Journal of Engineering Design

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/cjen20

Reasoning support for predicting requirement change volatility using complex network metrics

Phyo Htet Hein, Elisabeth Kames, Cheng Chen & Beshoy Morkos

To cite this article: Phyo Htet Hein, Elisabeth Kames, Cheng Chen & Beshoy Morkos (2022) Reasoning support for predicting requirement change volatility using complex network metrics, Journal of Engineering Design, 33:11, 811-837, DOI: 10.1080/09544828.2022.2154051

To link to this article: https://doi.org/10.1080/09544828.2022.2154051

Published online: 06 Dec 2022.

JOURNAL OF ENGINEERING DESIGN
2022, VOL. 33, NO. 11, 811–837
https://doi.org/10.1080/09544828.2022.2154051

Taylor & Francis
Taylor & Francis Group

Reasoning support for predicting requirement change volatility using complex network metrics

Phyo Htet Hein[a], Elisabeth Kames[b], Cheng Chen[c] and Beshoy Morkos[c]

[a]Department of Mechanical and Civil Engineering, Florida Institute of Technology, Melbourne, FL, USA; [b]Department of Mechanical Engineering, Florida Polytechnic University, Lakeland, FL, USA; [c]College of Engineering, University of Georgia, Athens, GA, USA

ABSTRACT

Due to inevitable design iterations, requirements are frequently changed and revised. If not effectively managed, undesired propagating changes may result in monetary and time losses, leading to project failure. Current requirement management tools lack formal reasoning for characterising change and its propagation. Based on prior research that suggests that requirement networks can be utilised to study change propagation and that requirement change volatility (RCV) can be measured through four classes: namely, multiplier, absorber, transmitter, and robust, which are defined based on how requirements behave to the initial change. This research investigates if complex network metrics can be used to predict RCV using computational methods. The RCV class metrics of each requirement are determined using industrial data and requirement relationships obtained from networks of a previously developed Refined Automated Requirement Change Propagation Prediction (R-ARCPP) tool. Complex network metrics are also computed for each requirement in the network. Regression analysis models, specifically ordinal output cumulative logit regression models employing information criterion based genetic algorithm model selection, and artificial neural network models, specifically backpropagation artificial neural networks, are employed to predict RCV using complex network metrics. Best performing models are discussed along with the limitations, conclusions, and future direction of this research.

ARTICLE HISTORY
Received 10 May 2021
Accepted 29 November 2022

KEYWORDS
Complex system; engineering change; requirement change propagation; requirement volatility; complex network

1. Requirement change and propagation

Requirements define stakeholders' needs, functions, and performance required of a finished product (Pedersen, Christensen, and Howard 2016) and desired effects of a system (Bock and Galey 2019; Ledoux, Teissandier, and Sebastian 2016). Requirements are managed by many engineers (Wang et al. 2021) may evolve over the course of a design project due to internal factors such as design environment and business needs or external factors such as stakeholders and the technological landscape. Considering that the real-world requirements network is complex, the behaviour of each engineering requirement can be

CONTACT Beshoy Morkos bmorkos@uga.edu College of Engineering, University of Georgia, 2040J iSTEM 1, 302 East Campus Road, Athens, GA 30602

deconstructed into four classes of volatility. Adding new requirements or evolving existing requirements can trigger such changes. To capture such change dependence more accurately, several studies have empirically investigated the co-evolution between design problems and solution spaces in different design projects (Duran-Novoa et al. 2018). However, this study adopts a computational approach to determine various types of requirements RCV classes. Additionally, it's important to note that this volatility approach is represented as an individual change scenarios (where a single requirement changes). However, change can occur due to multiple requirement changes that change over multiple time instances – a phenomenon we coin cumulative change propagation and have studied in previous research (Morkos, Shankar, and Summers 2012; Morkos, Mathieson, and Summers 2014). Singular requirement changes can be manageable, however, requirement changes rarely happen alone. A change to one requirement may cause other requirements to change due to functional or non-functional dependencies (Hein, Voris, and Morkos 2018; Hein et al. 2015; Wen, Tuffley, and Dromey 2014; Li et al. 2021). This phenomenon is known as 'requirement change propagation'. Requirement changes can propagate uncontrollably, resulting in unforeseeable and undesirable effects. Therefore, the management of requirement changes is essential to ensure that the effects are not detrimental, as studies have shown that engineering changes (ECs) determine 70–80% of product cost (Fei 2011) and a single EC is estimated to take multiple months to implement. To address this deficiency, this research seeks to develop formal reasoning to assess unanticipated requirement change propagations (Shafqat, Oehmen, and Welo 2022; Wichmann, Gericke, and Eisenbart 2021) caused by the initial change, allowing for the mitigation of detrimental effects such as added cost and delayed time.

Advancements in technology increase the difficulty of the requirements, therefore, making it more challenging to study requirement change propagation. This creates the need to develop computational models to assist in predicting requirement change propagation (Li et al. 2022b; Ullah et al. 2017). We specifically consider the requirements' volatilities in assessing change propagation (Hein et al. 2021). Our initial study explored different complex network metrics for predicting requirement change volatility (RCV) using multilabel learning (MLL). In this new study, we have extended our previous work by using OOLR and BPANNs to predict four different RCV classes. We measure this volatile behaviour through four volatility classes: (1) multiplier, (2) absorber, (3) transmitter, and (4) robust, which characterise the requirement's ability to multiply, absorb, transmit the initial change, or be robust to it. A multiplier is a requirement that is altered by an initial change and propagates engineering changes to other related requirements. An absorber, however, is a requirement that does not propagate the change to other related requirements. A transmitter is a requirement that is not changed by an initial change but propagates the change to other related requirements, whereas robust requirement does not propagate the change.

RCV classes are observed and defined according to the topology of the requirement network (Hein et al. 2021). To predict each RCV class, this requirement network is treated as a classification problem. Based on every initial change scenario and assumptions, each requirement has the potential to alter a different role, resulting in further actions. Consequently, the predicted result of RCV classes is not simply governed by the topology structure of each network, but is also affected by every initial change in design. Therefore, numerous machine learning models were used in this study to capture different information for every change instance.

1.1. State of the art

In a change context, change can be regarded as an input to the system and change propagation can be considered as the response of the system entities (component-, subsystem-, or system-level) due to interactions along their relationships. To study these characteristics, a Design Structure Matrix (DSM) is utilised given their use in many engineering design (Mollajan and Iranmanesh 2021), system modelling (Stirgwolt, Mazzuchi, and Sarkani 2022; Wang et al. 2022; Paparistodimou et al. 2020), and change management approaches (Mollajan and Iranmanesh 2021; Hein, Morkos, and Sen 2017).

The first group of research studies explored do not use a complex network approach explicitly and will be referred to in this paper as non-network approaches. Some examples of non-network approaches include Change Prediction Method (CPM) models in (Clarkson, Simons, and Eckert 2004; Hamraz et al. 2013), the Requirement Change Propagation (ReChaP) by Ibrahim, Kadir, and Deris (2011), and the requirement relationship model by Chen, Cheng, and Yin (2010). However, non-network studies have limitations including, but not limited to: unsuitability for early design use before physical component or system architecture information is available (Stirgwolt, Mazzuchi, and Sarkani 2022), inability to generate objective dependency relationships free of experts' subjectivity, inapplicability to heterogenous electromechanical systems, and unsuitability for all requirement types.

The second group of research studies explored utilise a network approach explicitly and will be referred to in this paper as network approaches. Some seminal works employed complex networks to model real world systems and discovered insights regarding technical and organisational dynamics (Braha and Bar-Yam 2004, 2007). Build upon Eckert's idea (Eckert, Clarkson, and Zanker 2004), Suh and deWeck (Suh, de Weck, and Chang 2007) developed an index namely the Change Propagation Index (CPI), which is calculated based on the number of ingoing and outgoing edges of the component node to classify the component's change propagation behaviour as a multiplier, carrier, absorber, or constant. Additional studies have observed Change Propagation Frequency (CPF), Change Acceptance Index (CAI), Change Reflection Index (CRI) (Giffin et al. 2009), and network centrality metrics (degree, closeness, betweenness, PageRank, and change impact rank – derivative of PageRank) (Wang, Huang, and Qu 2014). These methods have similar limitations as the non-network approaches, and none explore a comprehensive list of metrics, although they utilise network features or metrics in their own contexts. The study by Menon (2015) explored relevant complex network metrics in the existing literature for use in predicting the change state of requirements due to change propagation. The literature, however, does not address the volatility of requirements in terms of change as defined in this paper.

As engineering changes can be categorised into functional, behavioural, and structural dependencies, modelling requirements can provide valuable insights (Koh 2017). In this study, however, the behaviour of requirements is further generalised into four classes of volatility. Engineering changes are more likely to become multipliers for system-level requirements. Previous studies have modelled such phenomena with first- and higher-order changes. As a natural extension, this study explores the behaviour of requirements in four classes of volatility. Note that requirements do not exist in the same level of abstraction. Change can be initiated within any requirement – including system level requirements. For instance, consider a case where a vehicle requirement is changed from 'less an 2000kg' to

'less than 1900kg'. This type of requirement change occurring at the system level requirement will have significant propagation – to anything that is related to the weight of the vehicle – due to its high connectivity. Fortunately, most of the requirement changes occur at the end nodes (specific components) of this study, which is conducive to further analysis.

The only tool for change propagation in the requirement domain, based on our literature review, is the Automated Requirement Change Propagation Prediction (ARCPP) tool (Morkos 2012). ARCPP uses syntactical natural language data, part of speech (POS) elements parsed from requirement sentences, to identify syntactical relationships between requirements. Using these relationships, which can be represented by either a DSM or a network, the change propagation prediction method identifies requirements where change may be potentially propagated. The propagation analysis involves prioritising these requirements based on their relative distances and validating the results against data gathered from industrial case studies. The refined version of the tool, namely the R-ARCPP tool tags requirements with only nouns POS elements representing the physical domain, identified as the most conducive domain for change propagation (Hein, Voris, and Morkos 2018). The tool extracts nouns based on the uniqueness of tagged word sets of all requirements to capture all possible relationships (Hein, Voris, and Morkos 2018). As the requirement networks generated in these studies are proven to characterise requirement change propagation, they will be employed in this paper to study RCV. More detailed information about the tool is not discussed here for brevity and can be consulted in corresponding references.

A recent study demonstrated that seven unrelated problem-solving networks with similar local motif structures display fast convergence rates (Braha 2020). A natural extension of this study can be the examination of the convergence of requirement networks. Modelling the diffusion of information through local motif structures is difficult for a large-scale requirement network. A single engineering requirement may contain four different volatility classes; therefore, every change path can result in a different way to propagate information based on domain knowledge. There may not be enough resources available at the organisational level to implement engineering changes. Convergence rates in design can be significantly affected by the propagation of design information through exploitation (i.e. local subgraph structure) and exploration. A future study should entail the use of representational learning, such as Node2vec (Grover and Leskovec 2016), to study different requirements networks.

Current approaches for modelling requirements volatility rely on PLM software processes such as change management systems or dependency relationship simulations (Dong et al. 2022; Padala and Maheswari 2022; Gräßler, Oleff, and Preuß 2022; Yin et al. 2022). The setup of these methods can be labour-intensive operations and the methods themselves only quantify the effects of change propagation. The use of our proposed volatility model could mitigate these issues by estimating the likelihood of requirement change propagation for more accurate simulation results. The significance of this approach is its focus on capturing a single change instance in complex designs and reducing uncertainty – which has long plagued requirements (Zhong et al. 2022).

2. Proposed research

The objective of this research is to develop computational models to assess change propagation in terms of RCV as shown in Figure 1. Complex network metrics of requirements

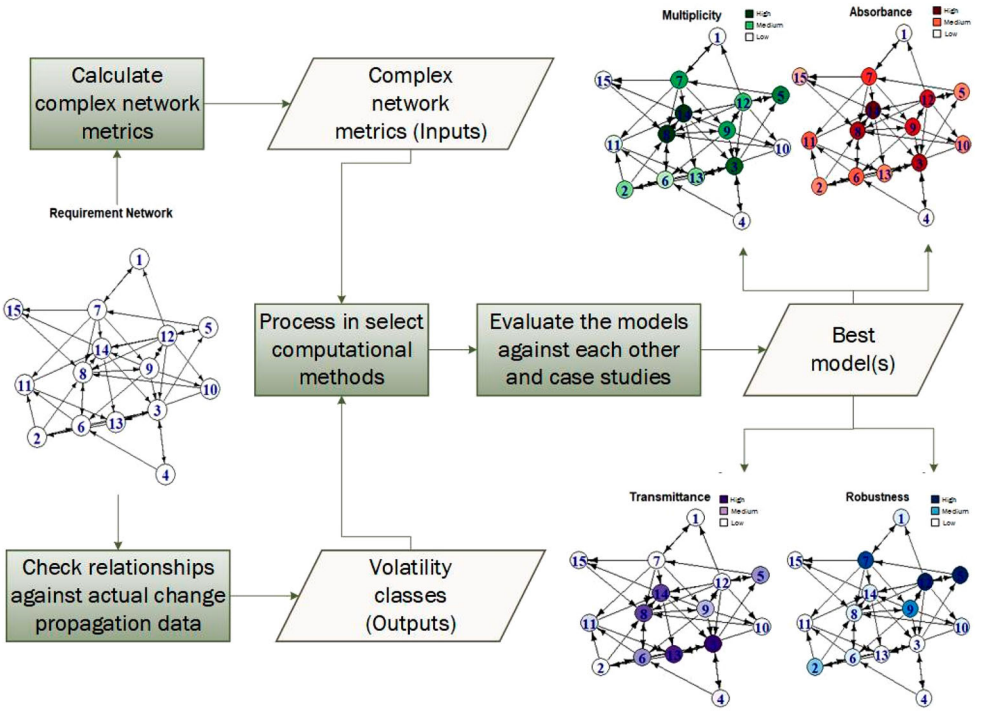


Figure 1. Computational model development.

will be computed from requirement networks obtained by processing requirements gathered from industrial case studies in the R-ARCPP tool. Note that the tool is only utilised to produce effective requirement networks. Modelled requirement relationships will be analyzed against actual requirement change propagation data to determine the four RCV in terms of their metrics: multiplicity, absorbance, transmittance, and robustness. Regression analysis and artificial neural networks will be applied on training and validation data to generate computational models capable of determining RCV using complex network metrics. The highest performing model(s) will be then selected as generalised model(s). Section 2.1 explains RCV classes, section 2.1.2 describes the explored network metrics, and section 0 presents computational methods utilised in this paper.

2.1. Requirement volatility classes

Our prior work identified the ability to classify requirement changes into four volatility classes (Hein et al. 2021) as shown in Figure 2, where requirement B is the requirement of interest. First, requirement B acts as a multiplier, propagating the change from requirement A to C. In the second case, requirement B is an absorber because it is changed by the initial change in requirement A, but it does not propagate the change to requirement C. Requirement B in the third case is a transmitter because it is not changed by the initial change in requirement A, but it does propagate the change to requirement C. In the final case, requirement B is a robust requirement when it is not changed by the initial change in requirement A and it does not propagate the change to requirement C. For instance, some

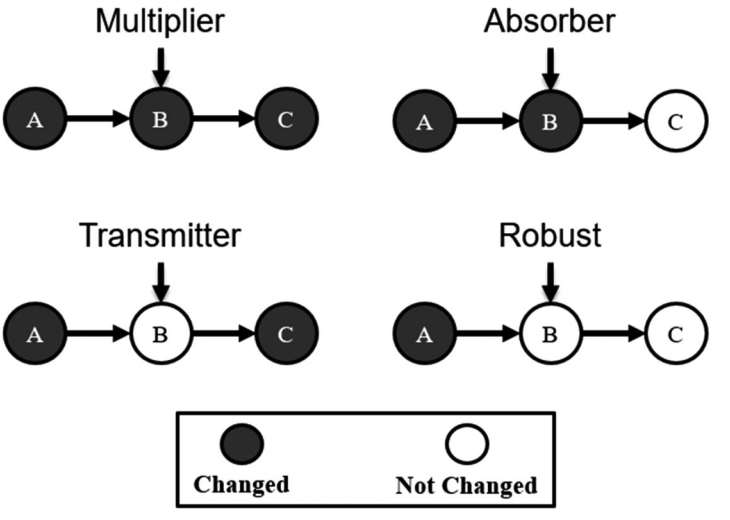


Figure 2. Visual descriptions of RCV.

requirements have a relatively big buffer or design excess (Yadav et al. 2019; Long, Ferguson, and Morkos 2020; Long, Morkos, and Ferguson 2021) to accommodate unexpected engineering changes. Thus, a change can happen in the component, but that requirement had enough excess built in that it could support that change (Long and Ferguson 2020). For example, consider a requirement that states the system must be less than 90C to ensure the adhesive maintains its properties. If another change in the system (for instance, the component the adhesive is holding) increases in temperature from 60C to 70C, the adhesive is affected, but it does not change because it has enough excess to propagate without changing itself. Typically, system engineers would either consult other engineers or make judgments based on their own domain expertise.

2.1.1. Requirement relationships extraction

The first step in determining the volatility classes of requirements is to acquire all potential requirement relationships in the requirement networks obtained from the R-ARCPP tool using industrial projects. Of the three industrial projects analyzed, the first project has 159 nodes and 9865 links, the second project has 214 nodes and 9245 links, and the third project has 202 nodes and 9558 links. Each node represents a requirement, and each link represents a dependency between two nodes. They exhibit small-world properties (relatively high clustering coefficients ~ 0.6 and short average path length ~ 1.8) of real-world system networks (Braha and Bar-Yam 2004, 2007; Braha 2016). As a requirement change can propagate throughout the network along its relationships to other connected requirements and be influenced by the volatility of some requirements, all potential relationships stemming from an initially changed node or source are extracted and analyzed against actual change propagation data. The maximum shortest path length is found as four in all case studies, even though the shortest path length between any two nodes can vary. Therefore, up to a path length and walk length of four are extracted, along with the maximum number of paths from each node to all other nodes are extracted. Cyclic relationships are not considered because there are no repeated changes in the actual change data from the case

Table 1. RCV scoring scheme.

Volatility scores of node on a relationship	Scoring scheme
Multiplicity (M)	Sum of multiplicative inverses of path lengths from it to nodes it multiplies the change to. $M_i = \sum \frac{1}{PL_{i\to j(\{i,j\}\subset\{1,2,3,\ldots,n\},i\neq j)}}$ Both R_i and R_j changed.
Absorbance (A)	Sum of multiplicative inverses of path lengths to it from changed nodes it absorbs the change from. $A_i = \sum \frac{1}{PL_{i\to j(\{i,j\}\subset\{1,2,3,\ldots,n\},i\neq j)}}$ Both R_j and $R_i$changed.
Transmittance (T)	Sum of multiplicative inverses of path lengths from it to nodes it transmits the change to. $T_i = \sum \frac{1}{PL_{i\to j(\{i,j\}\subset\{1,2,3,\ldots,n\},i\neq j)}}$ R_i not changed. R_j changed.
Robustness (R)	Sum of multiplicative inverses of path lengths to it from changed nodes it is not affected by. $R_i = \sum \frac{1}{PL_{j\to i(\{i,j\}\subset\{1,2,3,\ldots,n\},i\neq j)}}$ R_i not changed. R_j changed.

studies. Only modified requirement changes are analyzed as changes due to the addition or removal of requirements are not found in the case study data.

2.1.2. Change instance identification

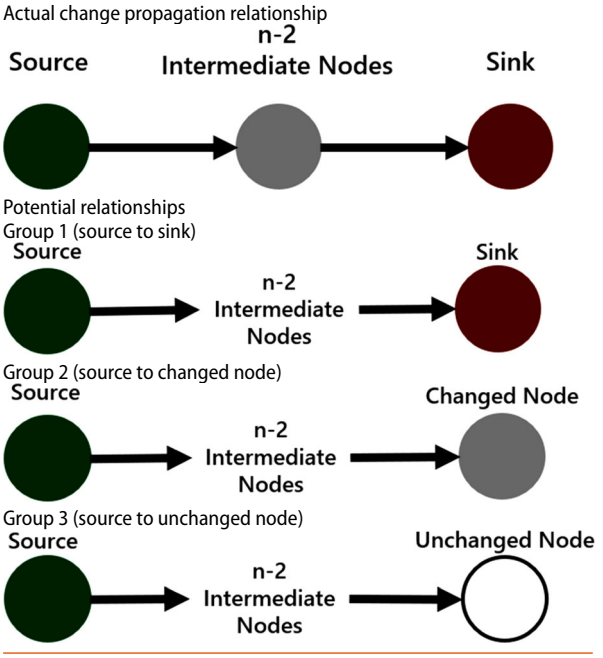
The second step is identifying actual change propagation relationships from actual change instances found in the case studies. This requires a complete analysis of an engineering change notification (ECN) form to map the changes and their propagations to the requirements affected by either initial changes or following changes. The actual change propagation relationship is, therefore, the path along which the initial requirement change propagates, changing all requirements that lie on it. It may contain up to (n) number of requirement nodes and $(n-2)$ intermediate nodes with $(n-1)$ edges.

2.1.3. Volatility metric scoring

The final step is to check all extracted potential requirement relationships against actual change propagation relationships to score each requirement for their volatility classes. Table 1 indicates the scoring schemes used to compute the volatility scores of a requirement, indicating the requirement's ability to multiply, transmit, absorb, or be robust to change, penalised by its path length (PL) distances to other nodes.

As all path length four potential requirement relationships are extracted as presented above, each has $(n=5)$ total requirement nodes, $(n-1=4)$ edges, and $(n-2=3)$ intermediate nodes. Table 2 provides an actual change propagation relationship and three groups of potential requirement relationships, with the initial changed requirement (source node) in green, changed requirements nodes in grey, unchanged requirement nodes in white, and final changed requirement nodes (sink node) in red. They are not necessarily subnetworks as they may make up the entire network based on how connected a project's requirements are. Each group has $2^{n-2}=2^3$ potential types of relationships based on the relative positions of changed intermediate nodes which can be either multipliers or

Table 2. Actual change propagation relationship and groups of potential requirement relationships.



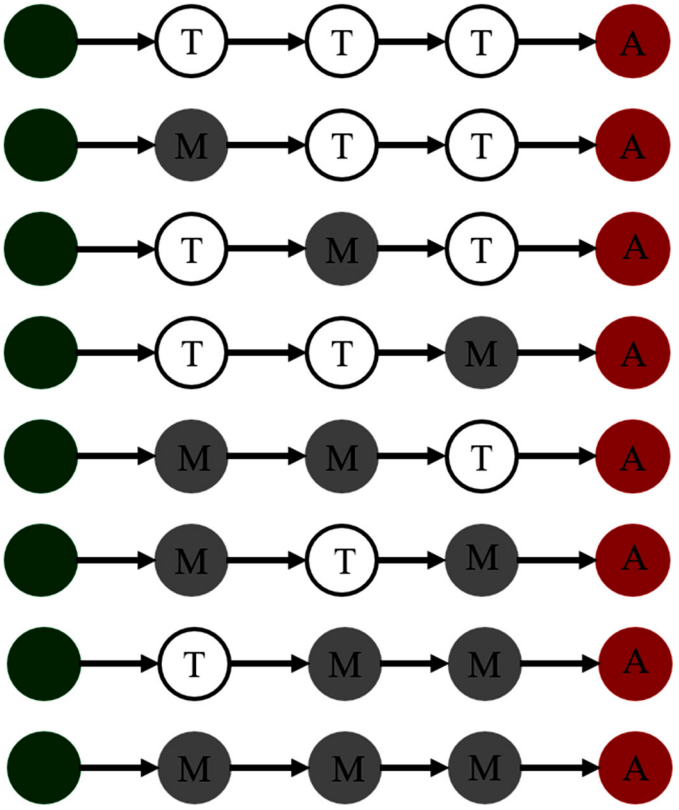
absorbers and unchanged intermediate nodes which can be either transmitters or robust requirements.

The relationships of group 1 are shown in Table 3 as an example to explain how requirements calculate volatility scores. This table represents how a change can stem from a single requirement (source) to propagated requirement (sink) and the various ways it can do so through intermediate requirements. For example, five requirements are employed and assigned as source, transmitter, multiplier, or sink. Across all requirements networks, four arrows represent the maximum shortest path length of information flow. The paths represent all possible combinations of change scenarios. A requirement must remain unchanged between engineering changes before and after to qualify as a transmitter. In reality, this may occur when the transmitter has an overlarge design buffer. The intermediate nodes can be either multipliers or transmitters for both group 1 and group 2, indicating the end nodes are changed. For Path 1C, the source node is not assigned a volatility score as the initial change can be due to any reason and cannot be predicted. The first intermediate node is a transmitter as it is not changed, but it propagates change to the second intermediate node and to the sink. The second intermediate node is a multiplier, as it changed and it propagates change to the sink. The third intermediate node is a transmitter, as it is not changed but propagates change to the sink. The nodes on the remainder of the paths in group 1 and group 2 are scored similarly. Group 3 paths are different in that the intermediate nodes can be multipliers, absorbers, or robust requirements, while the end nodes are robust requirements that are not changed.

The volatility class metric of each requirement is summed over all extracted potential relationships and normalised with respect to the other requirements for all change

Table 3. Example RCV scoring of a requirement relationship.

Group 1: 2^{n-2} potential relationships at $n = 5$



Example volatility scores of requirements on path 1C

1st node (T): Transmitter	$\frac{1}{PL(T \to M)} + \frac{1}{PL(T \to A)} = \frac{1}{1} + \frac{1}{3} = 1.33$
2nd node (M): Multiplier	$\frac{1}{PL(M \to A)} = \frac{1}{2} = 0.5$
3rd node (T): Transmitter	$\frac{1}{PL(T \to A)} = \frac{1}{1} = 1.0$
The sink (A): Absorber	$\frac{1}{PL(S \to A)} + \frac{1}{PL(M \to A)} = \frac{1}{4} + \frac{1}{2} = 0.75$

instances to get a score range between 0-1. After volatility scoring, each requirement i will be attributed by $\overrightarrow{v_{i,j}} = (M_i, T_i, A_i, R_i)_j$ which is the vector of volatility scores for change instance j.

Because of the observed sparsity of the RCV scores, they are divided into multiple ordered categorical levels to improve the granularity and distributions of the data to obtain robust computational models. For example, transmittance is divided at two thresholds into three levels to classify requirement as having high, medium, or low transmittance. The volatility class levels serve as categorical outputs or dependent variables in the explored computational methods.

A review of current literature shows requirement change propagations are defined in different ways. Eckert, Clarkson, and Zanker (2004), Cheng and Chu (2012), Ma et al. (2017b), and our papers focus on different measurements and capture different types of information regarding change. Eckert and Cheng's paper demonstrates how changes in information propagate across a physical domain (Eckert, Clarkson, and Zanker 2004). The basic change instance was later studied and redefined in Ma's paper, which categorises change statuses based on design properties (Ma, Jiang, and Liu 2017a). To capture each instance of requirement change, our approach provides volatility scoring measurements within requirement networks, assuming that each requirement may alter its role. This study has developed a framework to understand the phenomenon of change propagation with an emphasis on a single change at a time within requirements (Li et al. 2022a), while also defining metrics for estimating the probability of RCV classes.

2.2. Complex network metrics

A comprehensive taxonomy of complex network metrics from the literature is adapted to meet the exploratory nature of this analysis (Menon 2015; Haibing et al. 2021). The four categories of these metrics are distance and clustering metrics, centrality metrics, subgraph and community metrics, and modified pairwise descriptive metrics. Distance and clustering metrics refer to a node's distance to others. For example, the clustering coefficient metric, measures how a node and its neighbours are clustered together to form closely related clusters (Newman 2005). Centrality metrics refer to metrics based on a node's importance. For example, betweenness centrality of a node represents the extent to which a node lies on paths between other nodes (Newman 2010). Subgraph and community metrics observe a node's properties in relation to the subgraph or community it belongs to. Pairwise descriptive metrics – for example, walk length and path length – quantify only the interactions between change initiated and propagated node pairs in Menon (2015), however, they are modified in this study because four RCV classes of all requirement nodes are determined from their corresponding network metrics which are dependent on all requirement network relationships. Therefore, as described in section 2.1.1, up to path length four, up to walk length four, and the maximum number of paths from individual requirement nodes to all other requirement nodes are considered. Metrics accounting for relationship directions, such as in-degree and out-degree – representing a requirement node's connectivity regarding the number of relationships incident to it and originated from it – are also considered. The full list of selected complex network metrics is shown in Table 4 with corresponding references.

All network metrics, apart from some categorical values, are normalised to obtain volatility score range of 0–1. Each requirement i will be attributed by $\overrightarrow{x_i} = (x_{i1}, ..., x_{iD})$ which is the vector of requirement i with D complex network metrics. These metrics serve as inputs or independent variables in the explored computational methods. It is important to note that although network metrics can be computed for any model-based strategies, doing so from the requirement networks proven in the literature to be capable of characterising requirement change propagation may reveal meaningful results that could be used to understand the change volatility phenomenon.

Recent literature has implemented complex network theory to represent a complex product design for exploring the topological properties of networks. Using network

Table 4. Selected network metrics.

ID	Category	Name	Source
1	Distance and Clustering Metrics	Eccentricity	Dankelmann et al. (1999)
2		Efficiency	Latora and Marchiori (2001)
3		Vulnerability	Costa et al. (2007)
4		Clustering Coefficient	Newman (2005)
5		Degree	Ma et al. (2017b), Newman (2010)
6		In – Degree	Newman (2010)
7		Out – Degree	Newman (2010)
8		Average Neighbour Degree	Barrat et al. (2004)
9	Centrality Metrics	In – Closeness Centrality	Newman (2010)
10		Out – Closeness Centrality	Newman (2010)
11		Delta Centrality	Latora and Marchiori (2007)
12		Bridging Centrality	Hwang et al. (2006)
13		Bridging Coefficient	Hwang et al. (2006)
14		Random Walk Centrality	Bollobás (2001)
15		Betweenness Centrality	Newman (2010)
16		Counting Betweenness	Bloechl et al. (2010)
17		Flow Coefficient	Honey et al. (2007)
18		Total Flow Paths	Honey et al. (2007)
19		Brokering Coefficient	Konganti et al. (2013)
20		Right Eigenvector Centrality	Newman (2010)
21		Left Eigenvector Centrality	Newman (2010)
22		Alpha Centrality	Bonacich and Lloyd (2001)
23		Katz Centrality	Newman (2010)
24		Subgraph Centrality	Estrada and Rodriguez-Velazquez (2005)
25		Coreness Centrality	Newman (2010)
26		PageRank	Newman (2010)
27		HITS: Authority Centrality	Newman (2010)
28		HITS: Hubs Centrality	Newman (2010)
29	Subgraph and Community Metrics	Participation Coefficient	Rubinov and Sporns (2010)
30		Within Module Degree	Rubinov and Sporns (2010)
31		Communities	Newman (2010)
32	Modified Pairwise Descriptive Metrics	Out – Path Length $\subseteq$ 1,2,3,4	Newman (2010)
33		In – Path Length $\subseteq$ 1,2,3,4	Newman (2010)
34		Out – Walk Length $\subseteq$ 1,2,3,4	Rucker and Rucker (2000)
35		In – Walk Length $\subseteq$ 1,2,3,4	Rucker and Rucker (2000)
39		Out – Maximum Number of Paths	Newman (2010)
40		In – Maximum Number of Paths	Newman (2010)

analysis, a weighted graph is designed to describe the phenomena of engineering changes. Each element of a complex system is represented as a node and the connections between elements are represented by lines. In the literature, many different network indicators are proposed to capture information changes within a complex system. Different measurements, including centrality measurement (Orouskhani, Shi, and Orouskhani 2021), design specification flow (Ma, Jiang, and Liu 2017a; Ma, Jiang, and Liu 2016), change propagation intensity (CPI) (Ma et al. 2017b), changeability indices (Cheng and Chu 2012), global assessment indices (Guodong et al. 2017), and network change rate (Dong et al. 2022), are studied that capture design information changes and support decision making (Shabi et al. 2021).

2.3. Explored computational methods

Logistic regression (LR) and artificial neural networks (ANNs) are explored due to their successes in classification problems in the literature similar to the RCV prediction in this research. Linear regression can determine the linear relationship between inputs and output that take a continuous value. When the output is categorical, the intention is then to

predict the probability P that is 1 or 0. As the characterisation of RCV using complex network metrics is one such problem, LR models that can deal with categorical outputs are considered. ANNs are computational models based on the function of a biological neuron which receives inputs from its dendrites, processes signals in its soma, and outputs the final decision by its axon. There are several popular ANNs in the literature, details of which are not discussed here for brevity. As LR and ANN models can deal with only one categorical output at a time, four RCV data will be separated into four individual data sets, each for a RCV class as an output.

2.3.1. Ordinal output cumulative logit regression model

The binary output logistic regression (BOLR), employing the binomial probability theory to determine the probability of the output belonging to one category or the other, would suffice to assess the relationship of the inputs with the single dichotomous output (Agresti 2003). If the output has $r > 2$ categories, multinomial LR should be used (Agresti 2003). If the output is nominal, the nominal output baseline category logit regression (NOLR) model can be employed (Agresti 2003). As the RCV metrics are divided into low, medium, and high levels, the LR model of interest in this paper is the ordinal output cumulative logit regression (OOLR) model that can deal with the ordinal output with $r > 2$ categories and account for the ordering of outputs by using cumulative probabilities up to a threshold, making the ordinal categories binary at the threshold (Bender and Grouven 1997). For a data set that has N number of cases ($i = 1, \ldots, N$), the outputs are represented as $y_i = (y_{i1}, y_{i2}, \ldots, y_{ir})$, and x_i representing the vector of length p inputs with corresponding p number of contribution coefficients b, for case i, a cumulative probability of the output less than or equal to category j is $P(Y \leq j) = \pi_1 + \ldots + \pi_j$ and the cumulative logit is defined as in the equation below (Agresti 2003; Hosmer and Lemeshow 2000). This logit describes the log odds of two cumulative probabilities of a response within category j or below versus in a category higher than j.

$$\text{Logit}_j = \log\left(\frac{P(Y \leq j)}{P(Y > j)}\right) = \log\left(\frac{P(Y \leq j)}{1 - P(Y \leq j)}\right) = \log\left(\frac{\pi_1 + \ldots + \pi_j}{\pi_{j+1} + \ldots + \pi_J}\right)$$

Then, incorporating inputs, this equation becomes (Agresti 2003):

$$\text{Logit}_j = a_j + b_j x$$
$$\text{where, } j = 1, \ldots, r-1$$

The OOLR model can use all $(r-1)$ logits by simplifying to make b_j identical across the $(r-1)$ logit equations with cumulative probability as shown in the equations below (Agresti 2003).

$$L_j = a_j + bx$$
$$\text{where, } j = 1, \ldots, r-1$$

$$P(Y \leq j) = \frac{e^{(a_j + bx)}}{1 + e^{(a_j + bx)}}$$

This model has different $(r-1)$ intercept terms and p number of inputs with corresponding coefficients b which are the same across all logit equations with a total of $(r + p - 1)$

parameters estimated (PennState Eberly College of Science). The different intercept a_j is the log odds of falling into or below category j when all $x = 0$ and are not used in model interpretation. The k^{th} element of b_j is the change in log odds of falling into or below category j with one unit change in k^{th} input, with others held constant (PennState Eberly College of Science), indicating the impact of the input. For model fitting, such as the BOLR, the OOLR model employs the maximum likelihood method in which the contribution coefficients maximise the likelihood function which expresses the probability of obtaining the observed output are estimated (Hosmer and Lemeshow 2000). The significance of the coefficients is checked by using the log likelihood ratio test, with its test statistic and corresponding p-values, that compares the log likelihood of the model with only the intercept term, with the model under consideration for the null hypothesis that all coefficients in the fitted model are equal to zero against the alternative hypothesis that at least one coefficient is not equal to zero (Hosmer and Lemeshow 2000). The log likelihood function of the model is as shown the equation below (Agresti 2003).

$$L(b) = \log\left(\prod_{i=1}^{N}\prod_{j=1}^{r}\pi_{ij}^{y_{ij}}\right) = \log\left(\prod_{i=1}^{N}\prod_{j=1}^{r}(P(Y \le j|x_i) - P(Y \le j-1|x_i))^{y_{ij}}\right)$$

$$L(b) = \log\left(\prod_{i=1}^{N}\prod_{j=1}^{r}\left(\frac{e^{(a_j+bx_i)}}{1+e^{(a_j+bx_i)}} - \frac{e^{(a_{j-1}+bx_i)}}{1+e^{(a_{j-1}+bx_i)}}\right)^{y_{ij}}\right)$$

This model is the most suitable model for the ordinal output RCV data because the interpretation of results is initiative due to the convenience that the effect of an input can be quantified by one coefficient across all output categories (Bender and Grouven 1997). More details of this model can be consulted in the respective references.

2.3.2. Backpropagation artificial neural networks

An artificial neuron that receives input signals from inputs and map the output signal is denoted as the perceptron (Basheer and Hajmeer 2000). The BPANNs consist of input, hidden, and output layers of perceptrons for capturing relationships. It uses a transfer function along with error correction learning rules to adjust weights based on the computed error between the output and the correct solution, which is propagated backward from the output layer to the hidden layer to the input layer (Basheer and Hajmeer 2000). In BPANNs, each input neuron in the input layer produces the input value forward to each hidden neuron. The net effect of each hidden neuron is calculated by the dot product of the input value and its randomly initialised corresponding connection weight. After the net effect at the hidden neuron is obtained, the activation of that neuron is determined by processing the net effect in the transfer function – such as the sigmoid function – to yield the output, typically 0 and 1, or $+1$ and -1. This activation is the new signal value fed forward to the hidden layer or the output layer. The same procedure is repeated until the output layer neurons produce the solution of the ANNs that may be different from the actual target solution due to arbitrary random weights. This difference becomes the error that is propagated back from the output layer through all hidden layers to the input layer to adjust the connection weights. This operation is iterated to search for the error surface as a function of connection weights until it is minimum.

There are several parameters influencing the computational performance of the BPANNs. They include: *weight initialisation* – different ways of initialising interconnection weights, *transfer functions* – different activation functions determining the net effect and different transfer functions determining the output based on the activation, *training modes* – different ways of presenting data to the BPANNs affecting how the error is calculated and backpropagated, *convergence criteria* – different criteria for the error convergence, *number of training cycles* – number of cycles or epochs determined by trial and error to monitor the error, *number of hidden layers (NHL) and number of hidden* neurons *(NHN)*, *learning rate* (η) – step size of weight updates, *momentum coefficient* (μ) – a parameter accounting for previous magnitude and direction of the weight to the current updating step to direct the error surface search to the global minimum. These parameters are influenced by problem types, data sets, domain areas, and their thumb rules or experimental values along with their advantages and disadvantages can be consulted in respective references.

2.4. Implementation of computational models

The levels of each RCV class are coded as ordinal factors (i.e. low = 1, medium = 2, high = 3) in the OOLR models and as one hot encoding (i.e. low = 100, medium = 010, high = 001) in the BPANNs. The continuous inputs are normalised within the range of 0.01-0.99 and the categorical inputs are coded as nominal factors. The data holdout method is employed by randomly sampling 75% of the data into the training set and 25% of the data into the testing set. This holdout validation is repeated three times for each of the three case studies for each RCV class.

Traditionally, stepwise methods that start from the full model (backward elimination) or the null model (forward selection) are used in regression to iteratively remove or add input variables based on their significance to improve the fitted model. The addition or removal of inputs is done by conducting several hypothesis tests such as the likelihood ratio test which can cause problems in choosing a relevant significant level (Calcagno and de Mazancourt 2010). Another way is to use Information Criteria (IC) to compare models at each step. Because stepwise iterative models depend on the starting point and the stopping rules, forward selection and backward elimination may not converge in the same model (or) none may converge to an optimal model based on the type of IC (Calcagno and de Mazancourt 2010). While the stepwise successive model comparison does not make full use of IC which can compare several models at a time, a full IC based approach using IC, such as Akaike Information Criterion (AIC), can compare all candidate models based on ranked IC values (Calcagno and de Mazancourt 2010). The OOLR models are developed through a full IC based approach to identify the best model from a candidate set of models using genetic algorithms (GA) by exploring faster only a subset of potential models randomly, but with bias towards some models. For each of the three holdout data sets, the GA is computed ten generations accounting for the variations in the initial model populations. For each GA generations, the best model out of three validation sets is selected to finally obtain ten best models for each RCV class for further comparison. This implementation is performed using R package *GLMULTI* (Calcagno and de Mazancourt 2010) and the OOLR models are fitted using R package *MASS* with default recommended parameters. Only such models that result

in AUC values greater than the random performance value of 0.5 are compared based on the significant differences across the evaluation measures.

For the BPANNs, the learning function used is the backpropagation algorithm with the parameters, learning rate (η), momentum coefficient (μ), error flat spot elimination value added to the derivative of the activation function to pass flat spots of the error surface (c), and maximum tolerated difference between input and output $(d_{\max})$ with their values $(0.2, 0.9, 0.1, 0)$. η and μ values are close to the recommendations by (Swingler 1996), and $d_{\max}$ and c values are default values in Bergmeir and Benítez (2012). The most favourable weight update mode for feedforward networks as recommended is in topological order to process the network layer by layer from the input to the output layer (Bergmeir and Benítez 2012). The weights and the bias are randomly initialised with distributed random values within the range of its default parameter values $(-0.3, 0.3)$ based on the recommendation by ASCE (2000). The NHN is varied in single hidden layer with a maximum of one hundred iterations with the batch training mode. As the NHN are dependent on problem types, data sets, ANNs types, etc. (Sheela and Deepa 2013), it is arbitrarily varied from 2 neurons to 150 neurons with 2 neuron increments totalling 75 BPANNs. The activation function for hidden and output neurons is the logistic sigmoid function. This implementation of the BPANNs is performed using the *RSNNS* package in R, a library interface to the Stuttgart Neural Network Simulator (SNNS) developed at the University of Stuttgart (Bergmeir and Benítez 2012). The rest of the parameters are deployed at default recommended values. Accounting for the variations in the randomly initialised weights, the NHN incremental loop from 2 to 150 is performed ten iterations and the results are averaged for each BPANN. As in OOLR models, only a few BPANNs that result in AUC values greater than 0.5 are compared.

2.5. Evaluation measures of computational models

The evaluation measures deployed in this study are derived from a confusion matrix based on the number of correctly predicted data points indicated by true positives (TP) and true negatives (TN) and the number of incorrectly predicted data points indicated by false negatives (FN) and false positives (FP) (Vihinen 2012). The models predict the class in terms of a numerical value of probability that exceeds a predefined threshold cut off value based on the number of true and false predictions varies (Vihinen 2012). The evaluation measures are calculated using the resulting number of true and false predictions. A combination of different measures, rather than a single measure, is used to capture the information from the contingency matrix to gauge a model's performance (Vihinen 2012). For categorical output, there are two common groups of measures: discrimination and calibration. Discrimination measures evaluate the ability of the model to distinguish between data points with and without the presence of the outcome (Steyerberg, Van Calster, and Pencina 2011). The discrimination measures used in this study include positive predictive value (PPV) or precision, true positive rate (TPR) or recall or sensitivity, specificity, accuracy, F1 score, and area under receiver operating characteristic (ROC) curve (AUC) with values ranging from 0 to 1, indicating 1 as the most desirable. Calibration refers to the agreement between the predicted probabilities and observed outcomes (Steyerberg, Van Calster, and Pencina 2011). The calibration measure employed in this study is Hosmer-Lemeshow goodness of fit, or HL statistics, which compares the mean predicted probability to the mean observed outcomes

of groups of data based on specific grouping schemes (Steyerberg, Van Calster, and Pencina 2011). The smaller the value, the more accurate the calibration is close to 0.

2.6. Statistical comparison of computational models

In observing the distribution of the evaluation measures, it is found that they do not follow normality for all RCV classes. Therefore, performance comparison cannot be achieved through a typical analysis of variance – which assumes normality of the data – rather, the Friedman test, a nonparametric test without distributional assumptions, is performed to identify which model differs from which others (Salkind 2006). The test statistic is approximated by the chi-squared distribution with $(K - 1)$ degrees of freedom and corresponding p-value (Salkind 2006). If the p-value is less than the significance level, the alternative hypothesis indicates that at least one differing model exists (Salkind 2006). When the Friedman test's null hypothesis is rejected, statistical tests namely post hoc multiple comparison tests are used to identify which model is different. Here, we use the Nemenyi test, to determine which models different significantly if their mean ranks differ by at least one Nemenyi's test statistic, critical difference (CD). The Friedman post hoc test for multiple comparisons and its test statistic is also performed. Additionally, the p-values of the test are adjusted using Shaffer's static procedure. If the corresponding p-value between any two models is less than the significant level, the alternative hypothesis indicates that they are different in performance.

2.7. Case studies background

As the major objective of this research is to develop computational models that can be deployed in real world applications, it is important that these models can be generalised and validated against industrial studies using correctly documented requirement change data collected from diverse projects. The case study projects utilised involve an automated yarn creel system, an exhaust gas recirculation flap, and steel pipe threading stations, with project timelines ranging from 0.75 to 2 years and project costs ranging from 700,000 to 2,500,000 in USD. Requirements and requirement change propagation data containing 16 requirement change instances documented from these case studies are analyzed. There are a total 575 requirements written in varying format by different stakeholders and engineers. Moreover, the studies can point out the models' ability to assist designers and engineers in managing the engineering change propagations in the requirement domain along the design process (Morkos 2012).

3. Results and discussions

For both the OOLR models and BPANNs, even though there are differences in raw rankings, there is no statistically significant performance difference found across all evaluation measures for the compared models at 0.05 significance level for all RCV classes according to the Friedman test p-values, Nemenyi test's CD, and Friedman post hoc test with Shaffer's adjusted p-values. Therefore, in the case of the OOLR models, the raw rankings are used to select the top 50% of the best performing models across all evaluation measures and then intersected to select the best model. If there are multiple models or no model in this

intersected set, the model that ranks higher more frequently than others is selected as the best model. In the case of the BPANNs, the BPANN with no significant performance difference with the least number of NHN are selected to avoid potential overfitting by assigning a higher number of NHN.

The performance of the best OOLR models and the best BPANNs for multiplicity and absorbance are observed no difference from random due to their AUC values of 0.5, therefore, their details are not presented here. The accuracy values are also observed high because the predicted multiplicity and absorbance levels by the models can be compared with data from actual measurements, which only has little few change multipliers and absorbers in the analyzed requirement changes. As there are only a few change multiplying and change absorbing requirements in the actual data and the majority on the extracted relationships have zero or low multiplicity and absorbance, the models may be predicting these majority levels resulting in high accuracy.

The best OOLR models for transmittance and robustness are shown in Tables 5 and 6 with their respective evaluation measures and selected complex network metrics with corresponding coefficient values. The AUCs of the transmittance OOLR model and the robustness OOLR model are 0.82 and 0.78 respectively, indicating more accurate model prediction, and other evaluation measures are also at moderate or high values. Transmittance of a requirement increases with increase in efficiency, participation coefficient, flow coefficient, within module degree, incoming walk of length 2, and outgoing path of length 3 based on their coefficient values. Transmittance decreases with an increase in incoming paths of length 4 and in-degree, and if the node belongs to cluster number 2 based on their coefficient values. Robustness of a requirement increases with an increase in participation coefficient, total flow paths, within module degree, outgoing path of length 3, and eccentricity. Robustness decreases with increase in the coreness centrality, outgoing walk of length 2, outgoing path of length 4, subgraph centrality, and left eigenvector centrality. Although there exist subtle differences in definitions between the models' network metrics according to the references in section 0, the metrics can be essentially grouped together based on similar network properties they represent. In the transmittance OOLR model, efficiency, participation coefficient, flow coefficient, and within module degree represent a node's ability to facilitate information flow in its neighbourhood. Incoming walk of length 2, outgoing path of length 3, and incoming path of length 4 represent a node's indirect connections for information flow. *Community:2* represent the membership of a node to the cluster number 2 and in-degree represent a node's information reachability through adjacent neighbours. In the robustness OOLR model, total flow paths, and within module degree represent a node's ability to facilitate information flow in its neighbourhood. Outgoing path of length 3, outgoing walk of length 2, and outgoing path of length 4 represent a node's indirect connections for information flow. Eccentricity and subgraph centrality represent a node's information reachability through all other nodes. Left eigenvector centrality represents a node's information transmissibility through adjacent neighbours.

The key takeaways drawn from the OOLR model development are as follows. The coefficient value of an individual complex network metric equals the change in the log odds function of a requirement belonging to a RCV class metric level with one unit change in that complex network metric while others are kept constant. Therefore, the function is dependent on all the complex network metrics of the model and the metrics with larger positive or negative coefficient values will dominate the output levels of the RCV classes more. The

Table 5. OOLR model for transmittance and robustness.

Transmittance OOLR model		
Complex network metrics	Coefficient values	Property
Efficiency	14.00	Ability to facilitate information flow in its neighbourhood
Participation coefficient	13.44	
Flow coefficient	6.77	
Within module degree	5.31	
In-walk length = 2	34.65	Indirect connections for information flow
Out-path length = 3	9.16	
In-path length = 4	−32.42	
Community:2	−1.06	Membership to cluster 2
In-degree	−8.00	Information reachability through adjacent neighbours

Evaluation measures						
Precision	Sensitivity	Specificity	Accuracy	F1	AUC	HL stats
0.66	0.62	0.86	0.83	0.63	0.82	9.26

Table 6. OOLR model for robustness.

Robustness OOLR model		
Complex network metrics	Coefficient values	Property
Participation coefficient	8.50	Ability to facilitate information flow in its neighbourhood
Total flow paths	7.08	
Within module degree	3.17	
Coreness centrality	−6.80	
Out-path length = 3	208.64	Indirect connections for information flow
Out-walk length = 2	−90.75	
Out-path length = 4	−123.25	
Eccentricity	4.30	Information reachability through all other nodes
Subgraph centrality	−27.19	
Left eigenvector centrality	−5.22	Information transmissibility through adjacent neighbours

Evaluation measures						
Precision	Sensitivity	Specificity	Accuracy	F1	AUC	HL stats
0.61	0.57	0.81	0.81	0.58	0.78	18.75

log odds function of a RCV class is independent of other RCV classes and represents only that RCV. As discussed in section 2.1.3, just because the requirement is less of a transmitter does not mean it is less or more of any of the other three RCV classes. It can belong to any level of multiple classes simultaneously. The performance of the OOLR models for multiplicity and absorbance are no difference from random due to the data imbalance, and only the transmittance model and the robustness model can be considered as valid generalisations. Although there are remedial solutions for data imbalance in the literature such as synthetic network simulation, data sampling methods, random noise injection, etc., they could add on more uncertainties that can still result in inconclusiveness with no guarantee that the results would be more robust and reproducible. Therefore, the authors present the current findings as they are and plan to conduct future analyses to enhance the quality of data.

In the transmittance and robustness models, there are several groupings of network metrics that represent specific common network properties useful in predicting the RCV classes shown in Tables 5 and 6. This is akin to the literature findings in seminal work representing real world complex systems by network models that suggest focusing on specific network metrics, such as the centrality metrics, and other strongly correlated ones can

communicate relevant information regarding system dynamics including design changes and their propagations (Braha and Bar-Yam 2004, 2007; Braha 2016). Having the ability to absorb and generate information, highly connected nodes were initially demonstrated as an essential component of propagating design changes across the engineering network (Braha and Bar-Yam 2007). A centrality measurement is built on this basis to explain the different dynamic properties of the network of design requirements. This finding was presented as evidence that employing comprehensive complex network metrics in the OOLR models can identify a set of network properties can characterise RCV classes using highly interconnected nodes. Before implementing a proposed requirement change, a designer can review such network metrics from the requirement network model to predict the RCV of other requirements in response to that change. Future studies based on this premise will attempt to set strategies to formalise and develop computational tools that can manage the RCV along the systematic design process. It is also observed that there are no causal relationships between the RCV classes and the complex network metrics. For example, in the transmittance model, higher in-degree values indicate lower change transmittance, but this does not mean that the requirement has lower change transmittance because it is pointed to by many adjacent neighbours. The OOLR model implementations here realise a data based solution that serves similar to the classical black box problem (Bathaee 2018), the authors have yet to recognise how each metric causally influences the RCV classes.

As shown in Tables 7 and 8, the AUC values of the BPANNs for transmittance and robustness are at 0.68 and 0.58 respectively, indicating more accurate prediction performance than the results shown for multiplicity and absorbance with moderate values of other evaluation measures. The bar plots of the variable importance weights of complex network metrics for the medium output level of transmittance and robustness are also shown in the tables as examples for visualisation of the individual complex network metric's importance or effect on the output (Olden, Joy, and Death 2004). It can be noted that the performance of the OOLR models is superior to some BPANNs for both transmittance and robustness. Unlike the OOLR implementations, the BPANNs cannot select the most useful subset of complex network metrics, but rather provides the variable importance weights as an indicator of relevance of network metrics on the RCV outputs. Casual inspection also cannot observe any trend in the network metrics' importance rankings across output levels of the RCV classes. Therefore, it may be worthwhile to explore if similar ranking of network metrics can be determined using more computationally efficient feature selection (FS) filter methods which are algorithm-independent statistical methods capable of ranking the inputs based on relevancy scores with regard to the targeted output (Kumbhar and Mali 2016; Bharti and Singh 2014). The relevance rankings of network metrics by the selected FS methods reviewed from the literature, namely, Correlation Criteria (CC), Chi-Square (CHI), Information Gain (IG), Minimum Redundancy-Maximum Relevance (mRMR), and Relief (RLF) method, are explored for their correlation with the importance weight rankings by the BPANNs using Spearman correlation coefficient.

Figure 3 depicts the significant correlations between the rankings by the FS methods and results from the BPANNs for low, medium, and high levels of RCV classes. For multiplicity and absorbance, the rankings by BPANNs for medium and high levels have perfect positive correlation to each other and strong positive correlation to the ranking by CC. The ranking for low level has near perfect negative correlation for medium and high levels and strong negative correlation to the ranking by CC. For transmittance and robustness, the rankings

Table 7. BPANN for transmittance.

Transmittance BPANN						
Precision	Sensitivity	Specificity	Accuracy	F1	AUC	HL stats
0.47	0.46	0.78	0.75	0.45	0.68	40.34

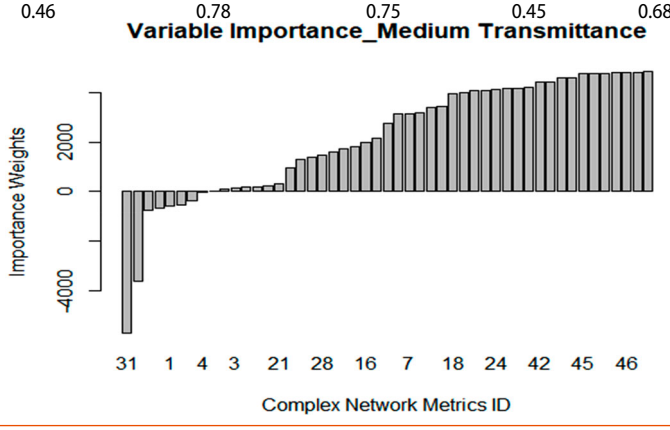
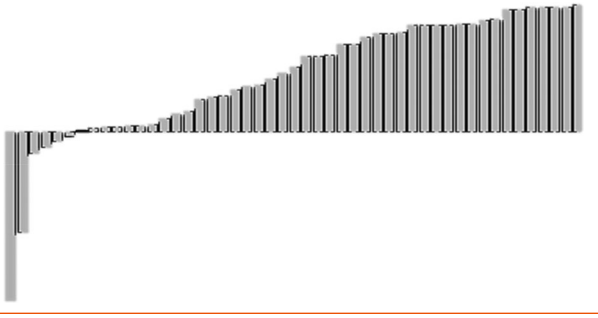


Table 8. BPANN for robustness.

Robustness BPANN						
Precision	Sensitivity	Specificity	Accuracy	F1	AUC	HL stats
0.31	0.41	0.70	0.78	0.36	0.58	12.517



by BPANNs for medium and high levels have near perfect positive correlation to each other and strong positive correlation to the rankings of CC, IG, and CHI in decreasing order. The ranking for low level has near perfect negative correlation to such values for medium and high levels and strong negative correlation to the rankings of CC, IG, and CHI in decreasing order. Based on these findings, the relevance scores rankings of complex network metrics by CC can be assessed beforehand to determine important metrics for the BPANNs: results with higher CC ranks can have high positive importance weights in the BPANNs for medium and high levels, and the values with lower CC ranks can have significant positive importance weights in the BPANNs for the low level in all RCV classes. However, this assessment is not performed in this paper due to the expensive computational nature of the BPANNs.

The key takeaways drawn from the OOLR model development are as follows. The BPANN of a RCV class is independent from other RCV classes and is dependent on all the complex network metrics, influenced more by higher positive or negative importance weights. The performance of the BPANNs for multiplicity and absorbance are no difference from random

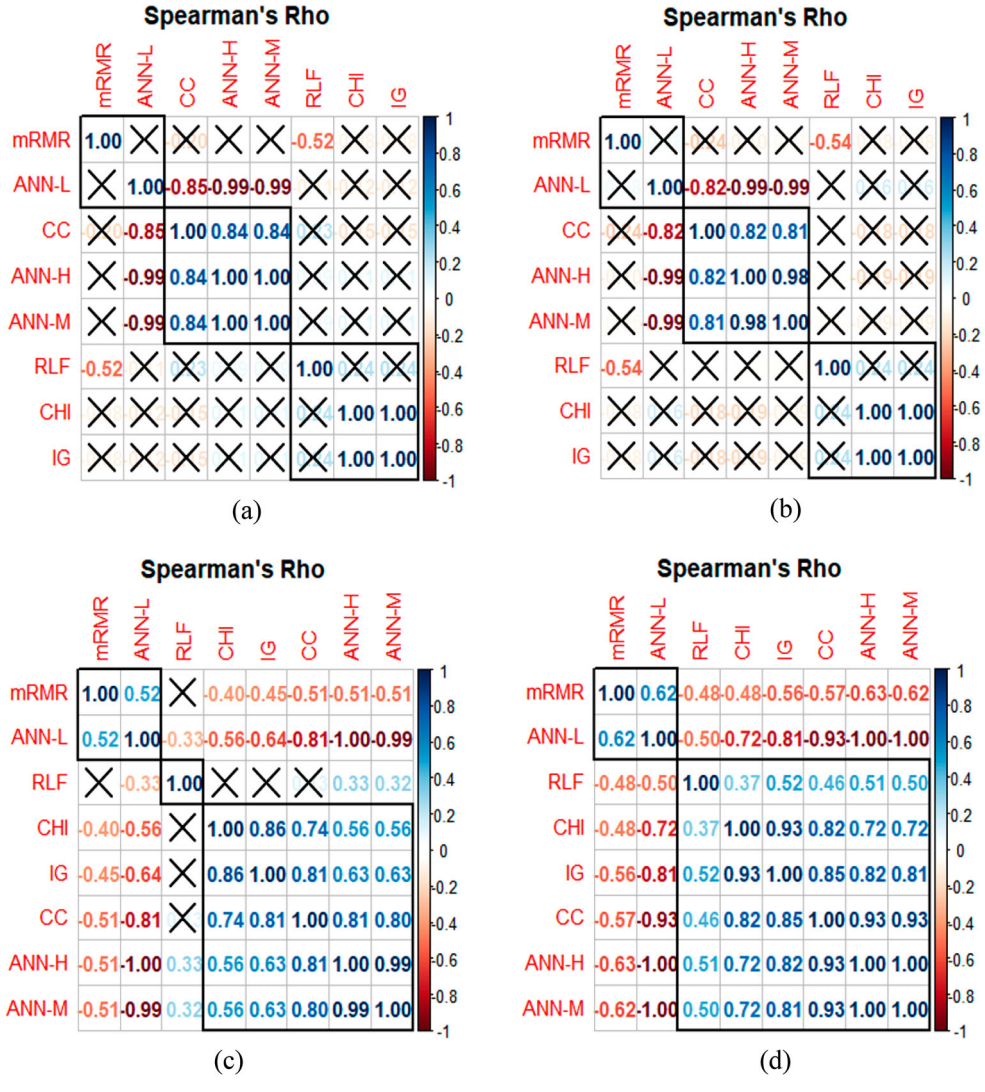


Figure 3. Spearman correlation matrix between relevance scores rankings of FS filter methods and importance weights rankings of BPANNs: (a) multiplicity (b) absorbance (c) transmittance (d) robustness.

due to the data imbalance, and only the transmittance BPANN and the robustness BPANN can be considered as valid generalisations. However, the performance of the OOLR models for transmittance and robustness are superior to the BPANNs based on modelled requirement network, analyzed data, and current algorithm parameters. Based on the correlation analysis between the relevance scores rankings of network metrics by the FS filter methods and the importance weights rankings of network metrics by the BPANNs, different cutoffs on the rankings of the CC method can be applied to the experiment with inclusions or exclusions of s network metrics and their effects on the RCV prediction performance can be observed. However, due to the inability to provide explanations and extrapolate solutions

outside of training data, repetitive parameter tunings, and unsatisfactory performance presented in the results, the BPANNs are not encouraged for use in characterising RCV classes without further research.

4. Conclusions, recommendations, and future work

This study theorised and analyzed requirement behaviour in response to every instance of change: (1) multiplier, (2) absorber, (3) transmitter, and (4) robust in terms of their respective metric values. Using these definitions, we have developed a framework for implementing OOLRs and BPANNs to determine RCV class output levels using complex network metrics as inputs. In addition to its theoretical contribution, this framework provides new insights into its practical implications through three industrial applications. Additionally, we discuss the reasons and challenges involved in implementing this proposed method for real-world projects with corresponding recommendations. As a result of data scarcity from these projects, only the OOLR and BPANN models are considered valid generalisations for transmittance and robustness. As requirements change frequently and unpredictable, BPANNs are not recommended for predicting RCV analyses without sufficient data. The rankings of network metrics by the FS filter methods, such as the CC method, that are found strongly correlated to the variable importance weight rankings by the BPANNs, should be explored further for use in combination with promising computational models as well.

The OOLR models utilise IC based GA method to select the best model containing a subset of complex network metrics with corresponding contribution coefficient values as shown in Tables 5 and 6. As the solutions provided by the OOLR models are similar to that of the classical black box empirical problem, the authors yet need to investigate how complex network metrics influence the RCV classes causally. However, several groupings of complex network metrics representing specific network properties useful in predicting the RCV classes are observed. For example, in the transmittance OOLR model, efficiency, participation coefficient, flow coefficient, and within module degree representing a node's ability to facilitate information flow in its neighbourhood. This is in agreement with the literature findings (Braha and Bar-Yam 2004, 2007; Braha 2016; Zheng et al. 2017) that recommended to focus on several characteristics of networks that model real complex engineering systems to capture system dynamics such as design changes, which in the case of this paper, is the change volatility. Based on this premise, after future studies and validations, formalised engineering guidelines, strategies, and computational tools can be developed around complex network properties to effectively manage RCV before implementing changes. If managing RCV can be introduced in industry (DelSpina et al. 2018) and academia (Joshi, Morkos, and Summers 2018; Morkos, Joshi, and Summers 2019), how requirements are managed can be one immediate area of impact. As the way requirements are elicited and written (i.e. characteristics including but not limited to natural language usage, systems and components boundaries, levels of details, requirement types, requirement coupling, and requirements topic (Chen, Mullis, and Morkos 2021) etc. that may define how closely a requirement relates to others and what information does it contain) can affect the network model and resulting characterisation of RCV, improved practices and standards may develop to capture RCV more accurately. Verification and validation of requirements, especially with unfavourable RCV values, will also be carefully managed to avoid cost and

time associated with re-verification and rework that can happen downstream of the design process.

The major contribution of this study is the exploration of both OOLR and BPANN using complex metrics as inputs. For each RCV class, the results ranked by BPANNs provide the computational results, which are statistically correlated with the network metrics. These findings provide a foundation for future investigations into predicting different RCV classes.

Although the performance of the OOLR models was relatively satisfactory, there is still a need for more validation to obtain more significant results due to the issue of data scarcity and imbalance was encountered as presented above. Although there exist data-analytic solutions to remedy this issue, the authors intend to address this limitation by conducting analyses with as many heterogeneous change propagation instances and requirements as possible for improving generalisability. Another limitation is the total number of possible combinations of algorithm parameters to explore in the computational models. Comprehensive search optimisation methods for the best parameters combined with more comprehensive types of computational models should be explored to obtain more robust models. Addressing these limitations in the future will allow for the development of improved computational models to assess RCV and avoid unanticipated propagating changes before committing resources.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This material is based upon work supported by the National Science Foundation [grant number 1463358].

References

Agresti, A. 2003. *Categorical Data Analysis*. Hoboken: Wiley.

ASCE. 2000. "Artificial Neural Networks in Hydrology. I: Preliminary Concepts." *Journal of Hydrologic Engineering* 5 (2): 115–123.

Barrat, A., M. Barthélemy, R. Pastor-Satorras, and A. Vespignani. 2004. "The Architecture of Complex Weighted Networks." *Proceedings of the National Academy of Sciences* 101 (11): 3747–3752.

Basheer, I. A., and M. Hajmeer. 2000. "Artificial Neural Networks: Fundamentals, Computing, Design, and Application." *Journal of Microbiological Methods* 43 (1): 3–31.

Bathaee, Y. 2018. "The Artificial Intelligence Black Box and the Failure of Intent and Causation." *Harvard Journal of Law and Technology* 31 (2): 889–938.

Bender, R., and U. Grouven. 1997. "Ordinal Logistic Regression in Medical Research." *Journal of Royal College of Physicians London* 31 (5): 546–551.

Bergmeir, C., and J. M. Benítez. 2012. "Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS." *Journal of Statistical Software* 46 (7): 1–26.

Bharti, K. K., and P. K. Singh. 2014. "A Survey on Filter Techniques for Feature Selection in Text Mining." *Advances in Intelligent Systems and Computing* 236: 1545–1559.

Bloechl, F., F. J. Theis, F. Vega-Redondo, and E. O. Fisher. 2010. "Which Sectors of a Modern Economy are Most Central?" *CESifo Working Paper Series*. 3175: 1–13.

Bock, C., and C. Galey. 2019. "Integrating Four-Dimensional Ontology and Systems Requirements Modelling." *Journal of Engineering Design* 30 (10–12): 477–522.

Bollobás, B. 2001. *Random Graphs*.

Bonacich, P., and P. Lloyd. 2001. "Eigenvector-Like Measures of Centrality for Asymmetric Relations." *Social Networks* 23 (3): 191–201.
Braha, D. 2016. "The Complexity of Design Networks: Structure and Dynamics." In *Experimental Design Research: Approaches, Perspectives, Applications*, edited by P. Cash, T. Stanković, and M. Štorga, 129–152. Cham, Switzerland: Springer.
Braha, D. 2020. "Patterns of Ties in Problem-Solving Networks and Their Dynamic Properties." *Scientific Reports* 10 (1): 18137.
Braha, D., and Y. Bar-Yam. 2004. "Topology of Large-Scale Engineering Problem-Solving Networks." *Physical Review E – Statistical Physics, Plasmas, Fluids and Related Interdisciplinary Topics* 69: 161131–161137.
Braha, D., and Y. Bar-Yam. 2007. "The Statistical Mechanics of Complex Product Development: Empirical and Analytical Results." *Management Science* 53 (7): 1127–1145.
Calcagno, V., and C. de Mazancourt. 2010. "Glmulti: An R Package for Easy Automated Model Selection With (Generalized) Linear Models." *Journal of Statistical Software* 34 (12): 1–29.
Chen, Y., P. Cheng, and J. Yin. 2010. "Change Propagation Analysis of Trustworthy Requirements Based on Dependency Relations." *2010 2nd IEEE Int. Conf. Inf. Manag. Eng.*, 246–251.
Chen, C., J. Mullis, and B. Morkos. 2021. "A Topic Modeling Approach to Study Design Requirements." *ASME 2021 Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*
Cheng, H., and X. Chu. 2012. "A Network-Based Assessment Approach for Change Impacts on Complex Product." *Journal of Intelligent Manufacturing* 23 (4): 1419–1431.
Clarkson, P. J., C. Simons, and C. Eckert. 2004. "Predicting Change Propagation in Complex Design." *Journal of Mechanical Design* 126 (5): 788–797.
Costa, L. D. F., F. A. Rodrigues, G. Travieso, and P. R. V. Boas. 2007. "Characterization of Complex Networks: A Survey of Measurements." *Advances in Physics* 56 (1): 167–242.
Dankelmann, P., W. Goddard, M. A. Henning, and H. C. Swart. 1999. "Generalized Eccentricity, Radius, and Diameter in Graphs." *Networks* 34 (4): 312–319.
DelSpina, B., S. Gilliam, J. Summers, and B. Morkos. 2018. "Corporate Requirement Culture in Development of a Large Scale Medical System: A Case Study." *Proceedings of the DESIGN 2018 15th International Design Conference*, 2621–2632.
Dong, C., Y. Yang, Q. Chen, and Z. Wu. 2022. "A Complex Network-Based Response Method for Changes in Customer Requirements for Design Processes of Complex Mechanical Products." *Expert Systems with Applications* 199: 117124.
Duran-Novoa, R., J. D. Weigl, M. Henz, and E. C. Y. Koh. 2018. "Designing in Young Organisations: Engineering Change Propagation in a University Design Project." *Research in Engineering Design* 29 (4): 489–506.
Eckert, C., P. J. Clarkson, and W. Zanker. 2004. "Change and Customisation in Complex Engineering Domains." *Research in Engineering Design* 15 (1): 1–21.
Estrada, E., and J. A. Rodriguez-Velazquez. 2005. "Subgraph Centrality in Complex Networks." *Physical Review E* 71 (5): 056103.
Fei, G. 2011. *A Methodology for Engineering Design Analysis Using System Modelling and Knowledge Management Technologies*. London, UK: University of Greenwich.
Giffin, M., O. de Weck, G. Bounova, R. Keller, C. Eckert, and P. J. Clarkson. 2009. "Change Propagation Analysis in Complex Technical Systems." *Journal of Mechanical Design* 131 (August): 1–14.
Gräßler, I., C. Oleff, and D. Preuß. 2022. "Proactive Management of Requirement Changes in the Development of Complex Technical Systems." *Applied Sciences* 12: 4.
Grover, A., and J. Leskovec. 2016. "Node2vec: Scalable Feature Learning for Networks." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
Guodong, Y., Y. Yu, Z. Xuefeng, and L. Chi. 2017. "Network-Based Analysis of Requirement Change in Customized Complex Product Development." *International Journal of Information Technology & Decision Making* 16 (04): 1125–1149.
Haibing, R., L. Ting, L. Yupeng, and H. Jie. 2021. "Multi-Source Design Change Propagation Path Optimisation Based on the Multi-View Complex Network Model." *Journal of Engineering Design* 32 (1): 28–60.

Hamraz, B., N. H. M. Caldwell, D. C. Wynn, and P. J. Clarkson. 2013. "Requirements-Based Development of an Improved Engineering Change Management Method." *Journal of Engineering Design* 24 (11): 765–793.

Hein, P. H., E. Kames, C. Chen, and B. Morkos. 2021. "Employing Machine Learning Techniques to Assess Requirement Change Volatility." *Research in Engineering Design* 32 (2): 245–269.

Hein, P. H., B. Morkos, and C. Sen. 2017. "Utilizing Node Interference Method and Complex Network Centrality Metrics to Explore Requirement Change Propagation." *Volume 1: 37th Computers and Information in Engineering Conference*, V001T02A081.

Hein, P. H., V. Okafor, V. Menon, and B. W. Morkos. 2015. "Exploring Requirement Change Propagation through the Physical and Functional Domain." *ASME 2015 International Design Engineering Technical Conference*, ASME, Buffalo, NY, DETC2015-47746.

Hein, P. H., N. Voris, and B. Morkos. 2018. "Predicting Requirement Change Propagation Through Investigation of Physical and Functional Domains." *Research in Engineering Design* 29 (2): 309–328.

Honey, C. J., R. Kotter, M. Breakspear, and O. Sporns. 2007. "Network Structure of Cerebral Cortex Shapes Functional Connectivity on Multiple Time Scales." *Proceedings of the National Academy of Sciences* 104 (24): 10240–10245.

Hosmer, David W., and S. Lemeshow. 2000. *Applied Logistic Regression*. New York: John Wiley & Sons.

Hwang, W., Y.-R. Cho, A. Zhang, and M. Ramanathan. 2006. "Bridging Centrality: Identifying Bridging Nodes in Scale-Free Networks." *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 20–23.

Ibrahim, N., W. M. N. W. Kadir, and S. Deris. 2011. "An Experimental Design Method for Evaluating Usability Factors of the Rechap Process Model." *International Journal of Innovative Computing* 1: 1.

Joshi, S., B. Morkos, and J. D. D. Summers. 2018. "Mapping Problem and Requirements to Final Solution: A Document Analysis of Capstone Design Projects." *International Journal of Mechanical Engineering Education* 47 (4): 338–370.

Koh, E. C. Y. 2017. "A Study on the Requirements to Support the Accurate Prediction of Engineering Change Propagation." *Systems Engineering* 20 (2): 147–157.

Konganti, K., G. Wang, E. Yang, and J. J. Cai. 2013. "SBEToolbox: A Matlab Toolbox for Biological Network Analysis." *Evolutionary Bioinformatics* 9: 355–362.

Kumbhar, P., and M. P. Mali. 2016. "A Survey on Feature Selection Techniques and Classification Algorithms for Efficient Text Classification." *International Journal of Science and Research* 5 (5): 1267–1275.

Latora, V., and M. Marchiori. 2001. "Efficient Behavior of Small-World Networks." *Physical Review Letters* 87 (19): 198701.

Latora, V., and M. Marchiori. 2007. "A Measure of Centrality Based on Network Efficiency." *New Journal of Physics* 9 (6): 188.

Ledoux, Y., D. Teissandier, and P. Sebastian. 2016. "Global Optimisation of Functional Requirements and Tolerance Allocations Based on Designer Preference Modelling." *Journal of Engineering Design* 27 (9): 591–612.

Li, Y., Y. Ni, N. Zhang, and Z. Liu. 2021. "Modularization for the Complex Product Considering the Design Change Requirements." *Research in Engineering Design* 32 (4): 507–522.

Li, Y., Y. Ni, N. Zhang, Q. Liu, and J. Cao. 2022a. "Towards the Evolution Characteristics of Product Structural Properties Based on the Time-Dependent Network." *Journal of Engineering Design* 33 (3): 207–233.

Li, Y., W. Zhao, W. Zhang, and M. Chen. 2022b. "A Dynamic Model for Engineering Change Propagations in Multiple Product Development Stages." *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 36: e13.

Long, D., and S. Ferguson. 2020. "Assessing Lifecycle Value Using Object-Based Modeling by Incorporating Excess and Changeability." *Journal of Mechanical Design* 143: 5.

Long, D., S. M. Ferguson, and B. Morkos. 2020. "Forecasting the Value of Excess in Personal Gaming Computers." *ASME International Design Engineering Technical Conference*.

Long, D., B. Morkos, and S. Ferguson. 2021. "Toward Quantifiable Evidence of Excess' Value Using Personal Gaming Desktops." *Journal of Mechanical Design* 143: 3.

Ma, S., Z. Jiang, and W. Liu. 2016. "A Design Change Analysis Model as a Change Impact Analysis Basis for Semantic Design Change Management." *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 231 (13): 2384–2397.

Ma, S., Z. Jiang, and W. Liu. 2017a. "Multi-Variation Propagation Prediction Based on Multi-Agent System for Complex Mechanical Product Design." *Concurrent Engineering* 25 (4): 316–330.

Ma, S., Z. Jiang, W. Liu, and C. Huang. 2017b. "Design Property Network-Based Change Propagation Prediction Approach for Mechanical Product Development." *Chinese Journal of Mechanical Engineering* 30 (3): 676–688.

Menon, V. 2015. *Application of Complex Network Metrics to Support Computational Reasoning of Requirement Change Propagation in Complex System Design*. Melbourne: Florida Institute of Technology.

Mollajan, A., and S. H. Iranmanesh. 2021. "Modularisation of System Architecture to Improve System Recoverability: A Unique Application of Design Structure Matrix." *Journal of Engineering Design* 32 (12): 703–750.

Morkos, B. 2012. *Computational Representation and Reasoning Support for Requirements Change Management in Complex System Design*. Clemson, SC: Clemson University.

Morkos, B., S. Joshi, and J. D. Summers. 2019. "Investigating the Impact of Requirements Elicitation and Evolution on Course Performance in a Pre-Capstone Design Course." *Journal of Engineering Design* 30 (4–5): 155–179.

Morkos, B., J. Mathieson, and J. J. D. Summers. 2014. "Comparative Analysis of Requirements Change Prediction Models: Manual, Linguistic, and Neural Network." *Research in Engineering Design* 25 (2): 139–156.

Morkos, B., P. Shankar, and J. D. Summers. 2012. "Predicting Requirement Change Propagation, Using Higher Order Design Structure Matrices: An Industry Case Study." *Journal of Engineering Design* 23: 12.

Newman, M. E. J. 2005. "A Measure of Betweenness Centrality Based on Random Walks." *Social Networks* 27 (1): 39–54.

Newman, M. 2010. *Networks: An Introduction*.

Olden, J. D., M. K. Joy, and R. G. Death. 2004. "An Accurate Comparison of Methods for Quantifying Variable Importance in Artificial Neural Networks Using Simulated Data." *Ecological Modelling* 178 (3–4): 389–397.

Orouskhani, M., D. Shi, and Y. Orouskhani. 2021. "Multi-Objective Evolutionary Clustering with Complex Networks." *Expert Systems with Applications* 165: 113916.

Paparistodimou, G., A. Duffy, R. I. Whitfield, P. Knight, and M. Robb. 2020. "A Network Science-Based Assessment Methodology for Robust Modular System Architectures During Early Conceptual Design." *Journal of Engineering Design* 31 (4): 179–218.

Pedersen, S. N., M. E. Christensen, and T. J. Howard. 2016. "Robust Design Requirements Specification: A Quantitative Method for Requirements Development Using Quality Loss Functions." *Journal of Engineering Design* 27 (8): 544–567.

PennState Eberly College of Science. Department of Statistics Online Programs. STAT 504 Analysis of Discrete Data.

Rubinov, M., and O. Sporns. 2010. "Complex Network Measures of Brain Connectivity: Uses and Interpretations." *Neuroimage* 52 (3): 1059–1069.

Rucker, G., and C. Rucker. 2000. "Walk Counts, Labyrinthicity, and Complexity of Acyclic and Cyclic Graphs and Molecules." *Journal of Chemical Information and Computer Sciences* 40 (1): 99–106.

Salkind, D. N. J. J. 2006. *Encyclopedia of Measurement and Statistics*. Thousand Oaks, CA: Sage.

Shabi, J., Y. Reich, R. Robinzon, and T. Mirer. 2021. "A Decision Support Model to Manage Overspecification in System Development Projects." *Journal of Engineering Design* 32 (7): 323–345.

Shafqat, A., J. Oehmen, and T. Welo. 2022. "Planning Unplanned Design Iterations Using Risk Management and Learning Strategies." *Journal of Engineering Design* 33 (2): 120–143.

Sheela, K. G., and S. N. Deepa. 2013. "Review on Methods to Fix Number of Hidden Neurons in Neural Networks."

Sreenivas Padala, S. P., and J. Uma Maheswari. 2022. "Modeling a Construction Project in a Matrix-Based Framework for Managing Requirement Changes." *International Journal of Construction Management*. Advance online publication. doi:10.1080/15623599.2022.2059739.

Steyerberg, E. W., B. Van Calster, and M. J. Pencina. 2011. "Performance Measures for Prediction Models and Markers: Evaluation of Predictions and Classifications." *Revista Española de Cardiología* 64 (9): 788–794.

Stirgwolt, B. W., T. A. Mazzuchi, and S. Sarkani. 2022. "A Model-Based Systems Engineering Approach for Developing Modular System Architectures." *Journal of Engineering Design* 33 (2): 95–119.

Suh, E. S., O. L. de Weck, and D. Chang. 2007. "Flexible Product Platforms: Framework and Case Study." *Research in Engineering Design* 18 (2): 67–89.

Swingler, K. 1996. *Applying Neural Networks: A Practical Guide*. London: Morgan Kaufmann.

Ullah, I., D. Tang, L. Yin, I. Hussain, and Q. Wang. 2017. "Cost-Effective Propagation Paths for Multiple Change Requirements in the Product Design." *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 232 (9): 1572–1585.

Vihinen, M. 2012. "How to Evaluate Performance of Prediction Methods? Measures and Their Interpretation in Variation Effect Analysis." *BMC Genomics* 13 (Suppl. 4): S2.

Wang, R., R. Huang, and B. Qu. 2014. "Network-Based Analysis of Software Change Propagation." *Science World Journal* 2014: 1–10.

Wang, S., Z. Li, C. He, D. Liu, and G. Zou. 2022. "Core Components-Oriented Modularisation Methodology for Complex Products." *Journal of Engineering Design*, 1–25.

Wang, G., L. Wu, Y. Liu, and X. Ye. 2021. "A Decision-Making Method for Complex System Design in a Heterogeneous Language Information Environment." *Journal of Engineering Design* 32 (6): 271–299.

Wen, L., D. Tuffley, and R. G. Dromey. 2014. "Formalizing the Transition from Requirements' Change to Design Change Using an Evolutionary Traceability Model." *Innovations in Systems and Software Engineering* 10: 181–202.

Wichmann, R. L., K. Gericke, and B. Eisenbart. 2021. "Analysing Risk to Function Fulfilment: Applying the Function Integrity Diagnosis and Documentation Method." *Journal of Engineering Design* 32 (7): 346–373.

Yadav, D., D. Long, B. Morkos, and S. Ferguson. 2019. "Estimating the Value of Excess: A Case Study of Gaming Computers, Consoles and the Video Game Industry." *ASME International Design Engineering Technical Conference*, Anaheim, CA.

Yin, L., Q. Sun, D. Tang, Y. Xu, and L. Shao. 2022. "Requirement-Driven Engineering Change Management in Product Design." *Computers & Industrial Engineering* 168: 108053.

Zheng, Y., Y. Yang, J. Su, N. Zhang, and Y. Jiao. 2017. "Dynamic Optimization Method for Configuration Change in Complex Product Design." *The International Journal of Advanced Manufacturing Technology* 92 (9): 4323–4336.

Zhong, L., Y. Yang, L. Shu, P. Jiang, and H. Wei. 2022. "A Surrogate Model-Assisted Robustness-Oriented Tolerance Design Method Based on 'Reverse Model." *Journal of Engineering Design* 33 (7): 491–516.