/

users were very satisfied with the software, and some of their suggestions are very useful for planning future development.

The main advantage of the proposed tool is that it provides users with the opportunity to test multiple GCRF-based algorithms without writing code, while the main limitation is that it is not fully free to use, since some of the algorithms require having a M TL B license.

s future work we would like to improve GCRFs tool according to the users' comments and to provide a web version to facilitate its usage. dditionally, we plan to upgrade the proposed software package with Phyton and M TL B libraries for GCRF-based methods.

CKNOWLEDGMENT

This work is supported in part by the U.S. NSF award CNS-212598, and the RL subaward 555080-78055 under Prime Contract No. W911NF2220001, and U.S. rmy Corp of Engineers Engineer Research and Development Center under Cooperative greement W9132V-22-2-0001, and Temple University office of the Vice President for Research 2022 Catalytic Collaborative Research Initiative Program I & ML Focus rea, and NSFC Grant 61902127. ny opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] S. Russell and P. Norvig, " rtificial Intelligence: Modern pproach," in *Malaysia: Pearson Education Limited.*, 2016.

[2] . Polychronopoulou and Z. Obradovic, "Hospital pricing estimation by gaussian conditional random fields based regression on graphs," in *Bioinformatics and Biomedicine (BIBM), 2014 IEEE International Conference on*. IEEE, 2014, pp. 564–567.

[3] J. Slivka, M. Nikolić, K. Ristovski, V. Radosavljević, and Z. Obradović, "Distributed gaussian conditional random fields based regression for large evolving graphs," in *Proc. 14th SI M Int'l Conf. Data Mining, Workshop on Mining Networks and Graphs*, 2014.

[4] T. Vujicic, J. Glass, F. Zhou, and Z. Obradovic, "Gaussian conditional random fields extended for directed graphs," *Machine Learning*, vol. 106, no. 9, pp. 1271–1288, 2017.

[5] V. Radosavljevic, S. Vucetic, and Z. Obradovic, "Continuous conditional random fields for regression in remote sensing." in *Proccedings of European Conference on rtificial Intelligence (EC I)*, 2010, pp. 809–814.

[6] C. Liu, E. H. delson, and W. T. Freeman, "Learning gaussian conditional random fields for low-level vision," in *Proc. of CVPR*. Citeseer, 2007, p. 7.

[7] J. Lafferty, . McCallum, F. Pereira *et al.*, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the eighteenth international conference on machine learning, ICML*, vol. 1, 2001, pp. 282–289.

[8] V. Radosavljevic, S. Vucetic, and Z. Obradovic, "Neural gaussian conditional random fields," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 614–629.

[9] N. Djuric, V. Radosavljevic, Z. Obradovic, and S. Vucetic, "Gaussian conditional random fields for aggregation of operational aerosol retrievals," *IEEE Geoscience and Remote Sensing Letters*, 2015.

[10] M. Wytock and J. Z. Kolter, "Sparse gaussian conditional random fields: lgorithms, theory, and application to energy forecasting." in *ICML (3)*, 2013, pp. 1265–1273.

[11] H. Guo, "Modeling short-term energy load with continuous conditional random fields," in *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, 2013, pp. 433–448.

[12] D. Gligorijevic, J. Stojanovic, and Z. Obradovic, "Improving confidence while predicting trends in temporal disease networks," in *4th Workshop on Data Mining for Medicine and Healthcare, SI M International Conference on Data Mining (SDM)*, 2015.

[13] S. Khorram, F. Bahmaninezhad, and H. Sameti, "Speech synthesis based on gaussian conditional random fields," in *rtificial Intelligence and Signal Processing*, 2014, pp. 183–193.

[14] M. F. Tappen, C. Liu, E. H. delson, and W. T. Freeman, "Learning gaussian conditional random fields for low-level vision," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8.

[15] S. Wang, L. Zhang, and R. Urtasun, "Transductive gaussian processes for image denoising," in *Computational Photography (ICCP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–8.

[16] L. Michell and . mos, "Girls, pecking order and smoking," *Social science & medicine*, vol. 44, no. 12, pp. 1861–1869, 1997.

[17] J. Glass, M. F. Ghalwash, M. Vukicevic, and Z. Obradovic, "Extending the modelling capacity of gaussian conditional random fields while learning faster." in *I*, 2016, pp. 1596–1602.

[18] H. N. I. Sample, "Healthcare cost utilization project (hcup). 2011," *vailable at: www. hcup-us. ahrq. gov/nisoverview. jsp. ccessed on May*, vol. 25, 2015.

[19] X. Zhou, J. Menche, .-L. Barabási, and . Sharma, "Human symptoms–disease network," *Nature communications*, vol. 5, p. 4212, 2014.

[20] J. Stojanovic, M. Jovanovic, D. Gligorijevic, and Z. Obradovic, "Semi-supervised learning for structured regression on partially observed attributed graphs," in *Proceedings of the 2015 SI M International Conference on Data Mining*. SI M, 2015, pp. 217–225.

[21] M. J. Menne, C. N. Williams Jr, and R. S. Vose, "The us historical climatology network monthly temperature data, version 2," *Bulletin of the merican Meteorological Society*, vol. 90, no. 7, pp. 993–1007, 2009.

[22] D. Gligorijevic, J. Stojanovic, and Z. Obradovic, "Uncertainty propagation in long-term structured regression on evolving networks." in *I*, 2016, pp. 1603–1609.

[23] C. Han, S. Zhang, M. Ghalwash, S. Vucetic, and Z. Obradovic, "Joint learning of representation and structure for sparse regression on graphs," in *Proceedings of the 2016 SI M International Conference on Data Mining*. SI M, 2016, pp. 846–854.

[24] S. Scepanovic, T. Vujicic, T. Matijevic, and P. Radunovic, "Game based mobile learning–application development and evaluation," in *Proc. of an 6th Conference on e-Learning*, 2015, pp. 142–147).

[25] J. Nielsen, "Usability 101: Introduction to usability," 2003.

[26] J. Rubin and D. Chisnell, *Handbook of usability testing: howto plan, design, and conduct effective tests*. John Wiley & Sons, 2008.

[27] J. Brooke *et al.*, "Sus-a quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.