Prediction of the disease controllability in a complex network using machine learning algorithms

Richa Tripathi[1, *], Amit Reza[1], and Dinesh Garg[1,2]

[1]Indian Institute of Technology Gandhinagar, Gujarat, India
[2]IBM India Research Laboratory - Bangalore, India
[*]richa.tripathi@iitgn.ac.in

May 5, 2020

Abstract

The application of machine learning (ML) techniques span a vast spectrum ranging from speech, face and character recognition, medical diagnosis, anomaly detection in data to the general classification, prediction and regression problems. In the present work, we solve the problem of predicting R_0 for disease spreading on complex networks using the regression-based state-of-art ML techniques. R_0 is a metric that determines whether the disease-free epidemic or an endemic state is asymptotically stable and hence indicates the controllability of the disease spread. We predict R_0, based on training the ML models with structural properties of complex networks, irrespective of the network type. The prediction is possible because: (a) The structure of complex networks plays an essential role in the spreading processes on networks (b) ~~The regression techniques~~ such as Support Vector Regression and Artificial Neural Network Model can be very efficiently used for prediction problems, even for non-linear data. We obtained good accuracy in the prediction of R_0 for the simulated networks as well as real-world networks using these techniques. Moreover, the ML model training is a one-time investment cost in terms of training time and memory, and the trained model can be used for predicting R_0 on unseen/new examples of networks.

1 Introduction

The problem of disease spreading has been studied using a system of Ordinary Differential Equations (ODE) [1] to predict the endemic disease state and devise effective control strategies. Earlier studies did not employ any spatial structure, and the dynamics generally depended on the population number and the probabilities of transitions from one disease state to others. However, the use of a spatial structure for determination of relative positions and interactions of individuals has taken a front stage recently. The complex network framework [3], where nodes represent the individuals, and the links govern the interactions between the nodes is thus very useful. Disease spreading on networks has been studied using various compartmental epidemiology models such as SI (Susceptible-Infected), SIR (Susceptible-Infected-Recovered), SIRS (Susceptible-Infected-Recovered-Susceptible), etc. [11]. The impact of disease in the population is measured using basic reproduction number, R_0 [14]. R_0 is the average number of individuals an infected person infects over its period of activity, such that if $R_0 < 1$ the disease will die out in the long run and if $R_0 > 1$ the disease-free stationary state is asymptotically unstable [24]. The fact that for a 100% effective vaccine, the fraction of individuals that need to be vaccinated is $1-\frac{1}{R_0}$ to prevent persistent disease spread, indicates that higher number of individuals need to be vaccinated if the factor R_0 is high for a disease. There have been several works [21,23] for determining the dynamical relationship between network and disease parameters for an epidemic spread occurring on networks. Machine learning (ML) models based on supervised and unsupervised learning algorithms have found important applications in the area of complex networks. For example, for optimal graph partitioning into community structure [16], for classification of diseased networks from the control networks using data from brain imaging [5], for classification of networks into various model networks [27], etc. Recently, a study [21] reported the relative relevance of network topological and disease parameters for disease spreading on complex networks, irrespective of the model network type. They found that the topology of population (or network) affects the disease spreading process, apart from the disease parameters. In essence, for the given initial conditions for the epidemic spread, the topological properties of networks govern the asymptotic disease state. Our work is based on this idea and focuses on exploring if the topological properties solely could predict R_0. The accurate determination/prediction of R_0 is of paramount importance to analyze the stability of the disease stage, concerning the infection outbreak. To this end, we train ML regression models, using large number of networks of various model network types. We used six structural properties of five different complex networks examples as input features and the corresponding R_0 evaluated after simulating the SIR dynamics on them, as output labels. While training, the model fits these inputs with the outputs and learns certain weights. Using the trained models (or the weights), we predict the R_0 value for test network example based on its own structural metrics. In particular, we trained three models: linear regression, support vector regression (non-linear regression) with different kernels and a neural network model. We optimized these models for the correct prediction of R_0 on test examples and evaluated two accuracy metrics: mean squared error and coefficient of determination [18] for each of them. We present the parameter list and their ranges for fine tuning of each of these models. Support Vector Regression (SVR) with radial basis function (RBF) kernel and the artificial neural network (ANN) model resulted in high accuracy of prediction over linear models, for both real and simulated networks. Moreover, the excellent overlap between the expected and predicted values of R_0 for these nonlinear ML models also point to the non-linear relationship between the model input and output variables. Hence, we show that simple ML techniques can be used to predict R_0 with high precision using the structural properties of the network. We also find that different network metrics have different relative powers of prediction. Based on this result, we can just use four most important measures out of the six, for the model training and prediction. However, ML model performances were marginally better with all the six features used.

2 Problem Statement

As explained earlier, the R_0 of disease spreading is an estimate of disease impact on the population and hence its controllability. For a dynamical process occurring on a network, the structure of the network plays a key role in determining the next stage of the process apart from inherent parameters of the process. Similarly for a disease spread on a network, the dynamics and hence the disease stages are a result of interplay between the network structural features and the epidemic model parameters such as the probability of infection, probability of recovery from infection, etc. For the R_0 calculation in the present work, we update all the disease related rate constants at each time step depending on the instantaneous values and the respective changes in the populations of the susceptible, infected and recovered individuals according to equations 2-5 in supplementary information (SI). Since in networks the interactions can only occur through node's neighbours, the instantaneous values and changes of populations of S, I and R over the whole network are indirectly determined by the local and global structure of the network. Hence for networks, the value of R_0, which is a dynamical descriptor of the disease and depends on the disease related rate constants is affected by the network structure. Given that R_0 is the average number of susceptible an infected individual infects over its period of being in infected state, different structural metrics of networks have specific effect on its value. For example, for an $Erdos - Rényi$ (ER) network, higher the clustering coefficient means higher the number of connections and hence higher the R_0. Similar trend is observed for network density, average degree and the maximum degree. On the other hand, higher the shortest path length means higher is the average shortest distance between two nodes and hence smaller the R_0 and vice versa. For a Small-World (SW) network, owing to the regularity of its structure, even the lower density of connections than an ER network shows similar potential for the disease spread. Hence, other topological features are also important for determining the R_0. In the same manner, the effect of topological parameters on the R_0 value can be intuitively understood for other model networks. In this work, we seek to predict the R_0 for any example network, given that we know its structural properties a priori. Given that R_0 is an important measure to understand the effect of disease on the population, it would serve as a warning for a presently unaffected population and device the vaccination strategies better for disease control. In this pursuit, we trained and optimized ML models with state of the art techniques and tested them on unseen artificial and real world networks. The results for SVR with RBF kernel and ANN show that R_0 was accurately predicted for these networks based on their known network properties. Hence, we have an estimate of disease controllability beforehand, without the need to simulate the SIR model on the test network.

3 Methodology

In this section we describe the procedure for generation of simulated data set. We also briefly describe the k-fold validation which a standard procedure used in ML model training and testing in the SI.

3.1 Generation of simulated data set

For simulation of the SIR model on networks, the parameters related to disease were fixed at: $k = 0.1$, $p_{ir} = 60\%$, $p_{id} = 30\%$, $p_{rs} = 10\%$. The starting population of individuals in each of the states was fixed at $S_o = 99.5\%$, $I_o = 0.5\%$ and $R_o = 0\%$. Each network had 1000 nodes and the network structure remains fixed throughout the simulation. The simulations were performed for 100-time steps and parameters a, b, c and e were determined using equations 2-5 in SI respectively. R_0 was calculated (using these parameters) and averaged over last 20 time steps, where the system reaches a stable regime (Figure. 4 in SI). The networks were obtained using the python library NetworkX [10], which returns a network as output, for the supplied input parameter(s) governing connectivity patterns. For obtaining n (say) number of networks of a model network type, n values of these parameters were chosen from a range. This range was carefully chosen, such that the network properties fall in more or less the same range for all the models. Around 500 networks of each model kind (exact number in the description below) each of size(N) 1000 were obtained. We use five model networks: *Erdos−Rényi* (ER) random networks [7], *Watts-Strogatz* small world (SW) networks [26], Scale Free (SF) networks [4], *Barabasi-Albert* [3] (BA) and Stochastic block model (SBM) networks [25] in our work. The parameters and their range of variation for each model network are described in the **Generation of model network examples** section in SI. The six network structural properties that were used as input features for training were Average Degree (avgdeg), Average Shortest Path Length (spl), Clustering Coefficient (cc), Network Density (den), Network Diameter (dia) and Maximum Degree (maxdeg). The definitions of these network metrics are presented in section **Complex Networks: Types and properties** of SI.

3.2 Training data-set and k-fold validation

For training the model, the *k-fold cross validation* routine of the sklearn library [18] was used. The k-fold cross-validation (CV) procedure avoids over-fitting by holding back a part of data from use in training the model; such that the model performance is evaluated and reported based on testing on the unseen data. The full data set is split into k folds, out of which $k-1$ folds are used for training the model and the remaining 1 fold is used for testing the model performance. The model performance score is recorded every time, and then the model is discarded. This procedure is carried out in a loop (k times) with a different fold held out for testing and $k-1$ folds used for training every time. In this way, each fold is used once for testing and $k-1$ times for training. Hence the model accuracy is averaged over all iterations of the procedure. In the present work all the ML model performances are evaluated using 10-fold cross validation technique. Also, please note that the data matrix was always permuted over rows before splitting it into testing and training parts, so that same model networks are not stacked together.

4 Model performance

The linear regression model resulted in a good fit when the networks from the same model were used for training and testing. The MSE and R^2 scores for the ER, SF, SW, BA and SBM networks as : (0.01, 0.99), (0.14, 0.87), (0.02, 0.99), (0.0, 0.99), (0.1, 0.99) respectively, indicating a good fit (corresponding plots shown in SI Figure 5). However, the linear regression lost the accuracy significantly when networks from all the models were used for training and testing (see Figure 5 with MSE and R^2 as (4.99, 0.69). This shows that linear-regression is not a reliable model for predicting R_0 irrespective of the network type. Also, the failure of linear regression confirms the absence of linear relation between the input and target variable and calls for testing of non-linear regression techniques.

4.1 Support Vector Regression

Owing to the failure of linear regression in accurately predicting R_0, we explored the performance of SVR with RBF kernel on the data set. The performances for the other two kernels (linear, polynomial) have also been reported (see Table 1 for results). The parameters for polynomial and RBF kernel functions are $\gamma = 0.1$, $degree(d) = 2$ and $\gamma = \frac{1}{no.\ of\ features}$ (refer Eqns:13-14 in SI) respectively. SVR with linear, polynomial

Table 1: Table of Model performance results

Model	**Description**	**(MSE, R^2)**
LR		$(4.99, 0.69)$
SVR	Linear Kernel	$(3.67, 0.73)$
	Polynomial kernel	$(11.30, 0.16)$
	RBF kernel	$(0.01, 1.00)$
ANN		$(0.093, 0.99)$

and RBF kernels show mean squared error and R^2 as $(3.67, 0.73)$, $(11.30, 0.16)$ and $(0.01, 1.00)$. Increasing the degree of the polynomial kernel to 3 improves the accuracy scores $(3.02, 0.81)$; increasing the degree beyond

3 resulted in an arbitrarily high error and requires much higher model training time than for degree 2. For the RBF kernel, the previously mentioned parameters were optimal concerning the accuracy and the required training time. Comparing the accuracy scores, we found that RBF kernel outperforms the other two kernels with a substantial margin and hence RBF can be used to predict R_0 with good precision. Please refer to Figure. 1 top panel and bottom panel (left figure) for the match of predicted output with the expected values, for all three kernels in SVR. The plots show predictions on only the first hundred data samples for better visualization.

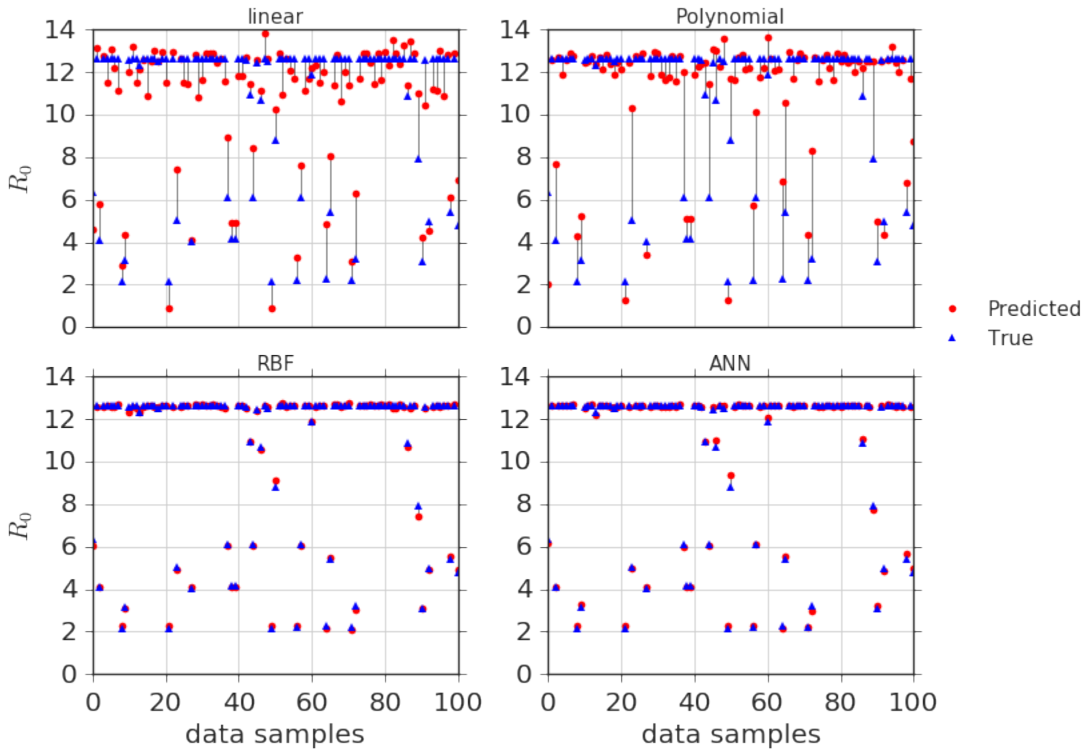


Figure 1: The figure shows the difference in predicted and true R_0 using vertical lines at data points for linear, polynomial, RBF kernel in SVR and ANN respectively. For better visualization of results only 100 randomly selected data points(true and corresponding predicted) are shown for all the models.

4.2 Neural Network Model

We also use a NN architecture (see Figure 2, left panel) that is optimized iteratively to gain maximum accuracy for regression. The NN model used here consists of three layers. The number of neurons in the input layer are conventionally fixed to be equal to the number of features of the data matrix. The output layer has one neuron, as the model performs regression to predict a number as an output (R_0). The hidden layer has 23 neurons that gather information from each neuron in the input layer and transmit it to the output layer. The number of neurons in the hidden layer were determined according to an empirical rule of thumb [12] that puts an upper limit on the total number of neurons without incurring the problem of over-fitting. The rule is,

$$N_h = \frac{N_s}{(\alpha\,(N_i + N_o))} \tag{1}$$

where N_i is the number of input neurons, N_o is the number of output neurons, N_s is the number of samples in the training data set, and α is an arbitrary scaling factor between 2 and 10. For $\alpha = 2$, we get the maximum number of neurons according to the above formula, and any number of neurons greater than this value will result in over-fitting. For our case, we chose $\alpha = 10$, to avoid over-fitting and reduce the number of free parameters or weights in the model. Putting the known values of N_i, N_o and N_s as 6, 1 and 2552, we obtained $N_h = 36$. The optimal value of N_h was then evaluated numerically by varying N_h over a range of numbers within 36. The accuracy metrics were evaluated for a different number of neurons in the hidden layer, and this exercise was repeated for ten trials on randomly permuted data set. The optimum number of neurons in hidden layer were found out to be 23, as can be seen from the variation of MSE and R^2 coefficient in Figure. 2, right panel.

We used Keras (deep learning library for Python) [6] for model construction and simulation. Other specifics of the model in terms of its hyper-parameters and parameters are as described in the Table 1 in SI. The weights

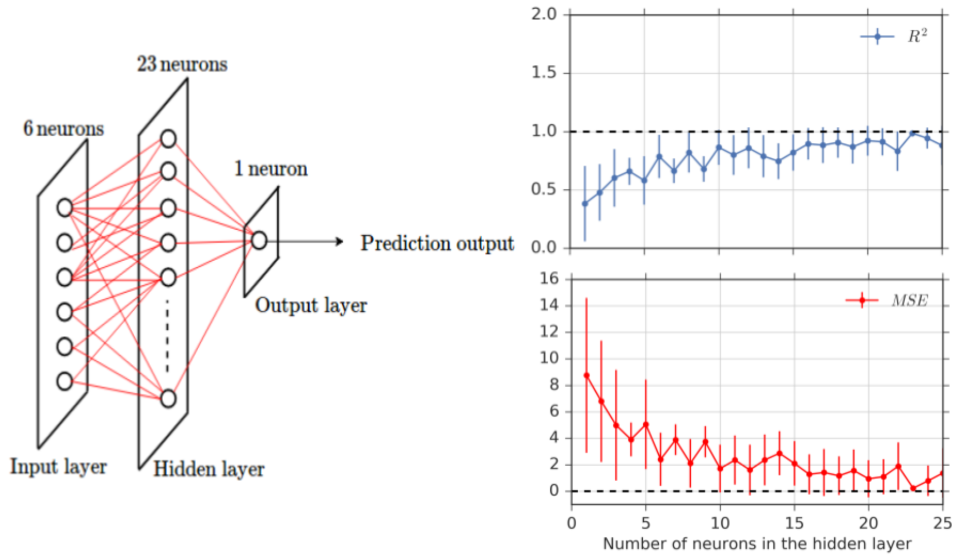
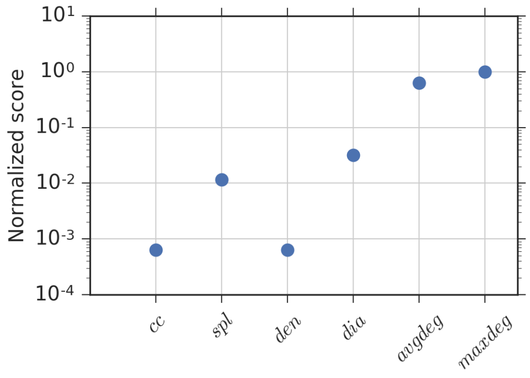


Figure 2: The left panel shows a schematic of the NN model used in the present work; with 3 layers and number of neurons in each layer specified. The figures in the right panel show R^2 coefficient (top) and MSE (bottom) for R_0 prediction, averaged over 10 realizations for different number of neurons in the hidden layer; it can be seen that for 23 number of neurons MSE touches the zero mark and R^2 touches the value one. This implies that using 23 number of neurons in the hidden layer gives the maximum accuracy.

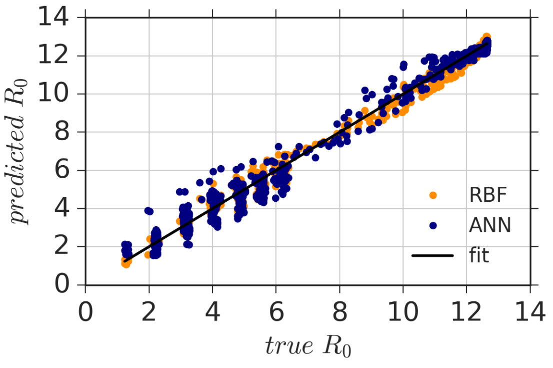
of edges in the NN network were chosen from a normal distribution using the kernel initializer function. The activation functions for neurons were governed by the rectified linear unit ("relu") i.e., the neuron activation was linearly related to the input. Adaptive Moment Estimation ("Adam") was used as optimizer, which is based on an adaptive learning scheme for updating of the network weights. This optimizer function updates the learning rates iteratively based on the moments of the gradient of the objective function. The objective function or the loss function was accuracy measured in terms of MSE. With epoch size ("Epochs") set at 50, the batch size of 5 and other parameters set as specified above in the NN model, the mean accuracy measured in terms of (MSE, R^2) for 10-fold cross validation was $(0.093, 0.99)$. We have shown the predicted and true R_0 for all the examples using SVR (with RBF kernel) and ANN model in Figure 3(b). We also trained all the ML model using only four (*avgdeg*, *maxdeg*, *dia*, *spl*) out of six features selected based on their contribution indices. These four features had highest values of the contribution indices (refer to subsection **Ranking of the features** in SI). We have shown the relative contribution indices for all the six features in the Figure 3(a). We observe that model performances are still fairly accurate in predicting the correct R_0. In the Table 2, the model performance metrics for SVR with RBF and ANN for training with six and four features respectively have been shown. We can infer from the results that there is a trade-off between accuracy of model predictions and number of features used for training the model. The accuracy of the predicted value is better with the all the six features in the data set than when only four feature vectors were considered. The precision of the prediction with top-four features is slightly reduced but it is in a bearable range (refer to the MSE and variance scores in table2). Therefore, one can remove *cc*, *den* from the feature set for the training without compromising much with the prediction accuracy.

Number of Features	ML technique	Accuracy Measures	
		MSE	R^2
Four	SVR(RBF)	0.11	0.99
	ANN	2.99	0.82
Six	SVR(RBF)	0.01	1.00
	ANN	0.013	0.998

Table 2: Table of ML model performance results on simulated networks.



(a)



(b)

Figure 3: (a) This figure shows relative importance of the network features based on their contribution index. The contribution indices are normalized between 0 and 1. (b) The figure shows the predicted R_0 vs true R_0 for ANN model and SVR model with RBF kernel for all the data points. The black line (fit) corresponds to ideal case where true R_0 is equal predicted R_0.

4.3 Performance on Real-world Networks

We tested the ML models for R_0 prediction for four real-world network datasets: *infect-dublin* [13], *infect-hyper* [13], *crime-moreno* and *email-univ* ([20], [8]). *infect-dublin* and *infect-hyper* are categorized as proximity networks based on face-to-face interactions between people, with the number of nodes (N) and number of edges (E) being (410, 2765) and (113, 2196) respectively. *crime-moreno* is categorized as interaction network with (N, E) = (829, 1474).*email-univ* is email network with (N, E) = (1133, 5451). Please note that we chose networks with single giant component only.
The accuracy metrics corresponding to the ML models on real-world networks is tabulated in Table 3. It is observed that the ML model performs very accurately for these networks as well, as for artificial networks. Especially, for *infect-dublin* and *crime-moreno* networks, both SVR and ANN models predict R_0 that almost matches the true R_0 value. Furthermore, these real-world test networks serve as unseen test examples for the ML models, and accurate prediction of R_0 authenticates the ML models even more.

Table 3: Table of Model performance results on real world networks

Dataset	True R_o	Pred. R_o (SVR)	Pred. R_o (ANN)
infect-dublin	5.63	5.97	5.13
infect-hyper	10.02	8.09	10.27
crime-moreno	1.95	1.75	2.01
email-univ	4.22	4.19	3.67

5 Conclusion and Future Prospects

The present work explored the applicability of ML regression techniques to predict basic reproduction number, R_0, a factor indicative of the effect of disease or its controllability, on complex networks. R_0, in general, depends on many factors: the duration of disease persistence in the population, the vulnerability of an individual to an infection, the number of infected neighbours to a susceptible individual, etc. On the other hand, if we have a population where all these above parameters are fixed to a reasonable value, how the social strata(complex network in our case) on which the disease spreads affect the disease spreading is still a question. To explore this, we examined whether R_0 can be predicted based on global properties of the network in hand, irrespective of the model network type it belongs to? A large number of networks were generated, and dynamics of the disease spreading were simulated on these networks, and the corresponding R_0 was recorded along with the network properties. Using the recorded data, three ML regression models were trained to predict R_0 values on the test data. These models were tuned based on their parameters to obtain good prediction. The results using RBF kernel in SVR and ANN models showed high accuracy of R_0 prediction, suggesting that there exists a significant correlation between the network properties and disease controllability. The generalizability of the trained models is convincing because the testing was always performed on unseen data using the k-fold validation technique. One of the improvements to the present work could be to train the models using a larger number networks of different sizes, such that a higher range of network properties such as shortest path length, clustering coefficient, etc. is spanned. These models will then yield correct prediction for any given test network. However, as one can see, this is just a scalability issue. Moreover, good predictions of these models on real-world networks is an exciting result. Our work reports two significant findings (a) The disease controllability on the network can be predicted using global network properties. (b) The standard ML techniques can be applied to processes on complex network. In our case it is predicting disease controllability on a network. The tunability of ML models offers immense power to forecast or predict processes on complex network systems. The computational cost for some of the features is high (especially for the clustering coefficient (cc) and the shortest path length (spl)). Hence, the time complexity for obtaining the features for the training data set will be high. This is one of the constraints of our approach for the ML model training. But, for predicting the value of R_0 for any arbitrary test network based on known network features, the prediction time is almost negligible. This implies that we can predict the value of R_0 at the very first stage of getting the test network (without waiting until the epidemic outbreak has completed or reached a stable state). Of course, underlying assumption is that we should know the value of these features beforehand at the time of testing. The prospects of the work may include using deep learning approaches for unsupervised learning of features. As we know that the numerical calculation of network properties for the training as well as testing the model is a time-consuming step, it would be great if the network itself could be made to train the model. Another prospect is to explore if the network adjacency matrix can be used to train a deep learning CNN architecture. Also, if network embedding algorithms can be used to learn the features that are instrumental in the disease spreading, it would be a significant leap forward from this work.

6 Supplementary Material

6.1 Complex Networks: Types and properties

The complexity of interactions possible in complex networks and the availability of various model networks in network science theory, which dictate their topology, offers us a plethora of settings for modeling of real-world interactions. For example, the Erdős-Rényi random networks (ER) are the ones in which the connections are randomly assigned between the nodes with some probability. A small-world (SW) network can be visualized as a distortion to a regularly connected circular lattice, with a fixed number of nearest neighbor connections. The distortion is caused by rewiring some of the connections to far off nodes that are not the nearest neighbors, accounting for long-range connections in such networks. A scale-free (SF) network obeys a power law in its degree distribution i.e fraction of nodes ($p(k)$) having k number of connections to other nodes is given by $p(k) \propto k^{-\alpha}$, where $2 \leq \alpha \leq 3$. The Barabási-Albert (BA) network model is an algorithm for generating a scale-free network based on a mechanism known as preferential attachment, wherein a new node is added to the network such that the existing nodes with an already greater number of connections to other nodes gather new nodes with greater probability and vice-versa. The Stochastic Block Model (SBM) network is another generative model for random graphs with community structure, where the inter and intra-community edge densities are governed by fixed numbers. The network properties: average degree, average shortest path length, clustering coefficient, network density, network diameter, and maximum degree determine the large scale structure and have been used in the present study as an input to train ML regression algorithms for prediction of R_0. They are defined as follows.

- **Average Degree (avgdeg)**: It is the average of the number of links that each node in the network has to the other nodes (degree).
- **Average Shortest Path Length (spl)**: The shortest path between any two nodes in the network is the shortest route or the one that involves the least number of edges in travelling between these two nodes. The average of all the shortest paths between all the node pairs is the average shortest path length.
- **Clustering Coefficient (cc)**: Mathematically, it is the ratio of total number of closed triplets to the number of all open or closed triplets of nodes present in the network.
- **Density (den)**: Density of the network is the ratio of the number of edges present in the network to the number of possible edges in the same network.
- **Diameter (dia)**: Diameter is the measure of the linear size of the network. It is the longest of all the shortest paths between all node pairs in the network.
- **Maximum Degree (maxdeg)**: It is the maximum of all the degrees of the nodes in the network.

6.2 The SIR Model

The compartmental model such as the basic SIR model [2] assumes that an individual in the population falls in one of the compartments (for example Susceptible population, Infected population, etc.) at a particular time. The transitions from one state to another are governed by the constants β, γ and ζ which are contact rates between susceptible and infected population and transition rate of the infected population to recovery, and transition rate of the recovered population to being susceptible, respectively. This model is highly predictive of dynamics of a class of airborne diseases, for example seasonal influenza, where an individual's immunity may diminish with time. There are advanced versions of the basic SIR model that involve a death rate, a birth rate, and the effect of vaccination to incorporate more realism concerning the actual processes in the living world into the model [2]. In this work, the SIR model used for simulations incorporates deaths of the infected individual due to the disease apart from trivial state transitions from S to I, I to R and R to S. The underlying assumption is that populations in different states are homogeneously distributed over the network, where a node at each time step represents an individual. The possible state transitions in the model are :

- A susceptible individual becomes infected with probability $p_{\mathrm{si}} = 1 - e^{-ki}$, where i is the number of infected neighbours at one edge distance and k is some disease parameter.
- An infected individual recovers from disease with probability p_{ir}.
- An infected individual may die due to disease with probability p_{id}.
- A recovered individual may become susceptible to disease with probability p_{rs}
- An individual in Susceptible, Infected or Recovered state may continue being in the same state in the next time step.

The ODEs describing the mean field model are as follows,

$$\begin{aligned}\frac{dS(t)}{dt} &= -aS(t)I(t) + cI(t) + eR(t)\\ \frac{dI(t)}{dt} &= aS(t)I(t) - (b+c)I(t)\\ \frac{dR(t)}{dt} &= bI(t) - eR(t)\end{aligned} \tag{2}$$

where S, I and R are the number of susceptible, infected and recovered individuals in the population, respectively. a is the infection rate constant; b is the recovering rate constant; c is the death rate constant related to the disease; e is the rate constant governing recovered individuals getting susceptible. The basic assumption of this model is $S(t) + I(t) + R(t) = N$, i.e. the total number of individuals remains constant. To maintain a constant total population the infected individuals that die due to disease appear as S-individuals in the population. Hence, the model in our paper is a SIR model (with vital dynamics) that incorporates death of infected individuals. The connection between the mean field dynamics of ODE and the dynamics resulting from nearest neighbor interactions in networks is established by following relations (please refer [21, 22] for details),

$$a \simeq \frac{\Delta I(t)_{S\to I}}{S(t)I(t)\Delta t} \tag{3}$$

$$b \simeq \frac{\Delta R(t)_{I\to R}}{I(t)\Delta t} \simeq p_{\mathrm{ir}} \tag{4}$$

$$c \simeq \big(1 - \frac{\Delta R(t)_{I\to R}}{I(t)\Delta t}\big)\frac{\Delta S(t)_{I\to S}}{I(t)\Delta t} \simeq (1 - p_{\mathrm{ir}})p_{\mathrm{id}} \tag{5}$$

$$e \simeq \frac{\Delta S(t)_{R\to S}}{R(t)\Delta t} \simeq p_{\mathrm{rs}} \tag{6}$$

The basic reproduction number is defined as $R_0 \equiv \frac{aN}{b+c}$. From the steady-state analysis of the Eq.2, it is clear that stability of the disease spreading can be classified based on different limits of R_0. The stationary state is asymptotically stable if $R_0 < 1$ and unstable if $R_0 > 1$; and the endemic stationary state is unstable if $R_0 < 1$ and asymptotically stable if $R_0 > 1$. The typical SIR dynamics on four of the model networks are shown in Figure. 4.

6.3 Generation of model network examples

- *Erdos − Rényi random network*: 525 network examples were generated using the erdos_renyi_graph(n, p) generator of NetworkX module that requires user to input probability of connection (p) and the number of nodes (n). Hence, 525 values of p were selected from the range (0.0072, 0.5) and n was fixed at 1000.
- *Watts-Strogatz Small World networks*: 520 network examples were generated using the watts_strogatz_graph(n, k, p) generator of NetworkX module which requires user to enter number of nodes(n), rewiring probability(p) and number of nearest neighbors(k) as parameters. n was fixed at 1000, value of p was varied between (0.1, 0.5) in the step size of 0.01 and m were varied in a range (2, 15) in steps of 1.
- *Scale Free networks*: Using powerlaw_cluster_graph(n, m, p) generator of NetworkX that inputs parameters n, m and p a total of *548* network examples were generated. The number of nodes(n) is 1000, the number of random connections every new node attains (m) is varied in the range $(2, 550)$ in steps of 1,and the probability of adding a triangle after adding a random edge(p) is fixed at 0.2.
- *Barabási − Albert networks*: Using barabasi_albert_graph(n, m) generator of NetworkX module, a total of 548 network examples were generated. These were generated by varying the number of connections between every new node to existing node (m) in the range $(2, 550)$ in steps of 1 and number nodes(n) was fixed at 1000.
- *Stochastic block model*: The SBM networks had 1000 nodes and two communities each; different examples were generated using different probabilities of within and across community connections under the constraint that the adjacency matrix is symmetric. SBM module of python was used to generate adjacency matrices and NetworkX module was used to generate networks from these matrices.

We ruled out the network examples which had more that one component because the infection cannot flow between the disconnected components. Moreover, the global network properties such as the shortest path length are meaningless and cannot be computed for disconnected networks. All the six network structural properties (described before) were calculated for all these networks along with the corresponding R_0 value (obtained after simulating the SIR dynamics on the network). The final size of the feature matrix formed by concatenating all the networks of five model types was 2552×6. The R_0 for each of the model networks ranges from 2 to 12 (approximately), except for the *Watts-Strogatz* model where it goes upto a maximum of 6.5 (approximately). The distribution of the R_0 values can be seen from Figure. 5 for each of the network types, where we have shown the linear regression fit to true R_0 values.

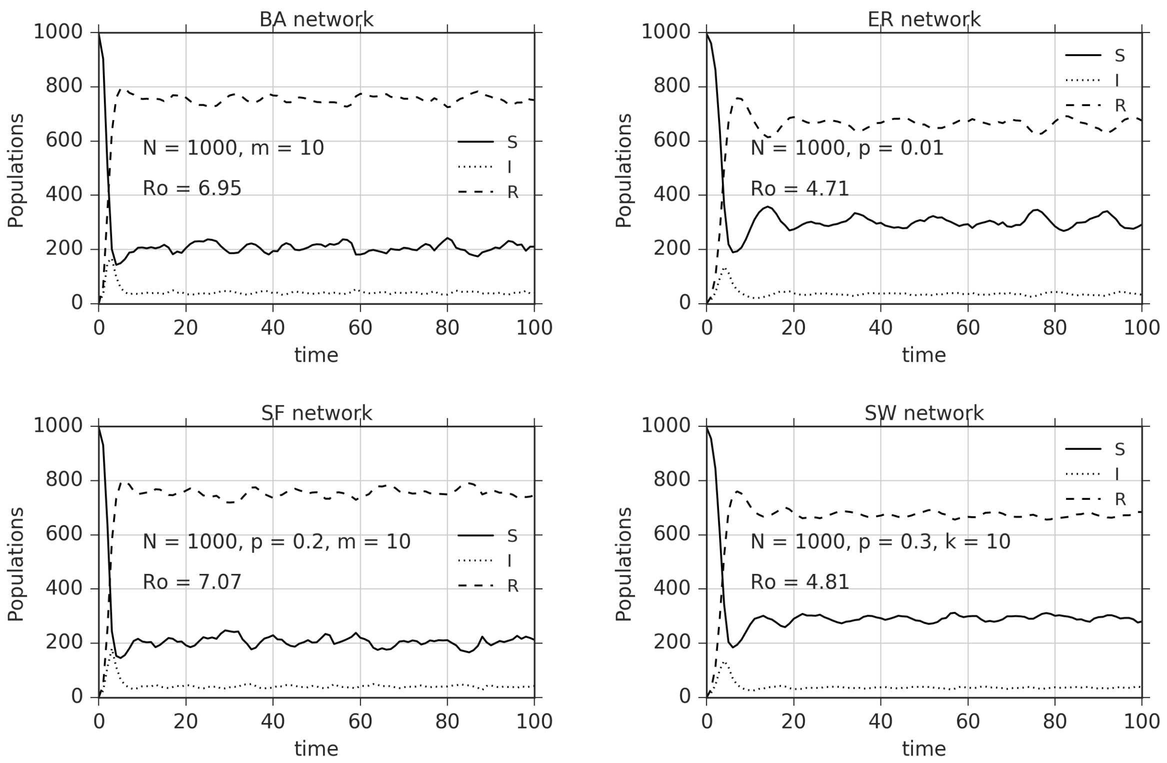


Figure 4: SIR dynamics on four model networks. The dynamics show that the system reaches permanent regime by 100 time steps.

6.4 Ranking of the features

All the features may have different predictive power, and hence their contribution in the prediction of R_0 is different. We obtained their individual contributions to understand the relative importance of these features for predicting R_0. The relative importance are useful to decide whether all the features are absolutely necessary to train the ML model or some features can be ignored.

The relative importance of each feature vector is decided based on the value of its contribution index. The contribution index is calculated based on principal feature analysis (PFA) [9, 15], which indeed performs the principal component analysis (PCA) of the data matrix D. For a data matrix D of size $n \times N$, where N is the number of features and n is the total number of samples, the scheme for calculating the contribution index is as follows:

- Compute the co-variance matrix $\Sigma_{n\times n}$ of the data matrix D.
- Obtain the principal components i.e. the eigenvectors $\vec{X_i} : i = 1, 2, .. \cdots n$ and corresponding eigenvalues Λ_i of Σ.
- Choose top-p principal components based on their eigenvalues.
- Project each of the column vectors of the data matrix along these p principle eigen directions i.e for jth column vector $(\vec{d_i})$, compute $c_j = \sum_{i=1}^{p} \vec{d_j} \vec{X_i}$.

The quantities $c_j : j = 1, 2, \cdots N$ are the contribution indices of each of the feature vectors.

6.5 Regression

Regression is a mathematical model for finding the relation between the dependent and independent variables of a system and is used for forecasting and prediction problems. Essentially, it solves a system of equations written in the matrix notation as $\mathbf{A}\,\overrightarrow{x} = \overrightarrow{y}$, where $\mathbf{A}$ is a data or feature matrix and x and y are vectors known as the weight and target vectors respectively.

The elements of $\overrightarrow{x}$ or weights are determined using $\mathbf{A}$ and $\overrightarrow{y}$, the independent and dependent quantities respectively in the system.

$$\mathbf{A}\,\overrightarrow{x} = \overrightarrow{y} \Rightarrow \overrightarrow{x} = (\mathbf{A}^T\ \mathbf{A})^{-1}\,\mathbf{A}^T\,\overrightarrow{y} \tag{7}$$

The quantity $(\mathbf{A}^T\,\mathbf{A})^{-1}\,\mathbf{A}^T$ is the Moore-Penrose (pseudo-inverse) [17], [19] of matrix $\mathbf{A}$. For machine learning applications, the weights are learned from the training data ($\mathbf{A}_{train}$, $\overrightarrow{y}_{train}$) and then used to predict $\mathrm{y}_{\mathrm{pred}}$ vector for the test data ($\mathrm{A}_{\mathrm{test}}$), obtained by multiplying learned $\overrightarrow{x}$ with $\mathbf{A}_{test}$ as,

$$\overrightarrow{y}_{\mathrm{pred}} = \mathbf{A}_{\mathrm{test}}\,\overrightarrow{x} \tag{8}$$

The predicted and expected values of target quantities $\mathrm{y}_{\mathrm{pred}}$ and y_{test} respectively are then compared to infer the accuracy of the ML model based on some standard metrics like mean squared error (MSE), the coefficient of determination (R^2 coefficient), etc.

6.6 Machine learning techniques for regression

In this section, we present an overview of linear and non-linear regression ML algorithms used in this paper for predicting the value of R_0 based on the network-features matrix.

6.6.1 Linear Regression

Linear regression (LR) [18] is a statistical method to analyze the linear relation between the observed responses (independent variable) and the target value (dependent variables) of a data set. Mathematically, the target value is defined as a linear combination of the observed responses i.e.

$$\hat{y} = w_0 + w_1\,x_1 + w_2\,x_2 + \cdots + w_p\,x_p \tag{9}$$

where $w = \{w_1, w_2, \cdots, w_p\}$ represents the weight vector. LR fits a linear model using optimum weights to the residual sum of squares between the observed responses and the responses predicted in the data set. Mathematically, it solves an optimization problem of the form:

6.6.2 Support Vector Regression

The kernel trick allows the model to fit the maximum-margin hyperplane in the transformed feature space optimally. The transformed space may be high dimensional. Some of the well-defined kernel functions are:

1. Linear kernel:
$$k(x, x') = \langle x, x' \rangle \tag{10}$$
2. Polynomial kernel:
$$k(x, x') = (\gamma\langle x, x' \rangle + r)^d \tag{11}$$
3. Radial basis function kernel (RBF):
$$k(x, x') = \exp(-\gamma\|x - x'\|^2) \tag{12}$$

where in polynomial kernel, r and d are a free parameter depicting trade off between the influence of higher-order and lower-order terms in the polynomial and degree of the kernel, respectively. γ is also a parameter in RBF kernel.

References

[1] Roy M Anderson and Robert M May. *Infectious diseases of humans: dynamics and control.* Oxford university press, 1992.

[2] Norman TJ Bailey et al. *The mathematical theory of infectious diseases and its applications.* Charles Griffin & Company Ltd, 5a Crendon Street, High Wycombe, Bucks HP13 6LE., 1975.

[3] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[4] B Bollobás, C Borgs, J Chayes, and O Riordan. Proceedings of the fourteenth annual acm-siam symposium on discrete algorithms. 2003.

[5] Sandeep Chaplot, LM Patnaik, and NR Jagannathan. Classification of magnetic resonance brain images using wavelets as input to support vector machine and neural network. *Biomedical signal processing and control*, 1(1):86–92, 2006.

[6] François Chollet et al. Keras. https://keras.io, 2015.

[7] P ERDdS and A R&WI. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[8] Roger Guimera, Leon Danon, Albert Diaz-Guilera, Francesc Giralt, and Alex Arenas. Self-similar community structure in a network of human interactions. *Physical review E*, 68(6):065103, 2003.

Table 4: Table of Neural Network parameters and hyperparameters

Parameters	**Value/Specification**
Dimension of Data Matrix	2552×6
Model	"sequential"
Number of layers	3
Neurons in input layer	6
Neurons in hidden layer	23
Neurons in output layer	1
Activation	Rectified Linear Unit ("relu"), for first two layers
Optimizer	Adaptive Moment Estimation ("Adam")
Loss function	"Accuracy"
Kernel initializer	"Normal", for each layer
Epochs	50
Batch Size	5
Metric 1	Mean Squared Error
Metric 2	R^2 Coefficient

[9] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[10] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[11] Herbert W Hethcote. The mathematics of infectious diseases. *SIAM review*, 42(4):599–653, 2000.

[12] hobs (https://stats.stackexchange.com/users/15974/hobs). How to choose the number of hidden layers and nodes in a feedforward neural network? Cross Validated. URL:https://stats.stackexchange.com/q/136542 (version: 2017-04-13).

[13] Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. What's in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology*, 271(1):166–180, 2011.

[14] Alfred J Lotka. Elements of mathematical biology. 1956.

[15] Yijuan Lu, Ira Cohen, Xiang Sean Zhou, and Qi Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 301–304. ACM, 2007.

[16] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[17] E Moors. On the reciprocal of the general algebraic matrix, abstract. *Bull. Amer. Math. Soc.*, 26:394–395, 1920.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] Roger Penrose. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press, 1955.

[20] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[21] Pedro HT Schimit and Fábio Henrique Pereira. Disease spreading in complex networks: A numerical study with principal component analysis. *Expert Systems with Applications*, 97:41–50, 2018.

[22] PHT Schimit and Luiz Henrique Alves Monteiro. On the basic reproduction number and the topological properties of the contact network: An epidemiological study in mainly locally connected cellular automata. *Ecological Modelling*, 220(7):1034–1042, 2009.

[23] Mark DF Shirley and Steve P Rushton. The impacts of network topology on disease spread. *Ecological Complexity*, 2(3):287–299, 2005.

[24] Andrew D Stewart, John M Logsdon, and Steven E Kelley. An empirical study of the evolution of virulence under both horizontal and vertical transmission. *Evolution*, 59(4):730–739, 2005.

[25] Tcoyze. Project title, March 2016.

[26] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440, 1998.

[27] Ruyue Xin, Jiang Zhang, and Yitong Shao. Complex network classification with convolutional neural network. *arXiv preprint arXiv:1802.00539*, 2018.

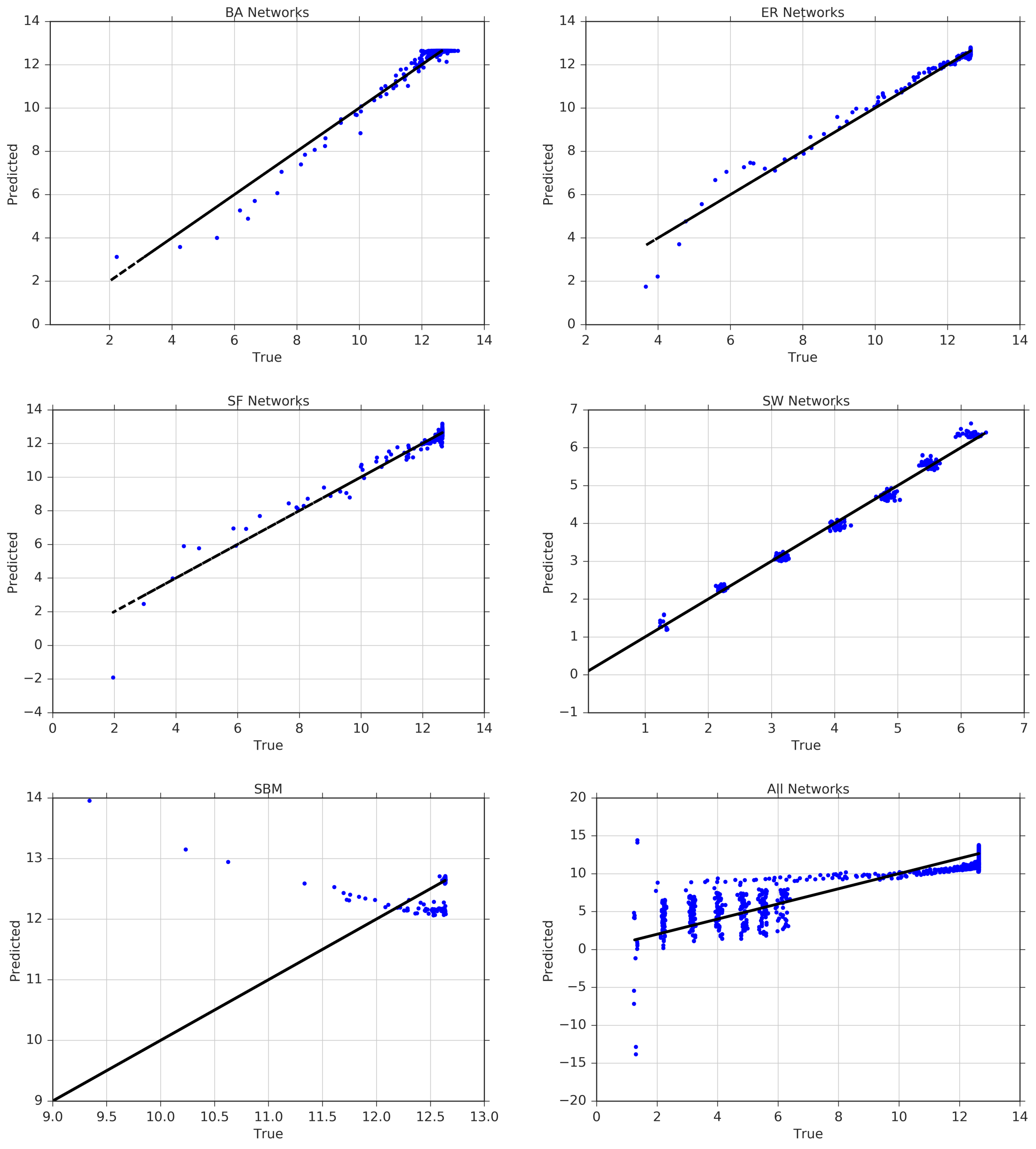


Figure 5: **Results obtained using linear regression**(a) ER Networks (b) SW Networks (c) SF Networks (d) BA Networks (e) SBM networks (f) all the networks. The blue dots represent the true/test labels (R_0) and the black dotted line in each curve (from (a) to (e)) shows the linear fitting to the test data, obtained from training the LR model using the train data with only the corresponding network examples. Figure (f) depicts that a linear model cannot fit true R_0 values, when the model is trained and tested using all the five networks examples.