Explainability and Uncertainty Quantification in Networks and Social Systems

Sophie Sadler

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Doctor of Philosophy

Department of Computer Science
Swansea University

August 15, 2024

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed [redacted]........................... (candidate)

Date 15\08\2024

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed [redacted]........................... (candidate)

Date 15\08\2024

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed [redacted]........................... (candidate)

Date 15\08\2024

Statement 3

The University's ethical procedures have been followed and, where appropriate, that ethical approval has been granted.

Signed [redacted]........................... (candidate)

Date 15\08\2024

Abstract

The complexity of deep learning models has motivated the development of explainability approaches within the field of artificial intelligence. However, there are several adjacent fields to deep learning where similarly complex models are used to make decisions, which can also benefit from improved interpretability. In this thesis, we therefore focus on the application of existing explainability approaches, including the use of visualisation, to problems outside the traditional scope of deep learning. In particular, our focus is on the fields of social network analysis and optimisation. In addition to the use of explainability approaches, we also explore how uncertainty quantification can be used to improve the trustworthiness of decision-making within social network applications.

In the first two chapters of this thesis, we propose a methodology to apply feature importance scoring to the community detection problem in network analysis, where common approaches typically provide outputs with little explanation. We propose a longlist of features on several levels (individual nodes, pairs of nodes, and sets of nodes) which we believe are interpretable to network analysis experts, and explore which of these can be used to understand the outputs of the algorithms.

We then apply existing uncertainty quantification approaches to a new prediction problem which arises in large online social networks, where we analyse how these approaches perform in the face of the unusual data distributions that we see in this setting. In particular, we are interested in the engagement that online content receives.

Finally, we propose a novel visualisation approach to aid understanding in fitness landscape analysis. We perform dimensionality reduction on the locations of points in the landscape, including the optima, before representing these with a network structure which encodes additional information about the landscape. This chapter focuses on optimisation as another domain beyond network analysis which can benefit from explainability.

Acknowledgements

The work of this thesis would not have been possible without the support, encouragement and hard work of various other people, to whom I am extremely grateful. I would first like to thank my supervisor, Prof. Daniel Archambault, for the extensive help he has supplied me, from helping me develop as a researcher, to providing challenging discussion on various relevant topics, and facilitating collaboration with others who assisted me in similar ways. These people include, though are not limited to: my second supervisor Dr Mike Edwards, as well as Dr Derek Greene, Dr Alma Rahat, and Dr David Walker.

My PhD experience would not have been the same if I were not a part of the AIMLAC (Artificial Intelligence, Machine Learning and Advanced Computing) CDT. The staff of the CDT are responsible for creating an environment where interdisciplinary collaboration flourishes, knowledge is shared, and friendship and fun thrive. My particular thanks to Gert, Roz, Rhian and Biagio for making my time as a doctoral student so special.

I was also extraordinarily lucky to be welcomed by the Core Data Science team at Meta for a year during my PhD, where I learned from the absolute best. As well as contributing their support and ideas to my work, one chapter of this thesis is dependent on the data and infrastructure I was provided while working with the team. I am eternally grateful to Dima and Thomas for mentoring me during this time, as well as to Daniel, Ami, and Milan for working closely with me on my research, and to everyone else I met during this experience.

Finally, I would like to thank the friends and fellow doctoral students I have met during my time studying in Swansea, in particular Alex for his unconditional love and support, and the friends I have made in the past, in particular Masha and Annabel for acting as sounding boards for every decision I make in my life, whether large or small. The biggest thanks of all goes to my parents and my sister, for everything. This thesis is dedicated to the three of them.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The field of artificial intelligence (AI) is advancing at an increasing pace, with far-reaching consequences for almost every aspect of life and society as we know it. Indeed, the pace of development is unparalleled by that of other scientific fields, due to the relative speed of computers compared to biological or chemical processes, and due to the incorporation of automated procedures in AI in particular. Though this technological growth comes with great potential to improve human life in enumerable ways, it also comes with an array of concerning challenges, over both near and long-term horizons. These challenges include concerns about privacy, bias, inequality, safety and security, almost all of which are concerns arising due to the *black-box* nature of the models used [1]. For this reason, explainable AI is a niche within the field with growing importance, which seeks ways to implement intelligent, automated processes while retaining a good understanding of how they arrive at their solutions.

Existing literature on explainability for machine learning models is well-established, focusing both on the development of new model types with greater inherent transparency, and on the generation of explanations as applied to existing models, which have already been trained. These approaches which involve analysis of trained models are known as *post-hoc* explanations. Such approaches can, in many cases, be used to understand outputs of models that have already been deployed and are often *model-agnostic*, meaning they can be applied to models of many types. In this thesis, we focus on post-hoc explanations, and in particular, on *feature importance* and *visualisation* as explainability approaches.

Common factors impacting a machine learning model's ability to be understood are those relating to their input features; in particular, the number of these and their individual interpretability. For example, the number of friends a user has is a feature with inherent interpretability for engineers of a social media platform, while an individual pixel in an image lacks this quality. Some of the most well-known methods for the generation of post-hoc explanations therefore involve an estimation of feature importance, where each of the input features to a machine learning model are ranked based on a metric evaluating their contribution to a specific model output [99]. In particular, LIME (Local Interpretable Model-agnostic Explanations) [135] focuses on the importance of interpretable features. By contrast, visualisation is an extremely large field with applications to many elements of data science, machine learning and AI, of which we will explore only those which are most relevant to explainability for network problems, the focus of this thesis.

Notably, much of the existing literature on explainability has emerged from the machine learning community, though there are many complex, stochastic algorithms outside the realm of machine learning which may still be considered black-box AI approaches. Any domain reliant on such algorithms has the potential to encounter similar challenges around bias, safety and so on, and thus it follows that explainability approaches can provide benefit in these settings too [9]. In this thesis, we therefore explore the application of explainability to fields outside of machine learning where there is less pre-existing literature. Our focus is on the fields of *network analysis* and *optimisation*, and, more specifically, on problems within *community detection* and *fitness landscape analysis*. Unlike machine learning models, the algorithms we consider in these fields do not depend on a pre-defined list of input features, and we therefore develop our own methodologies for identifying features that can be included in post-hoc explanations and ranked for their ability to contextualise the outputs of the algorithm.

Uncertainty quantification itself is not usually considered to be an explainability technique, as the approaches used to quantify uncertainty can themselves be “black-box” models. However, it shares common goals with the field of explainability in improving the trustworthiness and robustness of decision-making using AI. Additionally, uncertainty quantification can be a powerful tool when used in conjunction with explainability. In general, the aim of using post-hoc explanations is to understand how an algorithm reaches a conclusion, at which point a human is then usually required to assess the logic or the process which has been illuminated by the explainability approach. This methodology therefore requires that the human either assesses all, or a large number, of the data points; or that some heuristic is used to determine which data points should be prioritised for explanation. One such metric which can be used to identify specific data points for which the model has been more likely to under-perform is the model’s degree of *uncertainty* about its prediction. This demonstrates the utility of combining approaches from the two fields when assessing model performance and generating insight.

However, most models and algorithms in AI do not provide an estimation of uncertainty by default, making uncertainty quantification a substantial area of research of its own. Uncertainty in machine learning is often classified into two types, depending on its source: epistemic uncertainty (which arises due to errors from the model) and aleatoric uncertainty (which arises due to variation in the data) [61]. The former can be caused by a model which is insufficient to represent the underlying distribution; a faulty training procedure; or a dataset used for training which is not representative of the distribution it is sampled from (for example, a large area of the feature space is not present in the training set). Data-dependent errors are caused by the information loss which occurs when real phenomena are represented with a dataset (for example, it may be possible with the features present in the dataset for a particular instance to be correctly classified in more than one way, as the chosen features are insufficient to determine which class it truly belongs to.) In this thesis, we extend our exploration of explainability in AI as applied to network analysis and optimisation to the inclusion of an exploration of uncertainty quantification to a machine learning problem in social networks.

As we employ the use of network structure in visual explanations for optimisation, all of the work herein therefore pertains in some way to the theory of networks. A network as defined in this work represents relational data, where nodes correspond to entities in the dataset (such as individuals in a social network) and edges correspond to relationships between these entities (such as friendships or academic collaborations). Most Explainable AI techniques have focused on algorithms which are typically applied to tabular or image data, though there has been some foundational works on explainability for graph neural networks [59, 100, 171, 183, 184, 186]. Nevertheless, explainability for problems on network data is relatively under-explored.

These observations bring us to the central question underpinning the research of this thesis:

> *How can we apply or adapt existing techniques from explainable AI and uncertainty quantification to new problems in network applications and social systems?*

We decompose this wider question into a smaller number of specific application-driven research questions, which motivate the individual contributions we have made:

- Which interpretable features can be identified for use in post-hoc explanations for community finding algorithms, in the network analysis setting?
- How can we quantify uncertainty for a machine learning approach which predicts the popularity of media content posted by users in a social network?

- How can we visualise explanatory information which aids the use of optimisation algorithms in fitness landscape analysis?

Informed by these research questions, our work makes the following novel contributions:

- The development of a methodology for identifying the most informative features for community detection algorithms, which may generalise to other problems in social network analysis.
- The application of this methodology to 3 community detection algorithms, and therefore the identification of node, node-pair and community features which may be relevant for visual analysis of community structure in networks.
- A study evaluating the performance of uncertainty quantification approaches in the face of a highly-skewed distribution type found in social networks, where uncertainty quantification has not previously been applied.
- The development of a technique for improving the performance of MC Dropout on this dataset.
- The development of a novel visualisation technique (the extrema graph) to capture the general characteristics of function landscapes for fitness landscape analysis.
- An analysis of the insight gained from this extrema graph approach on six benchmark problems.

The remainder of the thesis is structured as follows. Chapter 2 details the related work for explainable AI as applied to network analysis and graph problems, as well as relevant visualisation work in fitness landscape analysis and uncertainty quantification. Additionally, some preliminary background on networks is provided in this chapter.

Chapter 3 presents the results of experiments to identify interpretable features for use in post-hoc explanations for community finding algorithms. We select a longlist of features ourselves, and present a methodology to identify the most predictive, which we apply to three community detection algorithms. This chapter is based on work published in the Springer journal Applied Network Science [140].

We then apply this methodology in a similar manner in Chapter 4 to identify interpretable features for community detection algorithms on a different scale. In the original experiments, we defined interpretable features on nodes and pairs of nodes, while here we adapt our methodology to define interpretable features on larger sets of nodes. This chapter is based on work presented at the Springer conference Complex Networks & their Applications (CNA) 2021 [139].

Chapter 5 presents an analysis of the performance of various uncertainty quantification approaches as applied to a machine learning problem in social media and networking. In particular, the problem in question is to predict the popularity of content posted on Facebook and Instagram. This work was completed in collaboration with Meta, during an internship and subsequent contractor role at the organisation.

In Chapter 6, we display our novel visualisation approach for visualising fitness landscapes, based on extrema graphs. We developed this in order to capture characteristics of function landscapes with input dimensions of three or higher, similarly to contour plots for two-dimensional input spaces. The work in this chapter is based on that presented at the LAHS workshop at ACM's Genetic and Evolutionary Computation Conference (GECCO) 2023 [141].

We examine the work presented in the previous chapters for limitations, position it within existing work, and make recommendations for future research based upon it in Chapter 7. Concluding remarks are then made in Chapter 8.

Chapter 2

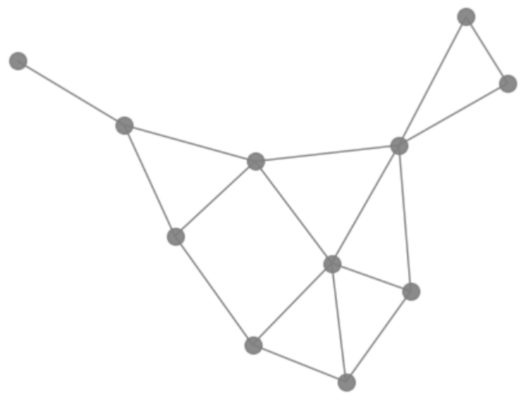
Background and Related Work

In this chapter, we place the content of this thesis within the context of existing concepts and related literature in relevant fields of research. As our focus is on developing approaches for explainability in problems within social networks and optimisation, in addition to the application of uncertainty quantification to the first of these, the relevant fields of research include explainable AI, social network analysis, fitness landscape analysis and uncertainty quantification, as well as the use of visualisation for explainability. In particular, we explore existing approaches for explainability, within our domains of interest as well as others with interesting similarities, and relevant background information on the problems we explore within the fields of social network analysis and optimisation themselves.

In doing so, we have three aims. Firstly, in section 2.1, we introduce the concepts and ideas which are prerequisites for understanding the work of the thesis. Secondly, in section 2.2, we highlight and summarise that existing literature which closely relates to ours, in order to establish the state of the art and to identify areas of research which have already been explored. Finally, in section 2.3, we discuss our own design choices, to clarify how our work differs from that which has come before, and why we have chosen to take either similar or different approaches to previous bodies of work.

2.1 Background

We begin by introducing definitions and approaches upon which the work of this thesis is dependent, or which are used without modification for our own experiments. These include definitions from social network analysis which describe the data and types of problems we explore; the community finding algorithms for which we develop an explainability approach, and the benchmark generator with which we generate the data in Chapters 3 and 4; and existing explainabiltiy approaches which we use without modification (permutation importance scoring and Shapley values).



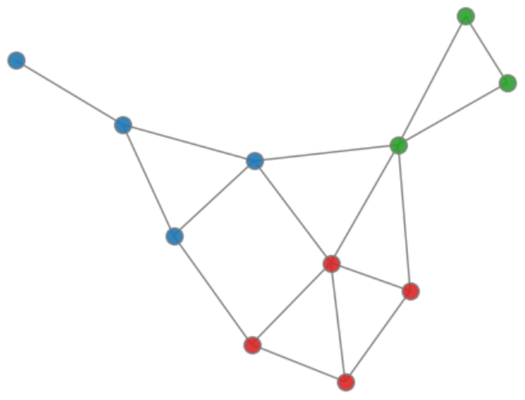


Figure 2.1: On the left is a graph with 11 nodes and 17 edges. On the right is the same graph with a possible community partitioning. Each node belongs to a single community, where its community is represented with colour. There are 3 communities in this graph.

2.1.1 Social Network Analysis Definitions

We first introduce definitions relating to the structure of networks, as our work primarily focuses on data which takes this format. A longer list of problem-specific definitions are introduced in later chapters (Sections 3.1.1, 3.1.2 and 4.1.3).

- **Graph** A graph, $G = (V,E)$ is a data structure which models pairwise relationships between objects. It is comprised of a set, V, of these objects known as *vertices* or *nodes*, and a set, E, of edges which connect two nodes within the set. Throughout this thesis, we use the terms *graph* and *network* interchangeably. An example of a graph is displayed in Figure 2.1A.
- **Degree** The degree of a node $i \in V$ is the number of edges adjacent to the node.
- **Community** Most generally, a community is a subset of nodes in a graph. Various more formal definitions of a community are used within the literature, however these commonly represent subsets of nodes which are more densely connected to eachother

than they are to other regions of the graph. In some cases, a node can only be assigned to a single community; these communities then form a *partition* of the network. In others, a node may be assigned to more than one community. However, for the experiments within this thesis, we focus only on partitioning communities. An example of a graph separated into partitioned communities is displayed in Figure 2.1B.

2.1.2 Community Finding Algorithms

Many real-world networks of interest display community structure. For example, consider the global network of users on a social media platform such as Facebook, who are connected as "friends" to people they know in the real world. We would expect to see clusters of more densely connected users for a variety of reasons, e.g. the staff and students of a school would share more common connections with eachother than to anyone outside of the school. Identifying these communities is of interest as it allows for greater understanding of both the network structure, and also the role that individual nodes play within the graph. However, this is a non-trivial problem and a variety of techniques have been proposed to detect these communities. As there is no universally agreed definition of a community, some of these approaches are designed with a specific research problem or dataset in mind. For the experiments in this thesis, we focus on a subset of community detection algorithms: the Louvain algorithm, the Infomap algorithm, and the label propagation algorithm (LPA). We introduce these here, and further discuss the reasoning behind our selection in Section 2.3.

Louvain The Louvain algorithm [13] performs optimisation on a measure known as modularity, which we denote Q, and is defined as follows for a weighted graph, $G = (V, E)$:

$$Q = \frac{1}{2m} \sum_{ij \in V} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

where m is the sum of all edge weights in the graph; $A_{ij} = 1$ if i and j are connected by an edge and 0 otherwise; k_i and k_j are the degrees of nodes i and j respectively; c_i and c_j are the community labels of nodes i and j respectively; and $\delta(c_i, c_j) = 1$ if these communities are the same, and 0 otherwise. Intuitively, this represents the mean difference between the number of edges we observe connecting nodes in a community and the expected number of edges within this community if they were distributed randomly. Therefore, a high modularity encapsulates our understanding of what makes a community: regions of the network where we see a higher density of connections.

There are two phases of the Louvain method which are then repeated iteratively. In the first phase, each node in the network is assigned to its own community. Then, sequentially, each node in the network is moved to the community of one of its neighbours which results in the greatest modularity increase. If no increase is possible, it remains in its original community. This is repeated until no modularity increase can occur by moving any node to a neighbouring community, at which time the first phase of the algorithm ends.

The second phases consists of reducing each community to a single node of a new network, where the edges of the new network are weighted according to the number of edges connecting the communities in the original network. The first phase is then repeated on this new network.

Infomap The Infomap algorithm [137] is similar to the Louvain algorithm, though instead optimises for an objective function known as the *map equation*. This makes use of an information-theoretic approach, where nodes are represented by codewords composed of two parts, the first of which is provided by the community it belongs to. The community memberships are optimised by minimising the average code length describing random walks on the network. A random walker will tend to stay longer in dense areas that equate to the communities we are attempting to find, and minimising the map equation corresponds to detecting the most modular structure possible.

The map equation depends on node visit rates. A node will be visited by the random walker with a frequency dependent on the proportion of edges which connect to that node. These rates are stored in a *codebook* per module (i.e. community), depending on which module the node belongs to. Module entry and exit rates (equivalent in undirected graphs, which we use in our experiments) then represent the frequency with which a random walker would enter or leave a set of nodes and are stored in a global *index codebook*. Shannon's source coding theorem [147] is used to convert these rates into information measured in bits. In particular, the amount of information needed to describe an average movement length of a random walker can be described using the entropy of the rates in the codebook, weighted by the use rate of each codebook (i.e. each community). The map equation is as follows:

$$L(M) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p_{\circlearrowright}^{i} H(\mathcal{P}^{i})$$

where M is a partition of the nodes into m modules; and $H(\mathcal{Q})$ and $H(\mathcal{P}^i)$ are the frequency-weighted average length of codewords in the index codebook and the frequency-weighted average length of codewords in module codebook i respectively. The other probabilities represented in the equation are defined as follows. With $q_{i\curvearrowright}$ being the probability of exiting module i, $q_{\curvearrowright} = \sum_{i=1}^{m} q_{i\curvearrowright}$, which is the probability that the random walker changes to a new module on any given step. With p_α being the probability of visiting node α, $p^i_{\circlearrowright} = \sum_{\alpha\in i} p_\alpha + q_{i\curvearrowright}$, which is the fraction of time the random walk spends in module i plus the probability that it exits the module, i.e. the rate at which the codebook for module i is used.

LPA The label propagation algorithm proposed by Raghavan et al. [130] makes the assumption that nodes should belong to the same community as most of their neighbours. Every node is initialised with its own community label, which are then propagated through the network. For this reason, densely connected regions reach a common label quickly, and these regions then expand until they meet other densely connected regions. The process occurs in 4 stages:

1. Nodes are initialised with a unique community label.
2. A random ordering of the nodes is determined.
3. Each node's community label is updated according to the most frequent among their neighbours, in the order determined in the previous step. If several community labels occur with the same frequency, the new community label is chosen at random from those which are most common.
4. If every node has the label that the majority of its neighbours has, the process ends. Otherwise the process repeats beginning with step 2.

2.1.3 LFR Benchmark

The Lancichinetti–Fortunato–Radicchi (LFR) benchmark algorithm generates synthetic networks with labelled communities, allowing comparison of different community finding networks. In our experiments of Chapters 3 and 4, we use a large number of these synthetic LFR networks to evaluate the different interpretable community metrics we have identified for possible use in explainability of the community detection algorithms' outputs. Although this benchmark generates synthetic networks, it aims to replicate the structure of real networks by accounting for heterogeneity of node degree and community size. An assumption is made that both the node degrees and the sizes of the communities (i.e. the number of nodes belonging to the communities) follow power law distributions, with exponents τ_1 and τ_2 respectively. Other inputs to the generator are n, the number of nodes; average degree, $< k >$; and the mixing parameter, μ. This mixing parameter introduces noise to the communities relative to its value. The algorithm then works as follows:

1. Generate a network such that the node distributions follow a power law distribution with exponent τ_1 and have average degree $< k >$

2. Generate a set of community sizes according to a power law distribution with exponent τ_1 until they sum to n.

3. In order to satisfy the condition set by the mixing parameter, each node i must have $(1-\mu)deg(i)$ connections within its community, and $\mu deg(u)$ outside its community. Initially, node's communities are not assigned. These are then chosen for each node in turn with the condition that the community is large enough for the node to have the required $(1-\mu)deg(i)$ connections within its community. If adding a node to a community would exceed its assigned size, a random node already in the community will be selected for reassignment to a new community, until all nodes have been assigned a community.

4. Edges between nodes are adjusted until all nodes have the correct proportion of connections within and outside its community.

For low values of μ, the communities remain well separated and thus easy to detect, but as this value increases, communities become harder to separate.

2.1.4 Feature Importance Scoring

In developing our methodology presented in Chapters 3 and 4, we make use of existing explainability approaches from the field of machine learning, with a particular focus on feature importance scoring. In machine learning problems, a model is typically trained on a dataset composed of a feature matrix, X, and a vector of labels, y. For example, this feature matrix may be information about posts on a social media platform such as Facebook, e.g. the number of friends the user has, the length of the post, the time of day it was posted, and so on. The vector of labels may then record how many views each post received, which we assume is dependent on the features recorded in the matrix. However, some features may be more predictive than others. Feature importance scoring is an area of explainability which aims to determine which features are most strongly contributing to the prediction of the trained model. This insight can improve model understanding by highlighting which input features the model is placing most emphasis on. For example, in a dataset owned by a social media platform of user data, it may indicate a poorly trained model or poorly cleaned dataset if the feature of greatest importance for predicting content popularity is the content ID. Several approaches generate importance scores for the different features in a prediction task, but for our experiments, we incorporate the use of two scoring methods in particular: SHAP, and permutation importance.

SHAP The work by Lundberg et al. [99] which proposed SHAP (SHapley Additive exPlanations) is based on the early game-theoretic work by Shapley [148]. In this original work by Shapley, each feature value in a datapoint is a "player" in a game where the prediction is viewed as the "payout". The Shapley value is then the contribution of a feature value to the payout, weighted and summed over all possible feature combinations:

$$\phi_j(val) = \sum_{S \subseteq \{1,2,...,p\} \setminus \{j\}} \frac{|S|!(p-|S|-1)!}{p!} (val(S \cup j) - val(S))$$

where S is a subset of features used in the model, of which there are p: $1,2,...,p$. The Shapley value is the only attribution method which satisfies the four properties which constitute the definition of a "fair payout": efficiency, symmetry, dummy, additivity, which in the context of machine learning represents a fair attribution of the prediction to the features. These properties are defined as follows:

- *Efficiency* All feature contributions (i.e. the Shapley values of each feature) must sum to the difference between the prediction and the average prediction of the model:

$$\sum_{j=1}^{p} \phi_j = \hat{f}(x) - \mathbb{E}_x[\hat{f}(X)]$$

  where $\hat{f}()$ represents the model and x is a given data instance.

- *Symmetry* If two features, j and k, contribute equally to all coalitions (i.e. all possible feature subsets) then they should have the same Shapley value:

$$\phi_j = \phi_k \text{ if } val(S \cup j) = val(S \cup k) \text{ for all } S \subseteq \{1,2,...,p\} \setminus \{j,k\}$$

- *Dummy* The Shapley value for a feature, j, which does not change the prediction for any coalition of features it is added to should be 0:

$$\phi_j = 0 \text{ if } val(S \cup j) = val(S) \text{ for all } S \subseteq \{1, 2, ..., p\}$$

- *Additivity* For a game with combined payouts, the Shapley value of a feature should be the sum of its Shapley values within the sub-games. This feature is particularly important for random forests, where the Shapley value of a feature in the random forest is the mean of its Shapley values in each of the decision trees.

Despite these beneficial theoretical guarantees, by the definition of the Shapley value, it requires consideration of every subset of features with or without the target feature included, to exclude feature interaction effects. Therefore, this approach requires retraining a model for every subset of input features, which is extremely computationally expensive, and infeasible for most modern machine learning problems. This motivated Lundberg et al. to develop their SHAP approach, which avoids the computational complexity associated with classical Shapley values using what they call *additive feature attributions*; in essence, this represents the Shapley values using a linear model. SHAP specifies explanations in the following way:

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

where g is the explanation model; $z'_j \in 0, 1^M$ is the coalition vector, which contains a 1 where a feature is present in the Shapley coalition and 0 if it's absent; M is the maximum coalition size; and ϕ_j is the Shapley value for a feature j. Using this representation as a linear model of the different coalition combinations is a trick to calculate the ϕs, where sample coalitions are used. Lundberg et al. developed two versions of SHAP: KernelSHAP and TreeSHAP, the latter of which is specifically designed for tree-based models such as random forests. SHAP approaches satisfy the four aforementioned properties of Shapley values, retaining the beneficial theoretical guarantees.

Permutation Importance This method was first introduced in the same paper as random forests [19] and is defined to be the change in test loss when a single feature value is shuffled throughout all data instances. It is concluded that features which result in the biggest decrease in model performance when shuffled are those which are *most important*, i.e. which are most greatly contributing to the model prediction. The full methodology is outlined in Algorithm 2.1.

Algorithm 2.1 Permutation Importance

```
Input: Fitted predictive model $\hat{f}$; feature matrix, $X$ (training or validation); target vector, $y$;
    loss function $L(y, \hat{f})$; number of repetitions, $K$
Output: Vector of importance scores $S$, where $S_j$ corresponds to feature $j \in \{1, ..., p\}$
1: Calculate the original loss, $l_{orig} = L(y, \hat{f}(X))$, e.g. mean-squared error
2: for feature $j \in \{1, ..., p\}$ do
3:     for repetition $k = 1$ to $K$ do
4:         Permute the column of $X$ corresponding to feature $j$ to receive permuted data, $X_{perm}$.
5:         Compute the new loss, $l_k = L(y, \hat{f}(X_{perm}))$
6:     end for
7:     Compute importance $S_j$ for feature $j$
           $$S_j = l_{orig} - \frac{1}{K}\sum_{k=1}^{K} l_k$$
8: end for
```

2.1.5 Statistical Methodology

In analysing our results of Chapters 3 and 4, we make use of a statistical methodology to ensure that the features we identify as most informative are truly distinct in their predictive power from the other features of our study, rather than merely appearing to be so due to chance. In other words, we analyse whether the identified greater importance of these features is *statistically significant*. The effect we are interested in is the *pairwise difference between means* of the individual distributions. To perform this analysis, we make use of several statistical tests and methods, which we outline here:

Power Analysis Power analysis is a statistical method to determine the sample size needed for a study to have a high probability of detecting a significant effect. Statistical power is composed of four elements: effect size (the quantified size of a result); significance (the level of statistical significance); statistical power (the probability of accepting the alternative hypothesis, that an effect is present); and sample size. Since we aim to determine the last of these, the others must be provided in order to perform the power analysis. Our use of power analysis assumes that the underlying feature importance distributions we are observing are normally distributed.

Shapiro-Wilk Test The Shapiro-Wilk test is used to determine whether a sample of data has come from a normal distribution. A significance level is set, e.g. 0.05, and if the observed p-value is less than this significance level we conclude that the underlying distribution is unlikely to be normally distributed.

Pairwise Comparisons of Means To compare the means of two distributions under an assumption that they are normally distributed, a pairwise t-test can be used. In the case that they are not normally distributed, a pairwise Wilcoxon test is used instead. This is a non-parametric alternative to the t-test.

Bonferroni-Holm Corrections When testing multiple hypotheses at once, as we do in our experiments, there is an increased probability of observing a false positive in at least one of the tests. Bonferroni-Holm corrections are one approach for controlling the family-wise error rate (FWER), by adjusting the rejection criterion, i.e. the significance level, for each test in turn. For example, if we have n hypotheses and a desired significance level of 0.05, then we would use a significance level of $\frac{0.05}{n}$ for our first hypothesis, $\frac{0.05}{n-1}$ for our second hypothesis, and so on.

2.2 Related Work

We now identify and summarise existing literature which relates closely to our work in this thesis. The majority of this literature falls under the field of Explainable AI (XAI), a term originally coined in the paper by Lent et al. [166] to describe the agents they developed for military simulations and computer games. Since then, many surveys have attempted to outline this field, or components of it [1, 22, 35, 142, 159]. Most of the existing literature focuses on explainability for machine learning, but researchers from other fields have also begun to understand the importance of applying explainable AI to their own research [9]. The focus of this thesis is explainability for problems in *network analysis* and *optimisation*.

In the terminology, *explainability* is sometimes exchanged for *interpretability* [4]. These terms are often used to mean the same thing, despite some previous attempts to distinguish independent meanings, such as in the paper by Rudin [138] where the term *interpretability* refers to models which can inherently be understood, and the term *explainability* refers to techniques designed to augment existing black-box models with additional insight. We ignore this distinction and avoid use of these terms to prevent any confusion, instead referring specifically to *model transparency* for a model's inherent ability to be understood and the term *post-hoc explanations* to describe those approaches which provide additional understanding for a black-box model. Our focus is on post-hoc explainability, thus we explore work in this area in more detail in the upcoming subsection of the related work.

The remainder of the related work in this chapter relates to the following topics: explainability for problems on network data; visualisation as a tool for explainability; context on the field of social network analysis; the state-of-the-art in community detection; context on the field of fitness landscape analysis, our focus within optimisation; and uncertainty quantification, in particular its intersection with and relation to explainability for AI.

2.2.1 Post-hoc Explainable AI

Post-hoc explanations typically provide some kind of rationale or evidence for the reasoning behind a "black-box" model's output [10, 42, 91, 157]. This field has arisen in the face of challenges developing well-performing models that are inherently transparent. Indeed, for certain types of data or for more complex algorithms, the use of a model which can inherently be understood might not be feasible or effective. In this section, we explore the prominent post-hoc explainability techniques before honing our focus to the application of these to network problems.

2.2.1.1 Feature Importance for Post-hoc Explainability

A recent survey by Molnar et al. [110] covers the history of interpretable machine learning as a field, and contains a useful classification of interpretability methods into those which assign meaning to model components; those which explore model sensitivity; and those which development interpretable surrogate models that mimic the behaviour of the model to be explained. Their use of the term "interpretable" to describe these approaches, which are largely post-hoc methods, evidences the use of the interchangeability of "interpretable" and "explainable" as terms in machine learning.

Component analysis, the first method, involves decomposing the machine learning model into smaller parts which can be understood individually. For easily interpretable models, such as decision trees, this may be a simple case of applying pruning to reduce the parts which need to be understood. However, it can also work for more complex models, such as CNNs. In this case, it may require finding or generating images which activate a feature map of the CNN. However, there remains the obvious disadvantage to the component analysis approach, which is that it depends entirely on the model and cannot be generalised to other ML approaches.

By contrast, the approaches which explore model sensitivity can be model-agnostic and work by manipulating input data and analysing the resulting outputs. One such approach is the permutation feature importance, as introduced in Section 2.1.4, though the calculation of feature importance scores in this way is a common approach in post-hoc explainability which extends to other techniques beyond permutation feature importance. In fact, two of the most well-known techniques in XAI overall are SHAP [99], also introduced in Section 2.1.4, and LIME [135], which are both approaches that calculate the importance of input features. The latter is an example of a surrogate model method in the Molnar classification of interpretability methods.

In order to explain a complex model, Ribeiro et al. [135] propose with LIME a method which generates a simpler model in the local area around a datapoint of interest, which is itself inherently transparent. This model is trained using perturbations on the target datapoint. Some element of the simple, local model can then be used as the importance score for the features, such as the weights. However, in order to generate useful explanations, the authors highlight the importance of using features which are themselves interpretable. For example, individual pixels in an image are not semantically meaningful, so for this type of input data they instead use so-called *super-pixels*. The importance of choosing input features that can themselves be understood by an end-user was a key consideration in the development of our experiments in chapters 3 and 4. The authors also introduced SP-LIME (submodular pick LIME), which differs in that it provides global explanations for the model as a whole, rather than for individual observations. Both SHAP and LIME are model-agnostic in the sense that they can be applied in conjunction with any classifier and do not require inspecting the internal workings of that classifier.

Beyond LIME, there are several other works in post-hoc explanation which are based on the use of surrogate models. Many of these focus on the distillation of neural networks into decision trees, which is considered by some to be an inherently interpretable model form [11, 46]. Other methods develop specific model-types for use as explanations [79,127]. In the work by Keane et al. [70], they propose a *twin-systems* strategy, where a complicated neural network model is mapped to a simpler, more interpretable “twin" model using case-based reasoning.

In some approaches, such as RuleMatrix [107], they have even used novel visual analytics approaches to display surrogate model information.In this case, their surrogate model identifies IF-THEN rules explaining the behaviour of the model, and displays these in a matrix.

For our experiments in this thesis which rely on existing explainability approaches, we are primarily focused on the identification of highly predictive, interpretable features. In particular, we are interested in predictive problems which relate to social network analysis. In the next subsection, we will explore other techniques which have been proposed for explainability on network problems.

2.2.1.2 Explainability for Network Problems

In this thesis, we aim to develop explainability approaches for problems which incorporate the use of networks, and, in particular, for those outside deep learning. In fact, the field of deep learning on networks is itself relatively young, and very little explainability literature exists for graph neural networks (GNNs), the dominant approach in this area. A survey by Yuan et al. [185] explored those techniques which do exist for GNNs, and the challenges which have hindered significant progress in this field. Some of the reasons that proposed in the survey for the difficulty in developing explainability approaches for deep learning on networks include: the common representation of networks with discrete adjacency matrices; the possible lack of semantic meaning for individual nodes in a graph making them unsuitable to use for explanations; the fact that many networks are inherently more challenging to understand in some cases than images, text or tabular data. Many network problems require extensive domain knowledge to be understood, for example biochemical expertise for problems where molecules are represented as networks. Many of these obstacles, such as the possible lack of semantic meaning for individual nodes or the necessity of domain knowledge, extend to the problems we address in social network analysis, which make use of algorithmic approaches other than deep learning. We can therefore learn from the literature on deep learning where they tackle explainability in the face of these challenges.

In the aforementioned survey, these approaches are distinguished with two main classes: instance-level methods, which generate explanations for individual datapoints, and model-level methods, which aim to provide a more general explanation for the behaviour of the whole model. In some cases, “instance-level” may refer to nodes or edges, for example, when node or edge classification is the problem of interest. In other cases, it may refer to whole networks, for example in graph classification. This distinction between explanations for individual datapoints, and for the model as a whole, has previously been seen in XAI for other data types. For example, the distinction between the two is made in the LIME paper [135] where they develop a separate approach, SP-LIME for model-level explanations. However, this highlights the early stage of the GNN explainability field, as the authors are only able to identify one approach which generates model-level explanations. This technique, XGNN [184], proposes to explain GNNs through a graph generation process. A separate generator is trained in order to maximise a target graph prediction, thereby explaining in some way the original model's expectation for what such a graph should look like. The original paper focuses only on graph classification, leaving model-level explanations for node classification an open area [185].

All other approaches focus on instance-level explanation. The majority of these employ a perturbation-based approach, where the effect on outputs is observed when perturbing the input [100, 183, 186]. Each of these has a very similar high-level pipeline, which makes use of a *mask* on the nodes, edges, or node features, depending on the specific approach. This mask takes the form of a matrix, where each position in the matrix represents a specific edge (for example), and the value at that position represents the importance of that edge in the problem of interest, similarly to how feature importance is used for other data types. A mask generation process is optimised in order to achieve this. The optimisation is done by applying the mask to the input graph, and iteratively adapting the mask generation process such that the new input graph has a similar prediction to the original one. As well as the type of mask, the mask generation process and the optimisation function also vary across the different approaches.

Beyond this common perturbation-based approach, some GNN explanation methods have also been developed in slightly different ways. For example, LIME has been adapted for the graph setting into GraphLIME [59]. As with the original LIME paper, this uses a simpler, surrogate model to replicate the behaviour of the original GNN, differing it from the perturbation-based approaches. The surrogate model is designed to be easier to understand than the model of interest, while mimicking its behaviour as closely as possible, at least in a local region. Another notable GNN explanation tool is PGM-Explainer [171] which also uses a surrogate-model based approach, this time using a probabilistic graphical model.

One of the aims of our work is to identify features that can be used in explanations for network problems. Our shortlist of features is restricted to those which are inherently interpretable to experts in social network analysis. Therefore, our approach does not depend on the use of GNNs, which are themselves complex and require explanation, thus this existing work on explainability for GNNs does not directly relate to the work of this thesis. Nevertheless, it does provide the beginnings of research on developing explainability for problems focused on network data, which is our focus. The early stage of this literature highlights the difficulties in developing explainable approaches for this type of data in contrast to images, text or tabular data. Nevertheless, we can learn from their attempts to circumvent the associated challenges. The necessary considerations in visualising this type of data are also relevant for our work on developing a novel visualisation technique for fitness landscape analysis, where we use a graphical representation to our advantage. In the next section, we will explore the use of visualisation for explainability in greater detail.

2.2.2 Visualisation as a Tool for Explainability

We now turn our attention to the existing literature on visualisation as a tool for explainability, both for network problems and artificial intelligence more generally. One comprehensive survey of the use of visual analytics in deep learning was presented by Hohman et al. [58]. This survey focuses on advancements relating to deep learning only, leaving out simpler classifiers and regression models. It also explores visual analytics used for reasons beyond explainability, but a large proportion of the papers mentioned are nevertheless relevant to XAI. They classify visual analytics papers under several headings: why the data is being visualised (intepretability, debugging, comparing models, teaching); what is being visualised (network architecture, learned parameters, etc.); when the visualisation occurs (during or after training); who the visualisation benefits (model developers, model users, or non-experts) and how the visualisation works (dimensionality reduction, node-link diagrams, etc.). For the work proposed here, some of these categorisations provide useful questions. For example, the question of what information a visualisation will show is highly relevant for our work developing a novel visualisation for fitness landscape analysis, as our method was designed to complement existing visualisation techniques. We explore the existing literature further in section 2.2.4. The profile of the end-user was also a key consideration in our development of interpretable features for explainable community finding. Other surveys have further explored the existing literature on visualisation for explainability [95, 97, 109].

2.2.2.1 Visualising Deep Learning

In this thesis, we prioritise the application of visualisation to optimisation problems which fall outside of deep learning, in particular, fitness landscape analysis within optimisation. Nevertheless, it is beneficial to review the literature on explainability for deep learning, as we can draw inspiration from existing approaches when developing ours for new domains.

Supervised Learning There is a lot of existing literature which aims to provide insight into the workings of convolutional neural networks (CNNs) using visual analytics. A survey by Qin et al. [128] summarises the main techniques used in visualising CNNs, including *Activation Maximization, Network Inversion, Deconvolutional Neural Networks (DeconvNet)*, and *Network Dissection*. The first of these was proposed by Erhan et al. [36] and is a technique which provides example input images to the CNN which maximise the activation of specific neurons. This can be used to show what each layer of the network is identifying individually. The technique was proposed over ten years ago and is still in common usage.

In contrast, *DeconvNET* explains the CNN from the perspective of an input image rather than from the perspective of the neurons. Using unpooling and deconvolutional layers, it takes the final feature maps back to the original image size. This shows which features of the input image were detected and used by the CNN. Zeiler et al. [188] have written a series of papers on the *DeconvNet* structure. A later paper by Liu et al. [94] proposes a full visual analytics system to show fully the inner workings of the CNN, known as CNNVis. In this paper, they represent the CNN as a directed, acyclical graph (DAG), which allows them to provide information on not only individual neurons, but the interactions between them. The use of a graphical representation here relates this work to ours where we use networks for the understanding of fitness landscapes.

Other than CNNs, specific visualisation tools have been developed to aid understanding of a variety of models, including RNNs [106], LSTMs [156], and sequence to sequence natural language models [155]. All of these are fully interactive visual analytics systems, though are less likely to be relevant to graph-based data in their specific implementations.

Generative Models One of the earliest works utilising visual analytics for interpretability of generative models was DGMTracker [93], proposed by Liu et al. They develop a technique that identifies which neuron of the network is responsible for a training failure, and further to this, an algorithm to identify how other neurons contributed to the output of this critical neuron. Included is a categorisation of visual analytics methods for interpreting machine learning models into *single-snapshot-based approaches*, which visualise a representative snapshot of training, e.g. outputs from the model, and *multisnapshot-based approaches*, which represent the changes in model weights as a time series. Liu et al. use a version of the latter and in order to reduce clutter, introduce a scheme to determine which subset of time series to visualise to identify critical neurons. An alternative method for decluttering is dimensionality reduction, but this would render it infeasible to determine specific neurons contributing to a training failure, which was the aim of their work.

While Liu et al. provided a general approach to interpretation of generative models, a later paper provided an investigation specifically into visualisation for GANs (Generative Adversarial Networks) in particular, known as GANViz [175]. The adversarial process provides an additional layer of complexity, due to the presence of two neural networks: the generator *G*, which produces new data samples, and the discriminator *D*, which learns to distinguish between real data samples and those generated by *G*. The aim of this paper was to provide a visual tool for use by domain experts, to understand a range of features of *G* and *D* separately during the training process. They also introduced a visual design, *TensorPath*, to provide insight into the workings of neural networks. Since this is a general approach, it can be applied outside the context of GANs. The same year, an additional paper was published providing an alternative visual analytics system known as GAN Lab [68]. The main difference between this and GANViz is that the tool aims to explain GAN training to non-experts, and does so in an interactive manner in a web browser. The researchers used *Tensorflow.js Core*, an in-browser GPU-accelerated deep learning library developed by Google, to allow training of complex GANs in the web browser.

Combining Models or Explanation Approaches Going beyond individual visualisations or visual analytics tools, there have also been other approaches that combine understanding using various approaches. In some cases, such as the work by Ren et al. [133], this involves the comparison of various classifiers on the same task. They propose a tool called Squares which allows a user to distinguish two classifiers that may have a similar accuracy score, but perform in different ways. For each class, the tool allows the user to explore data points which were correctly or incorrectly labelled by the model as being in that class, as well as data which was incorrectly labelled as not being in that class. In other cases, such as the explAIner framework presented by Spinner et al. [152], this can involve the combination of multiple explainability methods in a single process, such as LIME and DeConvNet as just two examples.

Though we develop our own approaches to explainability, these implementations of interpretable information in visualisations are useful references in the design of a future visual analytics system that could incorporate the findings of our explainable community finding work, as well as in the design of our novel visualisation technique for fitness landscape analysis. Some approaches generate particularly relevant insight, such as the use of DAGs to provide graphical representations. In our work on fitness landscape analysis, we also explore the use of a graphical representation for landscapes in continuous space. Similarly, the end-to-end visual analytics tools provide inspiration for how the findings of our explainable community finding work could be applied in future work. Although the community finding algorithms we focus on are not themselves dependent on deep learning, our aim to identify highly predictive interpretable features is closely related to approaches such as SHAP and LIME which quantify feature importance in deep learning models, and whose outputs are visualised in tools such as the explAIner tool.

Having established this foundation of literature in post-hoc explainability and visualisation, we now turn our attention in the next sections to our specific domains of interest where we intend to apply such techniques: social network analysis, and fitness landscape analysis.

2.2.3 Social Network Analysis

The first field where we aim to apply the principles of explainability is network analysis. Although there has been little work on explainability for problems in this field, there is a wealth of existing literature on the problems themselves, and their solutions. In particular, for our work we have focused on the community detection problem, an important task in understanding the structure of networks [43]. Community detection algorithms assign nodes of a network to communities, where nodes belonging to the same community are densely connected by edges and those belonging to different communities are more sparsely connected. Outside of this vague understanding of what contributes a community, algorithms developed to identify them use a range of more precise definitions, and there is no one universal approach. One of the more common metrics with which communities are defined is *modularity*, which is defined as follows for a weighted graph, $G = (V, E)$:

$$Q = \frac{1}{2m} \sum_{ij \in V} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

where m is the sum of all edge weights in the graph; $A_{ij} = 1$ if i and j are connected by an edge and 0 otherwise; k_i and k_j are the degrees of nodes i and j respectively; c_i and c_j are the community labels of nodes i and j respectively; and $\delta(c_i, c_j) = 1$ if these communities are the same, and 0 otherwise. Intuitively, this represents the mean difference between the number of edges we observe connecting nodes in a community and the expected number of edges within this community if they were distributed randomly. Therefore, a high modularity encapsulates our understanding of what makes a community: regions of the network where we see a higher density of connections.

Typically, nodes belong to a single community, making the community assignment of all nodes a partition, however there are some algorithms which can assign a single node to multiple communities [122]. For our work, we focus on those algorithms which partition nodes and therefore do not explore literature on overlapping communities any further.

To our knowledge, explainability of these community detection algorithms is an area which has so far received little attention. The study by Bothorel et al. [16] proposes a methodology to describe results of the community detection algorithm to non-experts, however this differs from ours in that their aim is to assist in making a choice of algorithm for a particular problem, not to specifically explain the algorithm's results. In generating this explanation, we focus on the identification of highly predictive, interpretable network features. In this case, we use *interpretable* to mean those features which are easily recognised and understood by social network analysis experts. An extensive survey of state-of-the-art network features which are in common use is provided by Chakraborty et al. [23], which we referred to when selecting features for our studies. Simple features which quantify the quality of a community are categorised in this survey into those which are based on internal connectivity of the community i.e. the number of edges within the community, external connectivity i.e. the number of edges outside the community, or a combination of the two. We make use of features of all of these types in the work of this thesis. We also incorporate the usage of features which aim to quantify the importance of individual nodes, known as the node's *centrality*. The full list of features used for each experiment are defined in the relevant chapters (Chapters 3 and 4).

In order to develop our explanations for community detection, we also rely on the notion of a node's ease of clustering. Nodes which are easy to cluster are those which are consistently assigned to the same community, while a node which the algorithm finds it hard to cluster will oscillate between two or more communities across successive runs. Existing literature in this vein originates in papers unrelated to networks and community finding, but focused on more general clustering algorithms, e.g. k-means clustering [12, 170]. Other works have also addressed the consistency of community finding algorithms [24, 44], though are less directly relevant as they do not relate this to the ease of clustering individual nodes. Similarly, some work has aimed to develop explainability for a problem in clustering algorithms [96, 112].

Despite the lack of existing literature on using features of networks for explainability, our approach is similar to some previous works which use community quality metrics to evaluate community finding algorithms [29, 120, 181]. Some previous literature has also explored the use of network annotations to provide a better understanding of community memberships. In contrast, in our work we focus on scenarios when such additional network metadata is unavailable [116]. Features of networks have also been used in visual analysis approaches such as the work by von Landesberger et al. [169], where many graph features similar to the ones we have chosen are incorporated into a visual analytics system to provide information about various smaller components of a larger graph. The feature importance information found by our approach could be incorporated into a similar system in the future, for example by highlighting those features which are most relevant for a particular task the researcher is interested in. We do not apply our findings to the development of a visual analytics technique, but envisage that this work could be done in the future.

Although we have previously introduced the community detection algorithms we will use for our experiments in Section 2.1.2, we place these within context of other well-established community detection approaches here, and continue by exploring those approaches at the intersection of community detection with machine learning. Finally, we introduce benchmarks typically used for evaluating community detection approaches.

2.2.3.1 Community Detection Algorithms

The literature on community detection is well-established, with many previously proposed algorithms which address this problem. Those which we use for our experiments have been established in detail in Section 2.1.2; we will now provide additional context on the field of community detection, and how our chosen algorithms fit into this landscape. Among the most well-known is the algorithm proposed by Girvan and Newman [51]. This algorithm takes a divisive approach, meaning that edges which connect nodes belonging to different communities are iteratively remove, so that communities become disconnected. In the case of the Girvan and Newman algorithm, edges are removed according to their centrality, a class of metric which estimates the importance of that edge according to some property. In their case, they used three variations on the *betweenness centrality*, which measures the frequency of the edge's participation in a certain process. Some variations on this approach have since been proposed [131, 162], and there are other algorithms which also take a divisive approach [129]. Nevertheless, the original algorithm of Girvan and Newman remains one of the most popular [43].

In the development of the Girvan-Newman algorithm, they proposed a new measure known as modularity to use as a stopping condition for the iterative process. Modularity is a measure of the density of connections within a cluster relative to the density of connections between clusters, essentially quantifying the concept of a community. Therefore, it has since become a central concept for many more community detection algorithms [26, 28, 33, 172]. Among these is the Louvain algorithm, proposed by Blondel et al. [13]. This algorithm uses a greedy approach, starting with individual nodes and combining them into clusters if it results in a modularity increase, before iterating on this process repeatedly. Other greedy algorithms were also proposed which the Louvain algorithm outperformed [26, 172], however it has since been shown that a flaw in the Louvain algorithm can cause it to yield arbitrarily badly connected communities and the Leiden algorithm [161] has been developed to remedy this.

The Infomap method, developed by Rosvall and Bergstrom [137] and among those introduced for our experiments in in Section 2.1.2, is similar to Louvain though does not depend on modularity, instead using their *map equation* in the objective function. Finally, the label propagation algorithm [130] uses a different approach to those above where every node is initially assigned a unique community label, and these labels are propagated through the network where. The algorithm is less computationally expensive than some other approaches.

The traditional algorithms described above all typically rely on iterative processes which calculate metrics on nodes, edges or clusters of nodes in the network. Some more recent work has attempted to solve the community detection problem using modern deep learning approaches, such as the paper by Yang et al. [182] which uses auto-encoders to identify communities based on modularity or the CommunityGAN paper by Jia et al. [66], which uses GANs to identify community structure. Research in this area is ongoing and faces challenges, including those relating to scalability, and the requirement to know the number of desired communities in advance. A full survey of methods for community detection using deep learning is presented by Liu et al. [92]. Some of the most recent relates to the usage of graph neural networks for community detection, a class of techniques that address the challenges and opportunities of both graph mining and deep learning. The paper by Shchur et al. [149] introduces an approach which integrates a Bernoulli–Poisson probabilistic model with a graph convolutional network (GCN), however this approach identifies overlapping communities, which is beyond the scope of our work. In general, these deep learning approaches offer an alternative perspective on the community finding problem. We briefly acknowledge these here due to the importance of deep learning in the field of AI, and because the black-box nature of deep learning approaches has been a motivating factor in the development of explainability approaches. However, we focus on more traditional techniques, since they are more well-established and consider the application of explainability to deep learning approaches for community detection an element of future work.

2.2.3.2 Community Detection Benchmarks

Some basic examples of networks with communities have existed as benchmarks since the 70s, including the well-known karate club network [187], which demonstrate the importance of community detection in social networks. Additional benchmark networks were introduced in the same paper as the community detection algorithm of Girvan and Newman [51]. More recently, the LFR benchmark algorithm, introduced in detail in Section 2.1.2, has been developed to generate synthetic algorithms with ground-truth communities [86]. This approach features heavily in the work of Chapters 3 and 4, where we use it to generate a large number of networks on which to evaluate interpretable features for post-hoc explanations.

Existing work on comparing the performance of several algorithms on these graphs has guided our choice of algorithms and data for the experimental evaluation of our proposed features. In a subsequent paper [84], the performance of several well-known community finding algorithms is compared on this benchmark data. Lee and Archambault [88] find that humans behave in a similar way to Lancichinetti et al. when observing their own social network, confirming that the Infomap, Louvain and Girvan-Newman algorithms were the best-performing. Previous work in computational social science has also compared the performance of community finding algorithms on real networks [30, 50, 125].

We have now introduced the key elements of social network analysis which are relevant for the work of this thesis, including the traditional community finding approaches we aim to explain, and the benchmarks we use for our experiments. Future work may also consider the more recent attempts to apply deep learning to community detection. We now turn to the second of our domains of interest for applying explainability: fitness landscape analysis.

2.2.4 Fitness Landscape Analysis

The second field where we aim to apply the principles of explainability is optimisation. Previous work by Bacardit et al. [9] called for the evolutionary computation community to consider the intersection of their field with explainable AI, highlighting the existence of interest in adjacent fields in the application of explainability. In chapter 6, we focus on the problem of fitness landscape analysis (FLA), a field where visualisation techniques have already been used to improve understanding. In particular, our aim is the development of a novel visualisation technique to complement those which already exist.

2.2.4.1 Existing Techniques in FLA

One of the principal techniques for visualisation within FLA is the *local optima network* [118] LON. LONs are designed to illustrate the structure of a landscape as a graph $G = (V, E)$ such that V is a set of nodes representing basins of attraction (local optima) connected by edges $e_{ij} \in E$ wherein a node v_i is connected to v_j if they are deemed to be neighbours. In the original work by Ochoa et al. [118], neighbourhoods were defined for the specific problem type, which dealt with binary NK landscapes, which consist of every string of length N when chosen from a given alphabet. Since that work, LONs have been extended to a wide range of solution representations and problem types (including multi-objective [40]). In the case of continuous problems, which are the focus of our work, a variant of the discrete LON formulation was proposed in which the distance between two nodes was used to identify neighbours [2]. Figure 2.2 illustrates a LON constructed for a 5D instance of the Rastrigin problem. Concepts from LONs have since been extended to formulate *search trajectory networks* (STNs) [117]. Since mapping the search process of an evolutionary algorithm through the space is beyond the scope of our work, we do not consider STNs further.

Inspiration for our approach comes from existing literature in the visualisation community on extrema graphs [62, 63]. Extrema graphs, used in volume visualisation, provide a structure to extract isosurfaces automatically from volumes. Two-dimensional surfaces are defined in three dimensions, similar to the one-dimensional contour lines on a map, that separate areas of higher density from areas of lower density in the volume. In isosurface extraction, the graphs are usually embedded in three dimensions, but that is not always the case for visual landscape analysis. Therefore, for the purposes of landscape analysis, we visualise the calculated extrema graph directly through dimensionality reduction of the maxima/minima along with samples along the edges between them.

2.2.4.2 Dimensionality Reduction

Dimensionality reduction takes data in high dimensional spaces and maps them down to lower dimensional spaces which for visualisation is often two dimensions. Multidimensional scaling

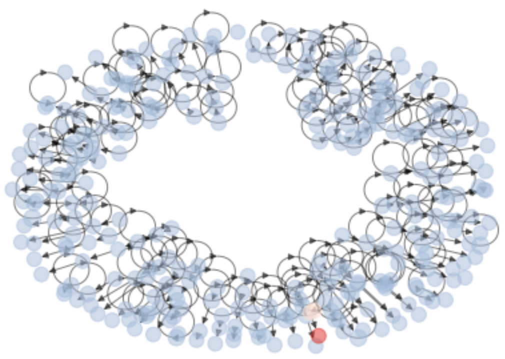


Figure 2.2: An example image of a LON, a prominent visualisation technique for fitness landscape analysis. This LON represents the Rastrigin function with 5-dimensional decision space using the default settings from Adair et al. [2]. The dark red node represents the global minimum. The series of red nodes show the basin of attraction for the global minimum, while other blue nodes represent basins for local minima. Arrows are also included showing the direction of travel in the basin-hopping algorithm used to construct the LON.

approaches [18, 27, 32, 81, 165] are one type of these methods which optimises distances between points in the low dimensional space so that they are representative of distances in the high dimensional space. MDS minimises a cost function called "stress", which is a residual sum of squares:

$$Stress_D(x_1, x_2, \ldots, x_n) = \sqrt{\sum_{i \neq j = 1,2,\ldots,n} (d_{ij} - ||x_i - x_j||^2)}$$

where D is the proximity matrix, with d_{ij} representing the distance between points x_i and x_j. MDS has previously been used in other approaches to embed search spaces for visualisation, such as in the work by Michalak [105]. However, our work differs in the use of a graphical representation. In addition, MDS has been used to visualise high-dimensional spaces in many-objective optimisation, presenting trade-off surfaces for problems comprising four or more conflicting objectives [173, 174]. Later work considered the visualisation of a multi-objective optimiser's route through the search space in a way that enabled landscape characteristics to be inferred from optimiser behaviour.

It is worth noting that dimensionality reduction is also a commonly used technique for visualisation of deep learning methods. For example, in the work by Rauber et al. [132] they project the learned representations of hidden layers of neural networks for visualisation. Deep learning has also been used to improve upon existing dimensionality reduction techniques by

Espadoto et al. [37]. Although the work of this thesis does not focus on the explanation of deep learning methods, these works demonstrate dimensionality reduction as a well-established technique for visual explanations which we adopt in our work on fitness landscape analysis.

Dimensionality reduction and visualisation approaches such as extrema graphs and LONs comprise the most essential elements applicable to our work on developing a novel visualisation approach for fitness landscape analysis, the second of our two domains of interest for explainability. To conclude the background and related work, we now give consideration to uncertainty quantification, a collection of approaches closely related to, though generally considered separate from, explainability techniques.

2.2.5 Uncertainty Quantification

In this final section of the related work, we draw our attention to the literature on uncertainty modelling and prediction interval generation in machine learning, which is well-established. We first explore the prominent techniques within the field, before commenting on the relationship between uncertainty quantification and explainable AI, including references to existing works that have addressed this interaction.

In general, neural networks are subject to both model-dependent errors, and data-dependent errors, leading to classification of uncertainty into two types: systemic uncertainty (caused by errors from the model) and aleatoric uncertainty (caused by variation in the data) [61]. Model-dependent errors are caused by an insufficient model, a faulty training procedure, or a training set which is not representative of the underlying distribution. Data-dependent errors are caused by the information loss which occurs when we try to represent real phenomena with a dataset.

A thorough classification of uncertainty modelling methods can be found in the survey by Gawlikowski et al. [49]. One of the first approaches, Monte Carlo (MC) Dropout, uses dropout as a Bayesian approximation [47]. These dropout layers are activated randomly several times during inference to generate a sample distribution from which the prediction can be estimated and a bootstrapped confidence interval computed. Deep Ensembles [83] were inspired by MC Dropout but differ slightly in that duplicate neural networks are used in parallel in place of random dropout activations. The duplication of neural networks required by Deep Ensembles mean that MC Dropout has a lower computational complexity. More recent work by Wenzel et al. [178] proposed the *hyper deep-ensemble* which displays an even greater performance, with a greater computational cost. DropConnect [108] is another uncertainty modelling method similar to MC Dropout, but differs in that incoming activations to a node are randomly dropped rather than dropping activations for all subsequent nodes.

The above methods focus on model, or epistemic, uncertainty [61]. One of the simplest ways for modelling aleatoric, or data, uncertainty is by using quantile regression [77]; another is through the use of conformal methods [146]. The former method is a broad class of approaches whereby a regression technique is used to predict not the expected value of a response variable, but a given quantile. For example, if the given quantile is 0.4, the regression model is trained to predict an output such that the true label is less than this output 40% of the time, and greater than the output 60% of the time. This approach can be combined with conformal methods through conformalised quantile regression [136]. Conformal methods are typically combined with another uncertainty quantification approach in the following way. Alongside classical training, validation and test sets, a segment is selected from the data to be the calibration set. The performance of the quantiles is evaluated on this calibration set, and a calibration value is then calculated which should be added or subtracted to all quantile predictions to mitigate errors.

These methods are well-established, but many other recent methods have been proposed and this remains an ongoing area of research. One limitation is the assumptions made about data distributions, something which is highly relevant for our work on social media content popularity where the distribution of received views is highly unusual. In their paper, Khosravi et al. [72] propose the LUBE (lower upper bound estimation) method to circumvent this issue. Another recent paper by Pearce et al. [124] further built upon this work with their quality-driven prediction intervals. In both cases, an objective function is designed to minimise the width of a prediction interval while maximising its accuracy (i.e. the number of true labels which fall within the predicted upper and lower bounds). In the latter case, they incorporate both epistemic and aleatoric uncertainty into their predictions, while LUBE tackles only the latter. Salem et al. [143] further built upon this by introducing a prediction interval model which additionally generates predictions of the expected value.

One of the most recently proposed methods, NOMU (Neural Optimization-based Model Uncertainty) [57], similarly uses an approach where multiple values are predicted by the neural networks, however in this case, the aim is to predict the expected value and the model uncertainty rather than the expected value and the bounds. For this reason, this approach deliberately separates the impact of model uncertainty from that of the data uncertainty, allowing for the two to be disentangled.

In our work, we apply uncertainty quantification to a problem in predicting information cascades, specifically the cascade of views received by an item of social media content. In this domain, we aim to predict the number of views a piece of content will receive over a given time horizon, based on features of the content such as the popularity of the user who posted it, the location the content is posted in, and features of the content itself, such as word embeddings for text contained in the content. Uncertainty quantification applied to this problem would provide beneficial additional information, as a single number prediction for the number of views is likely to be inaccurate, where upper and lower bounds may give a better idea of the possible range of views. Perhaps surprisingly however, uncertainty modelling has received little attention in the recent literature on predicting information cascades, with a few exceptions [179, 180, 192], which have focused on applying variational auto-encoders to mine rich representation from structural, temporal and content properties of information cascades. Nevertheless, even this existing literature does not address the problem of social media content popularity, which is of particular interest to social media platforms. We anticipate that the problem of predicting numbers of views will be particularly challenging due to the unusual skewed distribution found in content popularity.

2.2.5.1 The Relationship of Uncertainty Quantification to Explainability

Uncertainty quantification is not itself usually considered to be a method for explainability. Indeed, many models which aim to quantify the degree of uncertainty in a prediction may themselves be black-box approaches, for example the neural network used in MC Dropout [47]. In fact, in the LIME paper [135], introduced in section 2.2.1.1, the lack of quantification of uncertainty on methods such as permutation importance is highlighted as a challenge for explainability. Nevertheless, uncertainty quantification shares some goals with the development of explainable AI. In the paper by Seuss [145], it is observed that "while the methods of Explainable AI try to show the way to the decision, the methods of Quantification of Uncertainty try to give a realistic evaluation regarding the reliability of the decision." Thus, both explainability and quantification of uncertainty are highly important for robust decision making, a process which AI is designed to assist with. Some work explores how models can be designed with both uncertainty and explainability in mind [25, 67, 103, 189], for example by applying explainability approaches to models which can generate an estimate of uncertainty, or by estimating the uncertainty of feature importance scores.

In this thesis, we tackle the quantification of uncertainty in a different setting to those where we aim to apply explainability techniques, but unite this work under the common theme of improving trust in complex algorithms for social network problems. For the earlier work of Chapters 3 and 4 where we use feature importance scores, we tackle the lack of uncertainty quantification by calculating them on a range of training datasets and performing statistical analysis to ascertain confidence intervals for the true importance of a feature.

2.3 Design Choices

We conclude this chapter by summarising the design choices we have made in light of the available literature, and clarifying the reasoning behind those choices within the context of our specific problem applications.

Explainable Community Finding In Chapters 3 and 4 we focus on the community detection problem, where we develop a novel methodology to identify features of importance which can be used to explain the output of community finding algorithms, from two different perspectives in the respective chapters. In doing so, we first select existing feature importance scoring methods to use within our methodology: permutation importance and SHAP. The former has been chosen as it is one of the simplest and most computationally inefficient approaches to implement, while the latter provides the beneficial theoretical guarantees of Shapley values, i.e. the four "fair payout" properties. Since the models we use in this thesis which require explanation are random forests, we use the TreeSHAP approach. The benefit of using this over other Shapley value approaches are its specific design for tree-based models, and the saving on computational cost.

For one of our classification tasks, we distinguish nodes which are "easy to cluster" from those which are "hard to cluster". We have chosen to centre our definition of a node's ease of clustering on its entropy in a *coassociation matrix*. Entries in the matrix describe how frequently two nodes are clustered into the same community. The concept of a coassociation matrix describing the relationship between pairs of nodes was derived from work proposed by Strehl [154]. In a similar approach, consensus clustering is explored for determining community structure over successive runs of the algorithm [85].

In our first experiments demonstrating this novel methodology, we then focus on generating explanatory information for the Louvain algorithm, the Infomap algorithm, and the label propagation algorithm. For the approach that we developed, we required algorithms which were stochastic with notable variation in results across several runs. These are particularly interesting from an explainability perspective as the user may wish to understand why one output was observed the first time the algorithm is used, while a different output is seen on another occasion. The version of our methodology presented in this thesis also required us to select algorithms producing partitioned communities where nodes belonged to only a single community. We selected these three algorithms in particular as they have been widely employed in the literature and also since each algorithm differs considerably in terms of the objective which it attempts to optimise. Thus, analysing the results with these three algorithms would allow us to see how dependent the results are on the objective of the algorithm itself. The previous work by Lee and Archambault [88] which found that humans behave in a similar way to Lancichinetti et al. [84]. when observing their own social network also confirmed that the Infomap, Louvain and the Girvan-Newman algorithms were the best-performing. This further guided our decision to include Infomap and Louvain in our experiments.

Though community detection approaches exist which are based on deep learning techniques such as graph neural networks, we chose to focus our research on these traditional algorithms instead. Choosing the best performing algorithms is not of interest, as our focus is on improving explainability rather than optimising for the community detection problem. In this case, the Louvain algorithm, the Infomap algorithm and the label propagation are all in common usage and can benefit from the development of explainability approaches to aid the researchers who are working with them. They also differ from other approaches within machine learning for which explainability techniques have already been developed, particularly in that they don't make use of a feature matrix, thus requiring a new approach to be developed. This makes them a strong candidate to be the subject of novel explainability research.

Our experiments allow us to identify which features from a longlist are the most informative for explaining the outputs of these algorithms, however, as they do not use a feature matrix, we selected this longlist ourselves. Our choice of features was informed by the survey by Chakraborty et al. [23], however, as our choice of features is restricted to those required for the experiments of a specific chapter, we have chosen to provide the full list of definitions in Chapters 3 and 4 themselves. These can be found in Sections 3.1.1 and 4.1.3.

Regarding our choice of data, we have chosen to focus on synthetic networks generated by the LFR benchmark algorithm as it allows us to generate much larger datasets than are readily available elsewhere in the literature. In particular, large datasets of real networks are in low supply, and the LFR benchmark aims to generate synthetic networks which closely simulate the structure of real-world networks. In addition to this benefit, using the LFR generator allows us to vary the mixing parameter, μ, to observe the effect that the degree of community mixing has on the results.

Uncertainty Quantification In Chapter 5, we focus on the problem in social networks of predicting popularity of social media content. Although elsewhere in this thesis we use approaches which are not dependent on black-box deep learning models, for this task we do employ the use of neural networks, as the benefits of explainability are dependent on the needs of users. In this case, researchers at Meta working on this problem use deep learning as it is a highly complex problem, for which they employ thousands of features and invest a large research allocation into developing the most accurate models possible. For this use case, a simpler model would not perform as well, so we instead apply uncertainty quantification to improve the general understanding of the model predictions, an approach which is beneficial due to the lack of information gained when making a point prediction for a regression problem where outputs take such a large range of values.

In the first case, we apply well-established approaches MC Dropout [47] and quantile regression [77]. We chose these as they are two of the most commonly used and well-researched approaches to uncertainty quantification, and also because they represent modelling of the two different uncertainty types - aleatoric and epistemic uncertainty. MC Dropout was chosen over other more computationally demanding approaches such as deep ensembles [83], which are less scalable in storage. Preliminary experiments were also done with quality-driven prediction intervals introduced by Pearce et. al [124], though we did not develop this approach as it performed poorly with our dataset, which has a highly unusual distribution. Recent advances in parallel work have led to new methods for prediction interval generation [57,72,143] which could be considered in future work.

Fitness Landscape Analysis In Chapter 6, we develop our extrema graphs visualisation approach as a complementary one to the LON [118]. In doing so, we make use of the extrema graph approach first seen in isosurface extraction in volume visualisation, where transitions are captured between both maxima and minima embedded in two dimensions through dimensionality reduction techniques. Our choice to use dimensionality reduction for visualisation was made in order to represent information for a high number of dimensions easily, providing a different perspective to that seen in the classic LON. In particular, we use multidimensional scaling (MDS) for our prototype, though another approach could be easily substituted. Our reasoning behind this choice was that MDS preserves relative distance, which would allow global information to be represented in the visualisation as well as local interactions.

2.4 Summary

In this chapter, we have placed our work into the context of existing literature from various fields which are relevant to the work of this thesis, including post-hoc explainability, visualisation, social network analysis, fitness landscape analysis, and uncertainty quantification. Our work shares the philosophy of both explainable AI and uncertainty quantification: that the information used to make decisions on the basis of black-box algorithms must be trustworthy and well-understood. It therefore draws upon ideas from both fields. The domains of application where we focus our attention are those where either social networks or systems play a key part, or where network science can be applied to bring novel insight. In the upcoming chapters, we explore several such applications where explainable AI or uncertainty quantification can be applied.

The first of these, in Chapters 3 and 4, is the community detection problem, a problem which belongs to the field of social network analysis. Here we apply existing post-hoc explainability techniques to identify interpretable features which can be used in visual analytics systems in the future. The second application we explore is the prediction of content popularity on social media platforms. In this setting, we analyse how existing uncertainty quantification approaches perform in the face of the skewed distribution found in social systems. This analysis can be found in Chapter 5. Finally, we develop a novel visualisation approach which makes use of network structure to aid understanding of fitness landscapes, and present this in Chapter 6. While we explore three different applications, in all cases our goal is develop methodologies which are able to improve the understanding of outputs from black box systems. In all cases, we take inspiration from more than one field explored in the reported literature of this chapter. Therefore, these various solutions are tied together thematically, even where they tackle different problems.

This concludes our exploration of the themes which are present throughout the thesis, and of the existing literature of greatest relevance to those themes. We will now begin, in the next chapter, by applying post-hoc explainability to the problem of community detection.

Chapter 3

Explainable Community Finding

Although explainable AI has primarily focused on applications in machine learning, complex algorithms are also used to generate insight and solutions to challenging problems in other fields. The first of these which we focus on is social network analysis. As well as being highly complex, these algorithms can also be stochastic, leading to results that vary across multiple iterations. For these reasons, issues relating to interpretability and trust which pervade machine learning are also present in the field of social network analysis.

One such problem which can be solved using complex, often stochastic algorithms, is that of community finding, or community detection. In this setting, communities are loosely defined as sets of nodes in a network which are densely connected by edges, while having more sparse connections to other nodes of the network outside of the community. Communities may overlap, though many common community detection algorithms partition the nodes into distinct sets. Such algorithms normally try to optimise a quality function, usually through a heuristic. Nevertheless, this process can be lengthy and convoluted, especially for networks with many nodes and edges, leaving even experts unable to obtain a deep understanding for the reasons why nodes have been clustered into certain communities.

Therefore, we propose the use of existing explainability approaches from machine learning applied to this problem setting in social network analysis. In particular, we focus on the identification of interpretable features, and the evaluation of their relative importance in a given community detection problem. The benefit of this approach is that features familiar to domain experts can be chosen such that explanations are tailored to their specific understanding. Here, we focus on social network analysis experts as the domain experts in question, and therefore consider common network analysis metrics as potential interpretable features.

In this chapter, we present a methodology for identifying those interpretable features from a given longlist which are best able to explain the outputs of community detection algorithms, as well as an application of this methodology to three well-known approaches: the Louvain algorithm [13], the Infomap algorithm [137], and the label propagation algorithm [130]. When applying our methodology, we utilise the LFR benchmark graph dataset generator, proposed by Lancichinetti et al. [86] to prepare a dataset of graphs on which to evaluate the longlist of interpretable features. This generator creates graphs with ground truth community labels on each of the nodes according to a mixing parameter, μ, which introduces noise to the communities relative to its value. For low values of μ, the communities remain well separated and thus easy to detect, but as the mixing parameter increases, communities become harder to identify. This allows us to vary community mixing for our experiments.

We conclude with a discussion of the insights gained from this analysis, where we find that the same features are identified across the three algorithms on LFR graphs. At the single node level, these features were: clustering coefficient; triangle participation; eigenvector centrality; and expansion. At the node-pair level, these features were: the Jaccard coefficient; the cosine similarity; and to a lesser degree, the maximum betweenness centrality of an edge along the shortest path between the two nodes. All of these features are defined in section 3.1. The work of this chapter is based on our paper published in the Springer journal Applied Network Science [140].

3.1 Problem Formulation

Due to their widespread adoption and suitability for our proposed methodology, in our experiments we focus on stochastic algorithms, where the community structure can change between successive runs, and on algorithms which find node partitions (i.e. each node belongs to exactly one community). Extending our approach to algorithms which generate overlapping communities will require additional steps, so we reserve this for future work, which is expanded in section 7.1. As the intention is to identify features which contribute intuitive understanding, our emphasis is on selecting features which are simple and easily understood to end-users, though specifically those with social network analysis expertise. We propose a model-agnostic methodology which can be adapted to any stochastic algorithm of interest; however, we test it here on three in particular.

We distinguish between two "levels" of graph feature, allowing for understanding of the nodes' community membership from two different perspectives. The first of these is at the *node-level*. Features at this level are calculated for individual nodes of the graph, with the aim to understand the community membership of that specific node. To motivate this problem in a social network context, suppose a node is occasionally classified as belonging to a community on certain runs of a community finding algorithm, where this community is of particular interest to the researcher as members are presumed to be more likely to share inappropriate or dangerous content on a social media platform. Understanding why this node has this varying classification would be important as this classification is not certain and could have important repercussions for the individual. The second is at the *node-pair-level*, where features are calculated for pairs of nodes. The aim is to understand why two nodes belong to either the same or different communities. If one node is identified as belonging to a community of interest and the other is not where the two appear to be quite similar, it could be interesting to understand what distinguishes the two.

In this work, we use a large number of synthetically-generated graphs to verify our approach. We employ the use of synthetic data to ensure the results are not a consequence of the characteristics of a single network (as real data is sparsely available). However, with the aim to apply these results to real data in the future, we use a synthetic generation process which can closely mimic the observed structure of real-world networks. Specifically, the synthetic graphs were generated using the implementation of the LFR benchmark algorithm [86] in NetworkX. An additional benefit of this approach is that existing work has already evaluated the performance of community finding algorithms on LFR graphs. We use several values of the LFR mixing parameter μ in order to ascertain whether the separation of the communities affects the identified features.

Our approach is to identify a longlist of features at both the node-level and the node-pair-level. We then use these features as the input data for a classification task, and extract the most informative features using both permutation importance (Algorithm 2.1) and Shapley values for our trained model. Since Shapley values are computationally expensive to calculate in the original closed-form definition [148], we use the high-speed estimation approach, Tree-SHAP, to approximate these [98]. Permutation importance values were preregistered in our statistical analysis report and subsequently used during our pilot study (described in the later section 3.2.2), so we therefore report these as the main results and perform further statistical analysis on these alone. However, Shapley values are well known among the explainability community, and are known to have mathematically desirable properties for producing explanations. Thus, we include experiments using this approach as additional results to support the findings from the permutation importance experiments and verify that the drawn conclusions are largely invariant to the explainability metric chosen. Henceforth we will refer to both permutation importance and Shapley values as "importance scores".

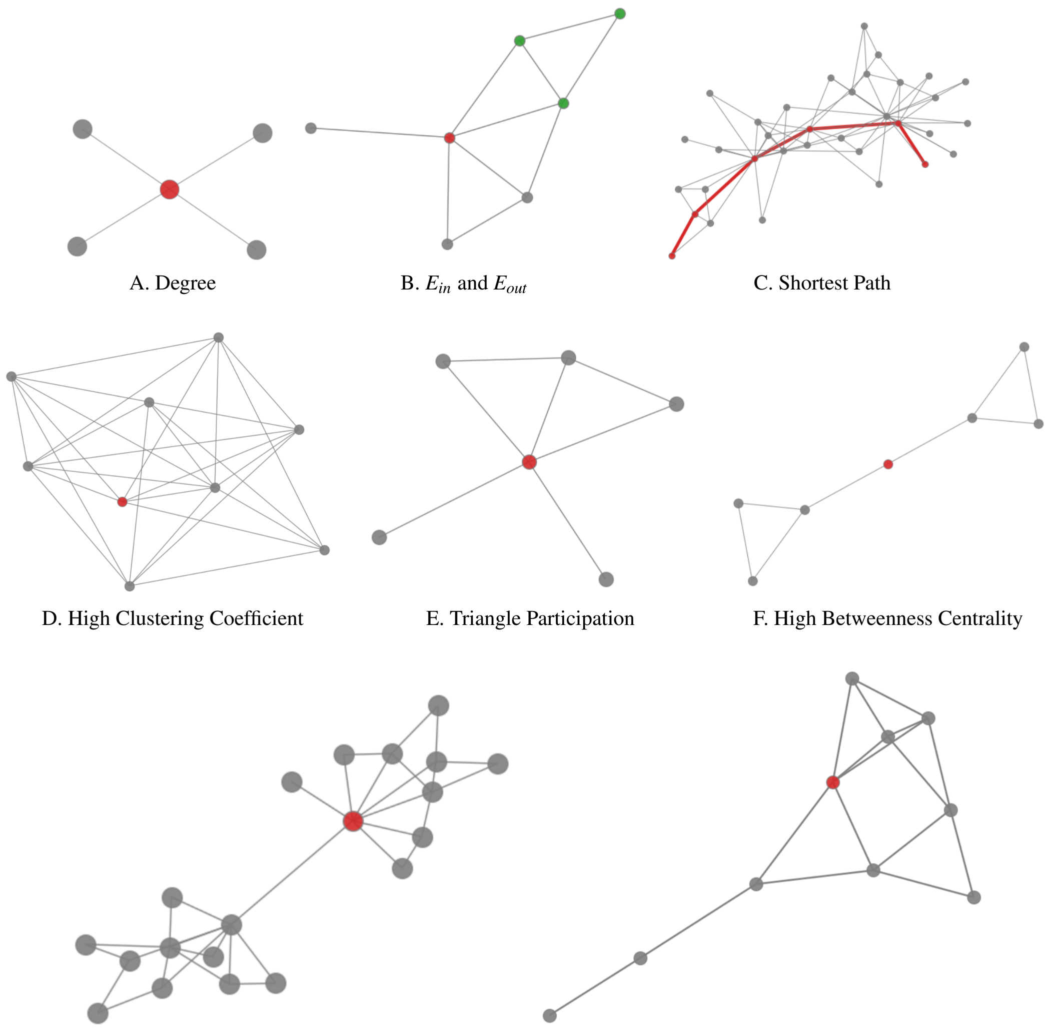
Since some of the features in our longlist depend on nodes' community labels, for example, the number of edges a node shares with members of its community, we calculate these from many runs of the community finding algorithm using a mean average. Although we have access to ground truth communities, we use those identified by the community finding algorithm for the following reason. When providing explanations for community membership, it is likely to be clearer why a node has been assigned to a community by an algorithm when analysing its presence in that community, rather than a ground truth community which was unknown to the algorithm. If the feature does not depend on community label, such as the degree of the node, it can be directly calculated.

As stated in the related work in Chapter 2, we consult the survey by Chakraborty et al. [23] to inform the selection of features which are widely adopted, state-of-the-art metrics for community evaluation such that they can be easily recognised and interpreted by experts on network analysis. We now provide definitions for the full list of features selected for evaluation in these experiments below.

3.1.1 Node Features

For a graph $G = (V, E)$ with V denoting the set of nodes and E the set of edges, the node-level features we have selected are defined as follows for a node $i \in V$, where i is a member of the community $C \subset V$. Figure 3.1 provides illustrative examples to assist with intuition of these definitions.

- *Degree*: The number of edges adjacent to i, $\deg(i)$. For the red node in Figure 3.1A, the degree is 4.
- E_{in}: The number of edges adjacent to i within its community. Adapted to a single node from the original definition by Radicchi et al. [129]. For the red node in Figure 3.1B, where the green nodes are those within its community and the grey nodes are those belonging to other communities, E_{in} is 2.
- E_{out}: The number of edges adjacent to i which connect it to nodes outside its community. Adapted to a single node from the definition by Radicchi et al. [129]. For the red node in Figure 3.1B, where the green nodes are those within its community and the grey nodes



A. Degree B. E_{in} and E_{out} C. Shortest Path

D. High Clustering Coefficient E. Triangle Participation F. High Betweenness Centrality

G. High Eigenvector Centrality H. High Closeness Centrality

Figure 3.1: Illustrative examples to aid with intuition behind node feature definitions. Reference nodes and edges are highlighted in red.

are those belonging to other communities, E_{out} is 3. Note that $E_{in} + E_{out} = \deg(i)$, which for the red node is 5.

- *E_{in} over E_{out}*: For a given node, the ratio of the number of edges connecting it to other nodes within the same community, relative to the number of edges it has to nodes in other communities:
$$\frac{E_{in}}{E_{out}}$$
If this metric has a value greater than 1, there are more edges connecting the node to its own community; if it has a value lower than 1, there are more edges connecting it to nodes outside its own community.

- *Out Degree Fraction (ODF)*: Adapted to a single node from the definition by Flake et al. [41], this is the ratio of edges connecting node i to nodes in other communities, relative to its total degree:
$$\frac{E_{out}}{\deg(i)}$$
This metric has a maximum value of 1, indicating that all the node's edges connect it to nodes outside of the community. The lower the value, the more of its edges connect it to members of its own community, down to a minimum value of 0 when all edges connect it to its own community.

- *Expansion*: Adapted from the definition by Radicchi et al. [129], this is number of edges from a single node i to nodes assigned to other communities, normalised with respect to the number of nodes in the same community as i:
$$\frac{E_{out}}{|C|}$$
High values indicate that the node has a large number of connections outside the community relative to the size of its own community, i.e. that it "expands" the wider circle of connections the community has. Lower values indicate that the node has a small number of connections outside the community relative to the size of its own community.

- *Cut Ratio*: Adapted to a single node from the graph cut measure discussed by Fortunato [43]. As with the metric above, this considers the number of outgoing edges from i to other communities, but in this case normalised with respect to the number of nodes **not** in the same community as i:
$$\frac{E_{out}}{|V| - |C|}$$

High values indicate that the node is connected to a high proportion of the nodes outside its community, while low values indicate that the node is connected to a low proportion of the nodes outside its community.

- *Conductance*: Adapted to a single node from the clustering objective described by Shi and Malik [150], this measure is the ratio between the connections for node i within its community and its total number of connections:
$$\frac{E_{out}}{\deg(i) + E_{in}}$$

- *Average Shortest Path*: The mean of the shortest path lengths from node i to all other nodes in the graph. An example of one shortest path is shown for two nodes in Figure 3.1C. A low value indicates that the node is central within the graph, and is not distanced by a large number of edges from any other node. A high value indicates that the node is far away from the centre of the graph, and is distanced from several other nodes by a large number of edges.

- *Triangle Participation*: Let c_i be the number of nodes with which node i shares a common neighbour within its assigned community. Then triangle participation is given by the fraction:
$$\frac{c_i}{|C|}$$
In Figure 3.1E, the red node shares a neighbour with 3 of its 5 neighbours; the other 2 are connected only to the red node. Therefore, it has a triangle participation of $\frac{3}{5} = 0.6$. A high value of triangle participation means that the adjacent nodes in the community are all highly connected; a low value means that the node has several unique connections, i.e. is connected to several nodes that are not connected to eachother.

- *Clustering Coefficient*: The local clustering coefficient of a node is a measure of how close its neighbours are to forming a clique, introduced by Watts and Strogatz [177]. Formally, let T_i be the number of triangles containing i across the whole graph. Then the clustering coefficient of node i is given by:
$$\frac{2T_i}{\deg(i)(\deg(i) - 1)}$$
In Figure 3.1D, the red node has high clustering coefficient as all its neighbours are also connected to each other. A low value of clustering coefficient would indicate that the adjacent nodes are not well connected to eachother.

- *Betweenness Centrality*: Let $\sigma(j,k)$ be the number of shortest (j,k) paths, and $\sigma(j,k|i)$ be the number of those paths that pass through i. Then the betweenness centrality of node i is given by [17]:
$$\sum_{j,k\in V} \frac{\sigma(j,k|i)}{\sigma(j,k)}$$
Note that if $j = k$, $\sigma(j,k) = 1$ and if either j or $k = i$, then $\sigma(j,k|i) = 0$. Intuitively, a high betweenness centrality score for a node often indicates that it holds a bridging position in a network. Specifically, the shortest path between every pair of nodes is calculated, and nodes with a high betweenness centrality are those nodes which a high number of these shortest paths pass through. The red node in Figure 3.1F has high betweenness centrality.

- *Eigenvector Centrality*: Proposed by Bonacich [14]. The eigenvector centrality of node i is the i^{th} entry in the vector $\mathbf{x}$ which solves the eigenvector equation:
$$\mathbf{Ax} = \lambda\mathbf{x}$$
where $\mathbf{A}$ is the adjacency matrix with node i represented in the i^{th} row/column. Specifically, the eigenvector solving this equation with the highest eigenvalue is chosen. Based on the definition above, this measure deems that a node is important if it is connected to other important nodes. In Figure 3.1G, the red node has the highest eigenvector centrality as it is connected to other important nodes in its own cluster, as well as the connected node in the second cluster.

- *Closeness Centrality*: Refers to the centrality measure proposed by Freeman [45]. Let $d(i,j)$ be the length of the shortest path between nodes i and j. Then the closeness centrality of node i is given by:
$$\frac{|V|-1}{\sum_{j\neq i} d(j,i)}$$
This provides us with an assessment of the extent to which node i is close to all other nodes in a network, either directly or indirectly. A high value indicates that the node is close to most other nodes, while a low value indicates that it is distanced from some other nodes by a large number of edges. The red node in Figure 3.1H has the highest closeness centrality of all the nodes in the graph.

3.1.2 Node-Pair Features

Figure 3.2 provides illustrative examples to assist with intuition of these definitions. Given a pair of nodes (i, j), we define a number of node-pair-level features:

- *Shortest Path Length*: The least number of edges separating nodes i and j, as defined to calculate the "average shortest path" node feature above. A low value indicates that the two nodes are close together, e.g. connected by an edge, or sharing common neighbours. A high value indicates that they are distantly separated by many intermediary connections.
- *Common Neighbours*: The number of shared nodes adjacent to both i and j, which we denote as n_{ij}. In Figure 3.2A, the red nodes share 2 common neighbours, while the remaining nodes are connected to only one of the two red nodes.
- *Max Edge Centrality*: The maximum over centralities of all edges along the shortest path. The edge centrality is defined in a similar manner to betweenness centrality for nodes [17]. That is, for a given edge e, we compute

$$\sum_{j,k \in V} \frac{\sigma(j,k|e)}{\sigma(j,k)}$$

  where $\sigma(j,k|e)$ now refers to the number of shortest paths between j and k passing through an edge e rather than a node i. Figure 3.2B shows two nodes which are connected by an edge with high edge centrality, as it participates in many shortest paths across the whole graph due to its position connecting the two clusters. Therefore, these two nodes will have high maximum edge centrality. Meanwhile, Figure 3.2C shows two nodes which are connected by an edge with low edge centrality, as it does not participate in many shortest paths across the whole graph. Therefore, these two nodes will have low maximum edge centrality, as this one edge comprises their shortest path.
- *Cosine Similarity*: Frequently used to measure similarity for textual data, but can also be applied to assess node-pair similarity in the context of graphs:

$$\frac{n_{ij}}{\sqrt{\deg(i)}\sqrt{\deg(j)}}$$

  In Figure 3.2A, we have established that the red nodes have 2 common neighbours. Their individual degrees are 4 and 5 respectively. Therefore, their cosine similarity is

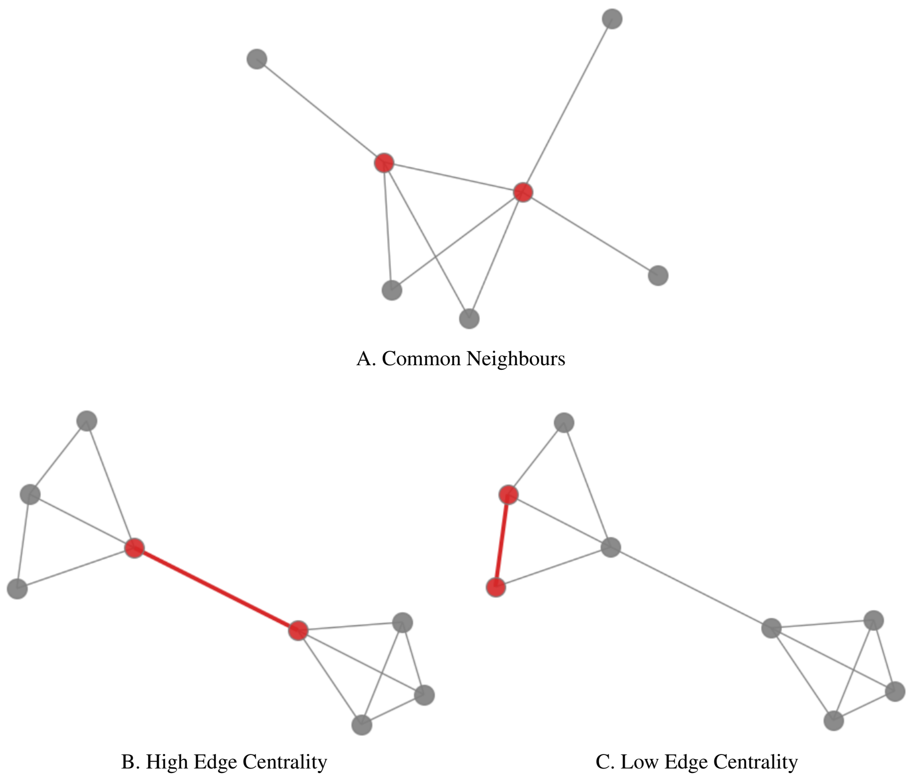


Figure 3.2: Illustrative examples to aid with intuition behind node feature definitions. Reference nodes and edges are highlighted in red.

$\frac{2}{\sqrt{4}\sqrt{5}} = \frac{1}{\sqrt{5}}$. The higher the cosine similarity, the higher the proportion of neighbours of the two nodes which are common neighbours.

- *Jaccard Coefficient*: A common set similarity measure, originally proposed in [64]. In a graph context, let $\Gamma(i)$ be the set of neighbours of node i. Then the Jaccard coefficient of nodes i and j is given by:

$$\frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|} = \frac{n_{ij}}{|\Gamma(i) \cup \Gamma(j)|}$$

A higher value for this measure indicates a greater level of overlap between the neighbours of i and j, relative to their full sets of individual connections. Using Figure 3.2A again, the union of their sets of neighbours is the whole graph, comprising 7 nodes. Therefore, their Jaccard Coefficient is $\frac{2}{7}$.

3.1.3 Classification Problems

For node-pair-level features, there is a simple binary classification problem where nodes are labelled as belonging to the “same community” or to “different communities”. Although we have ground truth for this classification, we must use the outputs of the community finding algorithms to construct our label, in order to ensure that the features we identify as important reflect the performance of the algorithm itself, which is the thing we aim to explain. Since the algorithms of interest in our work are stochastic in nature, a pair of nodes may sometimes be in the same community, while for other runs of the algorithm the pair may not appear in the same community. Over the course of many runs of a given algorithm, pairs can simply be labelled as “same community” if they are in the same community for more than half of the runs, and “different community” if they are in the same community for less than half of the runs. In the unlikely event they are in the same community for exactly half of the runs, we have chosen arbitrarily to label them “same community”.

For node-level features, defining a classification problem is harder since, on consecutive runs of the community detection algorithm, the number of communities can vary, or the community labels can be permuted. Thus, classifying a node into its “correct” community is not a well-defined problem. Instead, we propose a binary classification problem determining whether the node is “easy” or “hard” to assign to a community, by observing how frequently it flips between communities on successive algorithmic runs. To define this mathematically, we require a coassociation matrix, described in Section 3.1.4 below. This will allow us to identify features that are predictive in whether a node is strongly associated with a specific community (near its “centre"), or whether it lies on the border between two or more communities. Nodes of the latter type may be of particular interest in certain domains, such as public health.

In order to label the nodes as “easy” or “hard” to assign to a community, we incorporate the use of a coassociation matrix, defined below.

3.1.4 Coassociation Matrix

For a given graph and community detection algorithm, we can construct a *coassociation matrix*, C, using the outputs of many runs of the algorithm on the graph. In our methodology, we use the same set of runs to calculate both the community-dependent features, and the coassociation matrix. Let r_{ij} be the number of runs for which nodes i and j are in the same community, and let R be the total number of runs. The value for the entry ij in the matrix is given by:

$$C_{ij} = \frac{r_{ij}}{R}$$

Intuitively, the coassociation matrix represents the proportion of runs for which two nodes are in the same community, for every pair of nodes.

In order to classify nodes as either “easy to cluster” or “hard to cluster”, we then calculate the entropy of each node from the coassociation matrix as follows:

$$E_i = \frac{\sum_j p_{ij}}{N}$$

where N is the number of nodes and p_{ij} is defined as follows:

$$p_{ij} = \begin{cases} -C_{ij}\log_2(C_{ij}) & \text{if } C_{ij} > 0 \\ 0 & \text{if } C_{ij} \leq 0 \end{cases}$$

This formulation means that if C_{ij} is close to 0 or 1, i.e. if nodes i and j are regularly in either the same or different communities, then there is not a large contribution to E_i for node j as p_{ij} is close to 0. However, if C_{ij} is somewhere between 0 and 1, i.e. if nodes i and j are sometimes in the same community and sometimes not, then p_{ij} is a larger value and makes a greater contribution to E_i. This means that the more node i switches between communities, the higher E_i will be, as it shares a community with different nodes on each run, resulting in various values of C_{ij} that are far from either 0 or 1. So a high value of E_i represents a node which is not easily clustered into a single community, whereas a low value of E_i represents a node which is clustered into the same community most of the time.

Unfortunately, these entropy values are not as intuitively understood as the raw coassociation matrix entries as they do not directly represent proportions. Thus, it is not as simple to label nodes as “easy to cluster” or “hard to cluster” directly from their entropy values as it is to label pairs as “same community” or “different community” directly from the coassociation matrix. Instead, once every node is assigned an entropy, we use one-dimensional k-means clustering (with $k = 2$ clusters) to separate nodes into two training classes: those with low entropy belong to the “easy to cluster" class, and those with high entropy belong to the “hard to cluster" class. Intuitively, these correspond to nodes which are often assigned to the same community by the algorithm and those which are often assigned to different communities.

3.1.5 Summary

Our aim is to identify human-interpretable graph features which relate to the community membership determined by a community finding algorithm. In order to select the more informative

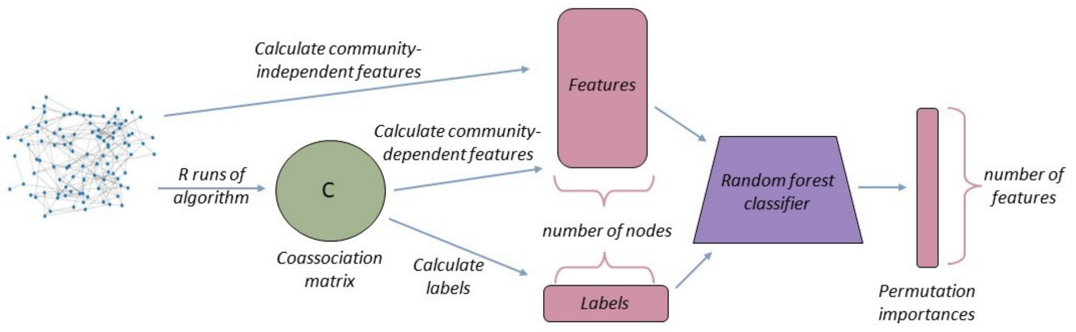


Figure 3.3: Experiments for determining explainable social network analysis metrics in the node feature experiment. A similar methodology is applied to the node-pair experiment. After R runs of the algorithm, a coassociation matrix is constructed encoding how often two nodes are classified in the same community. Feature values are computed and provided as input to a random forest classifier to determine permutation importance, shown here, or Shapley values. The distributions of the importance scores can be compared across all graphs to identify explainable metrics.

features from a predefined longlist of candidates, we define two simple binary classification problems: one for node-level features, where we will predict a node's ease of assignment to a community; and one for node-pair-level features, where we will predict whether the two nodes belong to the same community or not. We will then find the permutation importance and Shapley values of each feature from our model to identify which features provide the most information about the output label.

3.2 Methodology

Our experiments take place on more than one graph, μ value (i.e. the degree of community mixing, where a lower μ means more well-separated communities), algorithm, and even classification task. Having several independent variables enables us to answer the following research questions:

> *RQ1: Do the most informative node features depend on the community finding algorithm used?*

> *RQ2: Do the most informative node-pair features depend on the community finding algorithm used?*

RQ3: How do the most informative node features vary with the degree of community separation, as defined by the mixing parameter, μ?

RQ4: How do the most informative node-pair features vary with the degree of community separation, as defined by the mixing parameter, μ?

RQ5: In all cases, what are the most predictive features?

Although we did not form a strong hypothesis for the latter three questions, we hypothesise that the most predictive features would vary by algorithm.

H1: The most informative node and features will depend on the community finding algorithm.

H2: The most informative node-pair and features will depend on the community finding algorithm.

It is important to note that our goal here is to identify predictive features that relate to the community membership according to an algorithm of interest, such that these can be used in future explanatory visual analytics systems. We demonstrate this on three commonly-used algorithms, with an aim to discover features that can be used to explain them. Our methodology presented here does not itself explain the outputs of the algorithms, and can be applied to other algorithms in the future. Although the informative features may depend on the chosen data, we have chosen to use LFR data as it allows us to generate a large dataset where we can find the most predictive features across a wide number of graphs. The LFR data is designed to mimic real world graphs, so we hope that the important features will translate to real applications, though verifying these results on individual real world graphs will be the subject of future work. In order to answer our research questions, we now present our experimental and statistical methodology. This methodology is also illustrated in Figure 3.3.

3.2.1 Experimental Methodology

We test our approach using three popular methods to detect community structure, each based on different concepts (for example, we only use one modularity optimization algorithm) and explained in further detail in Section 2.1.2:

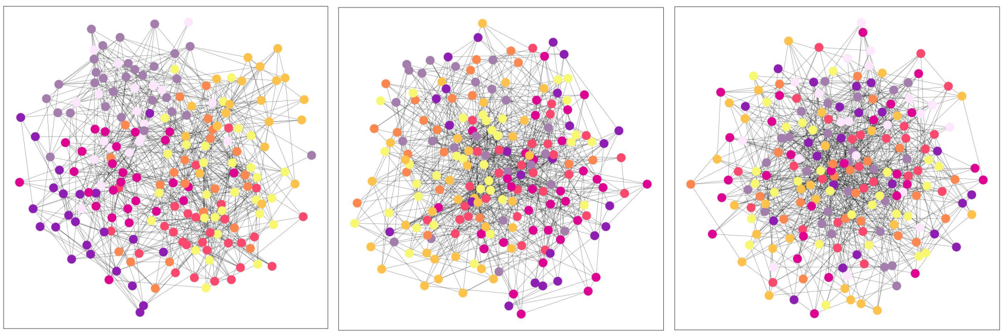
1. **Infomap** [137], also known as the map equation, uses an information-theoretic approach, where nodes are represented by codewords composed of two parts, the first of which is provided

by the community it belongs to. The community memberships are optimised by minimising the average code length describing random walks on the network.

2. **Louvain** [13], is a modularity optimization approach which involves two steps. Firstly, the common modularity objective is optimized at a local level to create small communities. Next, each small community is treated as a single node and the first step is repeated. By following this agglomerative process, a hierarchy of communities is constructed.

3. **LPA**, the label propagation algorithm proposed by Raghavan et al. [130], assumes that nodes should belong to the same community as most of their neighbours. To begin, each node is initialised with a unique community and then these labels are then iteratively propagated through the network. After each iteration, a node receives the same label as the majority of its neighbours. Once this process is complete, nodes sharing the same label are grouped together as communities.

When constructing our networks, we selected μ values of $0.2, 0.3$, and 0.4. This parameter controls the level of separation or mixing between communities, where the higher the value of μ, the less easy it is to distinguish between different communities. These values were chosen as, in preliminary experimentation, we found that lower values of μ resulted in communities that were so well-separated that all of the algorithms produced the same community clustering every time. By contrast, higher values of μ resulted in communities that were too hard to cluster; indeed, we already found that LPA failed to generate meaningful communities for a μ value of 0.4, clustering every node into its own community.



A. $\mu = 0.2$ B. $\mu = 0.3$ C. $\mu = 0.4$

Figure 3.4: Example graphs with 200 nodes at the three μ values. Communities shown with colour. Increased mixing parameter increases the prevalence of edges between communities.

At each value of μ, a set of graphs, Γ, are generated before any experiments take place. This set of graphs is the same size, $|\Gamma|$, for each value of μ. In order to match the hyperparameters used by Lancichinetti et al [86] in the original LFR benchmark paper, which were chosen to mimic the structure of graphs seen in the real world, we use the LFR generator in NetworkX to generate networks with 1000 nodes of average degree 20 and maximum degree 50. We set the hyperparameters τ_1 and τ_2 to 3 and 2 respectively. These hyperparameters correspond to the power law exponent for the degree distribution and the community size distribution in the created graph, respectively.

Each experiment is then defined by three categories: the μ value; the community detection algorithm; and the feature type (node vs node-pair). This results in 18 possible experiments from the 3 algorithms, 3 mixing parameters and 2 feature types. Data from the $|\Gamma|$ graphs at the relevant value of μ are used for the experiment. For each μ-algorithm-feature type combination, the following procedure is then performed.

Firstly, the algorithm is run 1000 times on each of the $|\Gamma|$ graphs. Using these runs, any community-dependent features are calculated, along with the coassociation matrix. Features which are community-independent are also calculated at this stage, although they do not depend on the runs. The nodes or pairs-of-nodes must then be labelled according to the binary classification problem. The labelling procedures are described separately for the two feature-types in the relevant experiment sections.

Now, for each of the graphs of the experiment, we have a dataset of either nodes or pairs of nodes, each member of which is labelled and has a list of feature values. A random forest with 100 trees is then trained to classify the dataset for the specific graph. Note that we chose to use random forests as the goal of our work is to generate information which can be used to provide trustworthy explanations. Random forests are a more inherently interpretable model than neural networks, for example, so do not add as much opacity to our methodology.

During training we use 5-fold cross-validation and repeat this for 10 training runs. A permutation importance or Shapley value is calculated for each node or node-pair feature after each of the 50 runs, using the held-out test data. At the end of the 50 cross-validation runs, a mean average of the 50 gathered importance scores is taken for each node or node-pair feature. This gives us its final importance score as generated by this graph. Overall, this results in $|\Gamma|$ importance values for each feature. The full experimental methodology for node features is represented in Algorithm 3.1. For node-pair features, the algorithm is identical, looping over node-pairs instead of nodes.

Algorithm 3.1 Node Features Experiment

```
Input: Graphs $\Gamma_\mu$ of mixing parameter $\mu$
Output: Matrix of importance scores, $P$, with shape $|\Gamma_\mu| \times f$ where $f$ is the number of features
 1: for graph $G = (V,E)$ in $\Gamma_\mu$ do
 2:     Run chosen community finding algorithm 1000 times
 3:     Calculate coassociation matrix, $C_G$
 4:     Calculate feature vector $X_G = (x_{G,1},\dots,x_{G,f}))$
 5:     for node $v$ in $V = \{1,\dots,n\}$ do
 6:         Calculate label $y_v$ according to $C_G$
 7:     end for
 8:     for training run $r = 1$ to 50 do
 9:         Train random forest using $(X_G, Y = (y_1,\dots,y_v))$
10:         Calculate importance scores for each feature
11:     end for
12:     Calculate mean importance score across the 50 training runs for each feature, creating
      a vector of importance scores for this graph $P_G = (p_{G,1},\dots,p_{G,f})$
13: end for
14: Output final importance score matrix $P = (P_1,\dots,P_{\Gamma_\mu})$
```

3.2.2 Statistical Methodology

For both experiments above, we have distributions of our features over the runs of the experiment. These distributions can be compared to determine statistical significance of the difference between them, and the size of this difference, in order to identify the features of interest. This statistical analysis and the final conclusions drawn are specific to the μ-algorithm-feature type combination of the experiment.

In order to develop an appropriate statistical methodology, we performed a pilot study using 20 graphs at each μ value (giving 60 graphs in total). The pilot study was performed before all main experiments using permutation importance values, and for this reason we report results for our main experiment with permutation importance values first.

In the pilot study experiments, the 20 graphs gave us 20 values of permutation importance for each feature, on which we carried out Shapiro-Wilk tests. In this pilot study, 67% of the features across all algorithms, feature types and μ values were normally distributed, so we started with a normal assumption. On this basis, the statistical methodology would be as follows:

1. Perform a power analysis with a normal assumption to determine the value of $|\Gamma|$ required to draw statistically significant conclusions.

Number of Communities			
	Mean	Median	Std.
Infomap, mu 0.2	40.65	41.00	2.45
Infomap, mu 0.3	39.93	40.00	2.33
Infomap, mu 0.4	35.76	36.00	2.65
Louvain, mu 0.2	34.45	34.00	1.83
Louvain, mu 0.3	29.56	29.00	1.79
Louvain, mu 0.4	24.20	24.00	1.61
LPA, mu 0.2	39.14	39.00	2.54
LPA, mu 0.3	36.33	36.00	3.06

Table 3.1: The numbers of communities identified by each algorithm on graphs with 1000 nodes. The mean, median and standard deviation are calculated across all 120 graphs in each case.

2. Carry out the experiments to obtain $|\Gamma|$ values for each feature of each experiment.

3. Confirm with a repeat of the Shapiro-Wilk tests that these $|\Gamma|$ values are indeed normally distributed in the majority of cases.

4. If the distributions are normal, perform pairwise t-tests with Bonferroni-Holm corrections using these values. Otherwise, perform pairwise Wilcoxon tests with Bonferroni-Holm corrections using these values.

Power analysis was conducted with the following parameters: Cohen's effect size of 0.3, significance level of 0.05, and a power of 0.9. The power analysis concluded 119 graphs were necessary for our experiment, which we rounded to 120. At this stage, we generated 360 new graphs (120 at each μ level) for our experiment.

The application of this methodology to our new data set revealed that the distributions of metric values were not normally distributed. Therefore, to determine significance, pairwise Wilcoxon tests with Bonferroni-Holm correction were applied to our permutation importance data to determine the significant results.

3.3 Exp. 1: Node Feature Experiment

Once the experimental data was collected, the 0.4-LPA-node and 0.4-LPA-node-pair experiments were omitted. This is because LPA clustered a majority of nodes into one large community at this μ value, generating features and labels that were not suitable for our experiments.

NMI Statistics			
	Mean	Median	Std.
Infomap and Louvain, mu 0.2	0.977	0.978	0.008
Infomap and Louvain, mu 0.3	0.949	0.949	0.012
Infomap and Louvain, mu 0.4	0.883	0.884	0.021
Infomap and LPA, mu 0.2	0.991	0.992	0.007
Infomap and LPA, mu 0.3	0.962	0.966	0.022
Louvain and LPA, mu 0.2	0.970	0.970	0.012
Louvain and LPA, mu 0.3	0.922	0.925	0.025

Table 3.2: Statistics on the normalised mutual information scores for two algorithms on graphs of the same μ value. For each row of the table, 1000 pairs of partitions were uniformly randomly chosen for each graph. As there are 120 graphs for each μ value, this means 120,000 values contribute to each statistic.

Essentially, LPA was unable to recognise community structure at this high degree of mixing. All other experimental data is reported.

Tables 3.1 and 3.2 respectively show statistics on the number of communities detected by each algorithm across graphs of a common μ value, and normalised mutual information (NMI) scores comparing the performance of pairs of algorithms. NMI is a measure used to evaluate the similarity between two clusterings of data. It is based on the concept of mutual information, which measures the amount of information gained about one random variable by observing another random variable. NMI scores are commonly used in clustering tasks, especially when comparing the performance of two clustering algorithms. They provide a quantitative measure of the similarity of the two clusterings. The higher the NMI score, the more similar the two are.

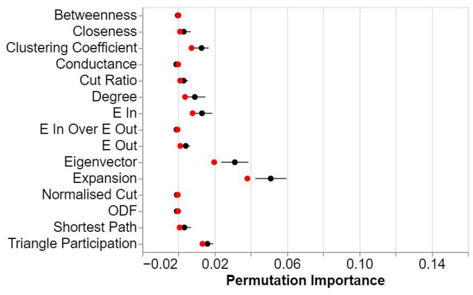
In Table 3.1, we can see that communities range in number from 24 to 40, resulting in a mean community size between 25 and 45 nodes. In reality, sizes of communities created using the LFR generator follow a power law, so many will be much larger or smaller than the mean. In the second table, NMI scores are generally high in all cases, although decrease as the μ value increases, as one might expect. Overall, the similarity between the outputs of the different alogrithms suggest that there are regions of the networks which are very stable, possibly communities with very large, stable cores, and that the nodes which frequently change across multiple algorithmic runs could be single nodes on the periphery of these large communities, or belong to the much smaller communities.

Proportion of “Hard to Cluster" Nodes			
	Mean	Median	Std.
Infomap, mu 0.2	0.087	0.060	0.072
Infomap, mu 0.3	0.076	0.067	0.044
Infomap, mu 0.4	0.192	0.180	0.070
Louvain, mu 0.2	0.169	0.152	0.101
Louvain, mu 0.3	0.199	0.175	0.120
Louvain, mu 0.4	0.227	0.221	0.081
LPA, mu 0.2	0.242	0.250	0.117
LPA, mu 0.3	0.253	0.250	0.115

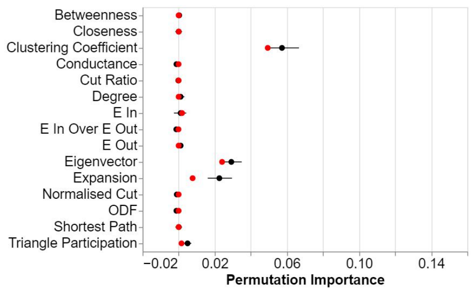
Table 3.3: Statistics on the proportion of nodes labelled as “hard to cluster" after running each algorithm on graphs of varying μ value. The mean, median and standard deviation are calculated across all 120 graphs in each case.

3.3.1 Experiment

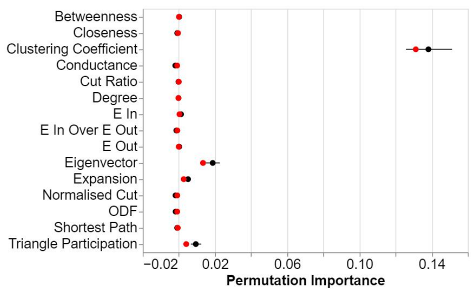
The classification labels for the node feature experiments are calculated for a single graph as follows. The entropy of each node is calculated from the coassociation matrix of the current graph, and k-means clustering of these entropy values is performed to separate the nodes into “easy to cluster” and “hard to cluster” nodes. However, using this process, we have a very low proportion of “hard to cluster" nodes. The proportion of nodes labelled as “hard to cluster" are reported in Table 3.3. For low mixing parameter values, this can be as low as 9%. This reinforces the finding from Tables 3.1 and 3.2 that there are large, central cores to the communities with a small number of nodes on the periphery or in smaller communities, as the number of nodes which are “hard to cluster” is small, suggesting that more nodes belong to stable regions. However, the proportion of “hard to cluster" nodes can rise to as high as 25% with an increased mixing parameter, indicating that this is a distinct class of nodes. Due to the low proportions of “hard to cluster" nodes, we use undersampling. Rather than undersampling randomly, we propose using the “easiest" nodes to cluster (those with the lowest entropy, i.e. those which least frequently flip between communities) until the number of “hard" nodes is 75% that of the number of “easy" nodes. Although this still results in some imbalance, using a smaller number of “easy” nodes would result in a very small dataset. Using this *strategic undersampling* method enables us to identify node features which distinguish between truly separate classes, rather than distinguishing between nodes with an entropy either side of the arbitrary cut-off generated by the k-means clustering.



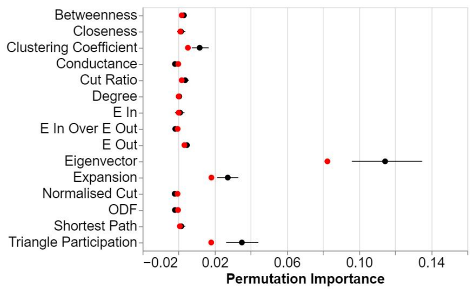
A. Infomap, $\mu = 0.2$



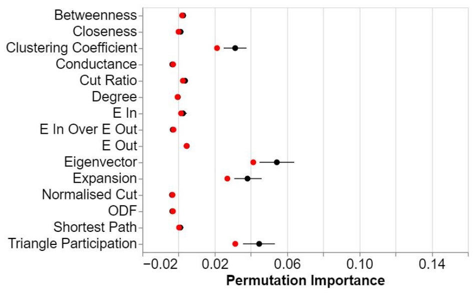
B. Infomap, $\mu = 0.3$



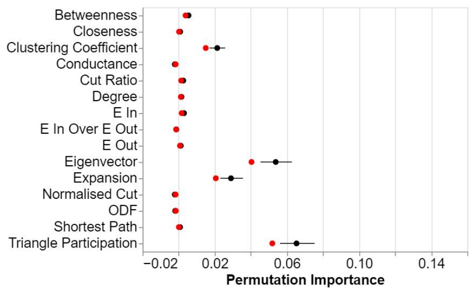
C. Infomap, $\mu = 0.4$



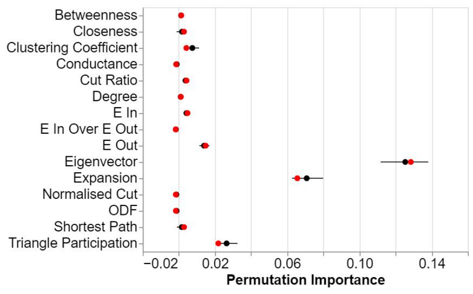
D. Louvain, $\mu = 0.2$



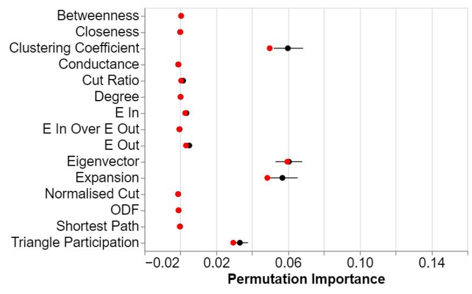
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$



G. LPA, $\mu = 0.2$



H. LPA, $\mu = 0.3$

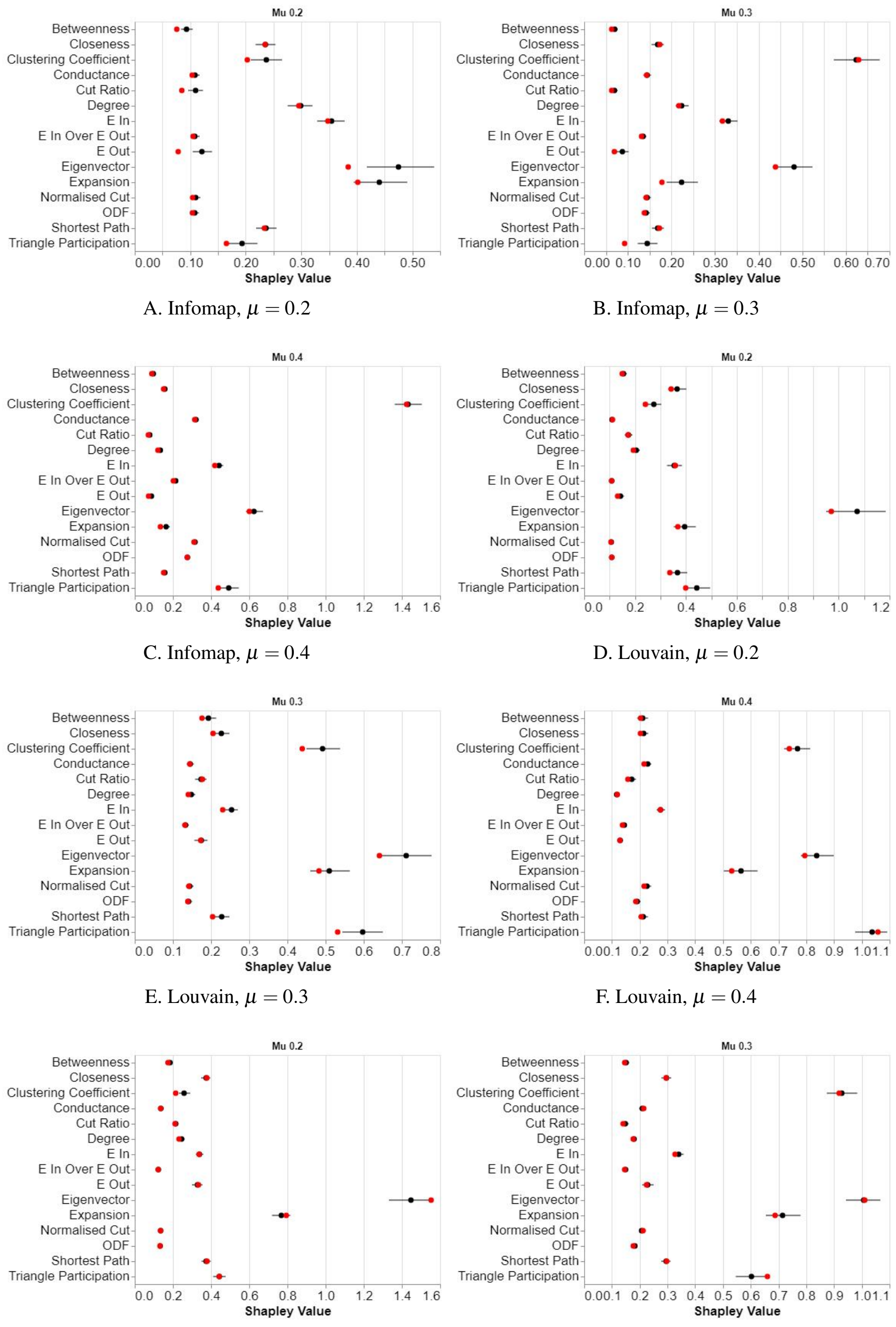
Figure 3.5: Results of the node feature experiments. Plots are of permutation importance of the metrics. Mean indicated as a black dot and median as a red dot. Lines indicate 95% bootstrapped confidence intervals.

3.3.2 Results

For our results plots, we visually represent the distribution of importance scores across all 120 graphs. The black and red circular marks in the plots represent the mean and median of these values, while the black bar represents a non-parametric bootstrap of the 95% confidence interval. For the experiments with permutation importance scores (displayed in Figure 3.5), we see that four of the features consistently have a non-zero permutation importance: clustering coefficient, eigenvector centrality, expansion and triangle participation. The same four features are frequently among the most important in the Shapley value experiments (displayed in Figure 3.6. This consistency confirms that the choice of importance score has not significantly affected the results. The results of Wilcoxon tests can be found in the heatmaps in Figure 3.7. These tests were run for every pair of features to identify whether the distributions of the permutation importance values for the two features were significantly different (subject to Bonferroni-Holm corrections). The colour of the cell in the heatmap represents whether there was a significant difference or not, with a key in the upper right. A significance of 0 represents no significant difference; a significance of 1 represents a significant difference. Across all experiments at all μ levels, our pairwise Wilcoxon tests confirmed that these four features, highlighted in both the permutation importance and Shapley value experiments, were significantly more important than the rest of the features, with the following exceptions:

- For Louvain at $\mu = 0.2$, clustering coefficient was not significantly different from betweenness centrality, cut ratio, or E_{out}.
- For Infomap at $\mu = 0.2$, clustering coefficient was not significantly different from degree, E_{in}, E_{out} or shortest path. Triangle participation was not significantly more important than degree or E_{in}.
- For Infomap at $\mu = 0.3$, clustering coefficient was not significantly different from closeness centrality, degree, E_{in} or shortest path.
- For LPA at $\mu = 0.2$, clustering coefficient was not significantly different from any of betweenness centrality, closeness centrality, cut ratio, degree, E_{in} or average shortest path.

These exceptions align with what can be seen qualitatively: at the lowest μ level, i.e. the lowest level of community mixing, some other features appear to be important such as degree,



A. Infomap, $\mu = 0.2$

B. Infomap, $\mu = 0.3$

C. Infomap, $\mu = 0.4$

D. Louvain, $\mu = 0.2$

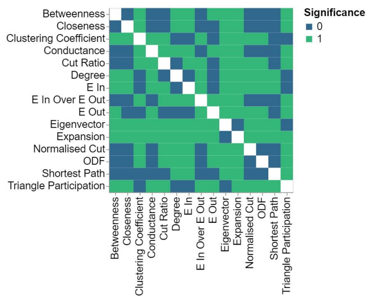
E. Louvain, $\mu = 0.3$

F. Louvain, $\mu = 0.4$

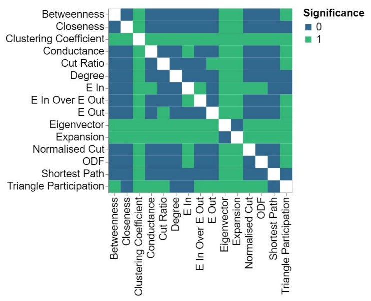
G. LPA, $\mu = 0.2$

H. LPA, $\mu = 0.3$

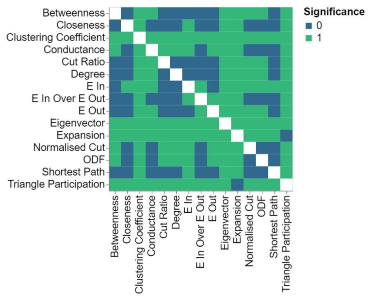
Figure 3.6: Results of the node feature experiments. Plots are of Shapley values of the metrics. Mean indicated as a black dot and median as a red dot. Lines indicate 95% bootstrapped confidence intervals.



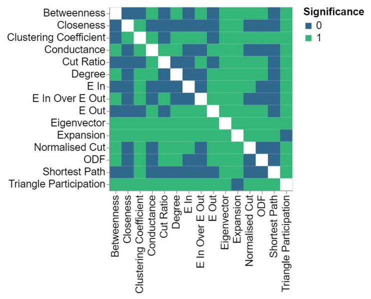
A. Infomap, $\mu = 0.2$



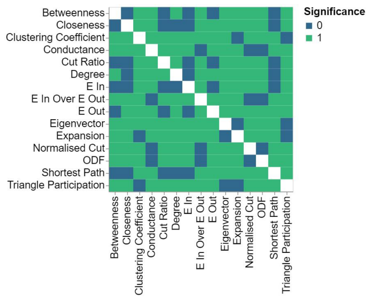
B. Infomap, $\mu = 0.3$



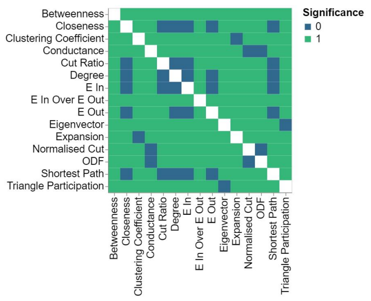
C. Infomap, $\mu = 0.4$



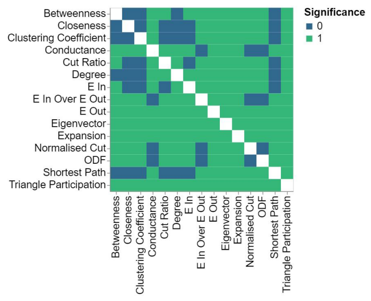
D. Louvain, $\mu = 0.2$



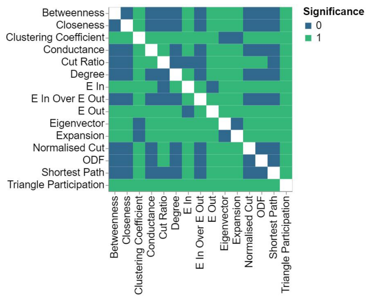
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$



G. LPA, $\mu = 0.2$



H. LPA, $\mu = 0.3$

Figure 3.7: Heatmaps showing significance of Wilcoxon tests on the node feature experiments. The colour of the cell in the heatmap represents whether there was a significant difference or not, with a key in the upper right. A significance of 0 represents no significant difference; a significance of 1 represents a significant difference.

E_{in} and perhaps even closeness centrality, cut ratio, E_{out} and average shortest path. However, the effect size for all of these is much smaller than for the four most prominent features, and their significance vanishes at the two higher μ levels, i.e. for more mixed communities.

3.3.3 Discussion

With respect to our research question RQ1 and in contradiction to our hypothesis H1, we observed that the four prominent features for predicting whether a node was difficult to classify did not depend on the algorithm used: clustering coefficient, triangle participation, eigenvector centrality, and expansion. In relation to the first two, this is not unexpected as these characteristics have previously been shown to be broadly indicative of good community structure [56]. One could conjecture that nearly all community finding approaches would try to preserve cliques (accounting for the importance of clustering coefficient and triangle participation). In fact, cliques have often been used as seeds for constructing larger communities in the context of quite different community detection algorithms [89, 122]. Meanwhile, it seems reasonable that a node with many links to nodes in other communities relative to the number of nodes in its own community would be harder to classify, as it likely lies on the periphery of its community, close to one or more others (accounting for the importance of expansion).

At a surface level, the prominence of eigenvector centrality is more surprising, especially given the level of its performance. This centrality measure has similarities to PageRank [121], where high values correspond to nodes at short distances from many high degree nodes. Nodes within a community's core are more likely to have high degree and to be a short distance from other high degree nodes, with edges that connect other nodes within the community. The relationship between eigenvector centrality and regions of high density within the core versus periphery of a network was recently highlighted by Bienenstock et al. [65]. Thus, in our case high values of eigenvector centrality might correspond to an increased chance that this node forms a part of the stable community core, rather than being an unstable node on a community's periphery which the algorithm therefore clusters differently on consecutive runs.

The results of our experiment with regards to changing mixing parameters μ (RQ3) indicate that these four features remain prominent. There is some evidence as well that the other features diminish in prominence as μ increases and the communities become more difficult to find. Thus, the same features are involved for explaining why a node is part of a stable core or changes communities between runs and all become statistically significant at higher mixing parameters.

Further investigation is required to find out why the important features consistently performed the best and the relative differences between them across other community finding algorithms.

3.4 Exp. 2: Pairwise Community Membership

As mentioned in section 3.3, the 0.4-LPA-node-pair experiment is omitted here as LPA classified the entire graph as one community on a number of occasions at the higher mixing parameter level.

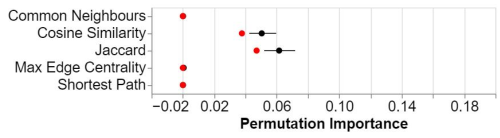
3.4.1 Experiment

As with the node feature experiments, labelling all pairs directly as "same community" or "different community" results in imbalanced classes. However, we have vastly more data for the pairs of nodes than for the single nodes. Therefore, we propose undersampling both classes by randomly selecting the same number of "same community" and "different community" pairs from the available data. We choose to undersample randomly here rather than "strategically" since there are no pairs of nodes close to the threshold of 0.5 between "same" and "different" community, (i.e. the point where the pair of nodes belong to the same community after exactly half of the runs), but choosing the highest and lowest values leads to a classification problem which is *too* easy, therefore making it less informative when differentiating the most useful features from the rest. We select 1000 training examples for each class.

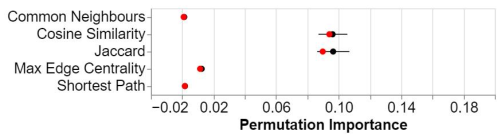
3.4.2 Results

As with our previous experiment, we found that two features were consistently important across the three community finding algorithms, for both permutation importance and Shapley values: cosine similarity and the Jaccard coefficient (displayed below in Figures 3.8 and 3.9). We also found that the maximum edge centrality along the shortest path became more important at higher mixing parameter levels. This varied a little by algorithm; for Louvain it became important even at the lowest mixing parameter level of 0.2, however for Infomap and LPA it didn't become important until the mixing parameter of 0.3.

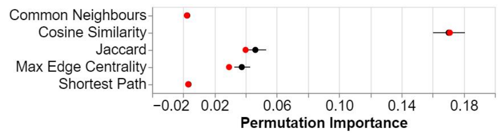
The complete set of Wilcoxon significance tests are reported in Figure 3.10. The pairwise Wilcoxon tests confirmed that all three were significantly different across all experiments, including for max edge centrality at the $\mu = 0.2$ level despite the small effect size.



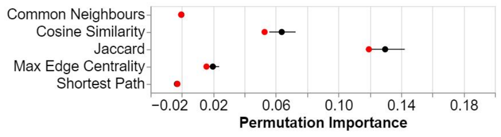
A. Infomap, $\mu = 0.2$



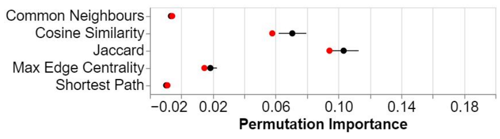
B. Infomap, $\mu = 0.3$



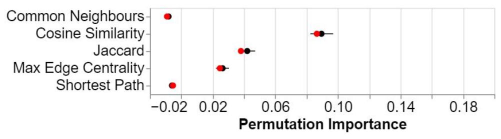
C. Infomap, $\mu = 0.4$



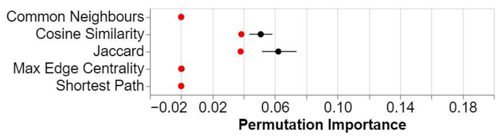
D. Louvain, $\mu = 0.2$



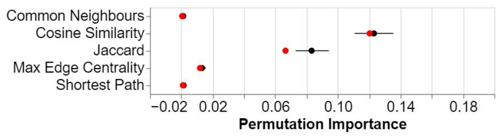
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$



G. LPA, $\mu = 0.2$



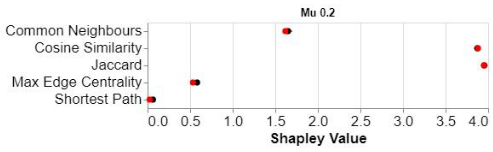
H. LPA, $\mu = 0.3$

Figure 3.8: Results of the pair feature experiments. Plots are of permutation importance of the metrics. Mean indicated as a black dot and median as a red dot. Lines indicate 95% bootstrapped confidence intervals.

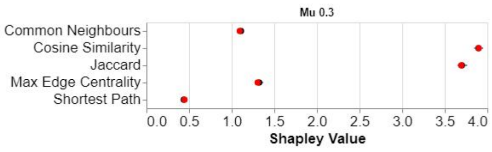
3.4.3 Discussion

In contradiction to hypothesis H2, all algorithms performed similarly, with the most important features being Jaccard and cosine similarity. Both features compare the neighbourhoods of the selected nodes. Their importance is supported by the *local consistency assumption* [191], frequently discussed in the context of instance-based classification, which asserts that neighboring instances will frequently share similar characteristics. In this case, the local consistency assumption would suggest that there are similarities in the properties of neighbouring nodes. In the context of unsupervised community detection, this corresponds to neighbouring nodes belonging to the same community, rather than having the same class label. This result is also congruous with the result of the first experiment, which found similar features to be important.

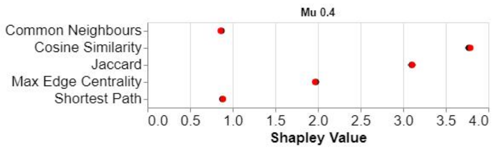
In response to RQ4, maximum edge centrality proved increasingly important as the μ-level of the generated data increased. This measure, which is the maximum edge centrality measure along the shortest path between the two nodes, could be indicative of important edges that



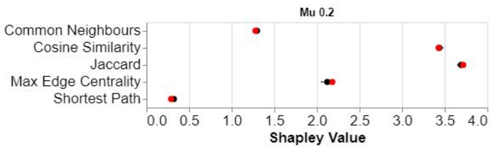
A. Infomap, $\mu = 0.2$



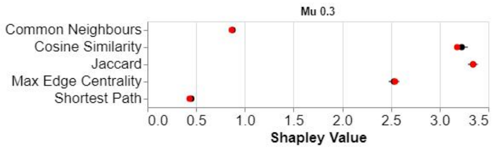
B. Infomap, $\mu = 0.3$



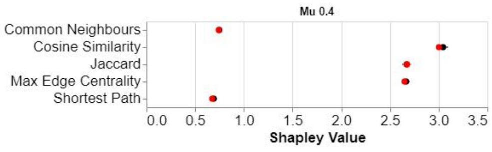
C. Infomap, $\mu = 0.4$



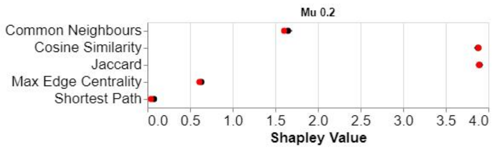
D. Louvain, $\mu = 0.2$



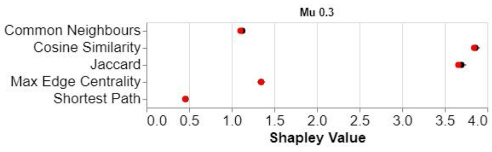
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$



G. LPA, $\mu = 0.2$

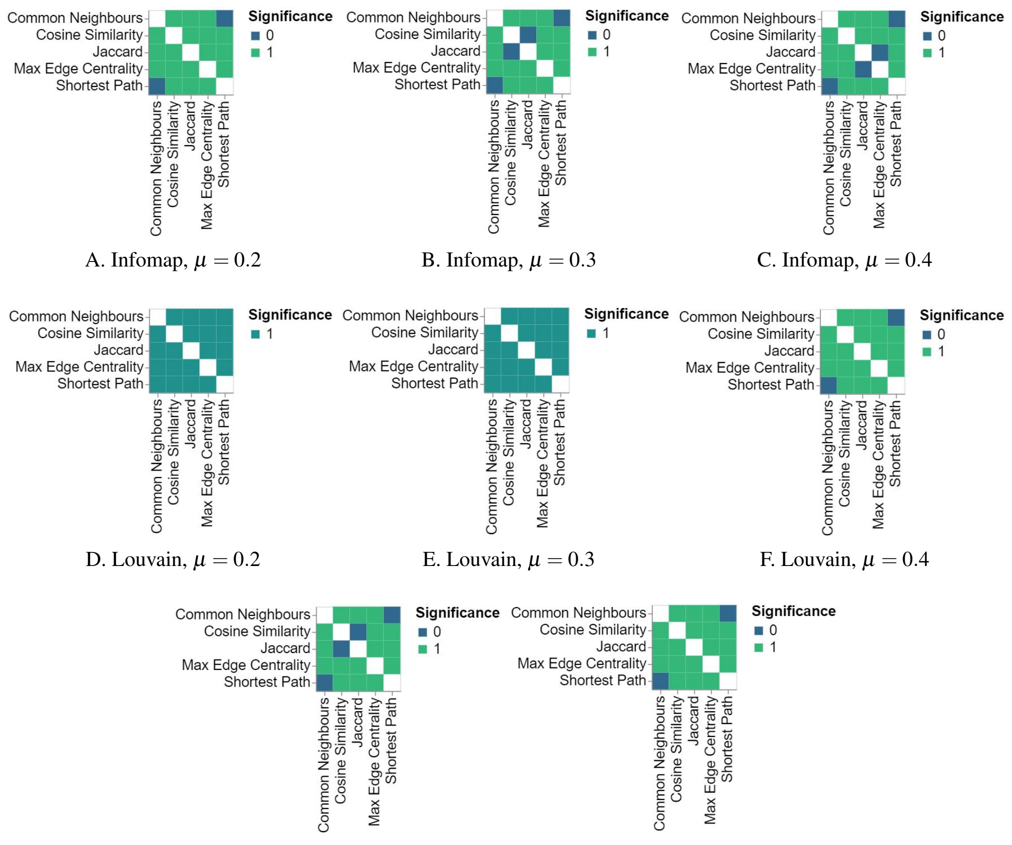


H. LPA, $\mu = 0.3$

Figure 3.9: Results of the pair feature experiments. Plots are of Shapley values of the metrics. Mean indicated as a black dot and median as a red dot. Lines indicate 95% bootstrapped confidence intervals.

bridge two communities, i.e. the weak ties, and has been used in the past for divisive methods of community detection [51]. The increased importance at higher values for the mixing parameter could be explained by how important local information is to determining if two nodes are within the same community. If μ is low, communities are well separated and local information almost completely describes if two nodes are in the same community. However, as μ increases, the value of local information decreases in importance. Instead, global information, such as determining if the path between the two nodes likely contains an edge that lies between communities, becomes critical in determining whether nodes belong to the same community.

Further investigation is required to find out why these features consistently performed the best and the relative differences between them across other community finding algorithms.



A. Infomap, $\mu = 0.2$ B. Infomap, $\mu = 0.3$ C. Infomap, $\mu = 0.4$

D. Louvain, $\mu = 0.2$ E. Louvain, $\mu = 0.3$ F. Louvain, $\mu = 0.4$

G. LPA, $\mu = 0.2$ H. LPA, $\mu = 0.3$

Figure 3.10: Heatmaps showing significance of Wilcoxon tests on the pair feature experiments. The colour of the cell in the heatmap represents whether there was a significant difference or not, with a key in the upper right. A significance of 0 represents no significant difference; a significance of 1 represents a significant difference.

3.5 Discussion

For both experiments, our hypothesis was that the important metrics would be dependent on the community finding algorithm, however, the same metrics were identified consistently. This indicates that there are common metrics that can be used to explain these phenomena, at least when producing explanations on the same dataset for the three algorithms tested. As our study is limited to networks generated by the LFR algorithm, the common metrics of importance could be indicative of structure produced by this method. Further experiments on other datasets may reveal variations in the metrics of importance. If the community finding algorithm can be taken into account, then these important metrics can also be weighted in a way that is in line with the algorithm and degree of mixing of the communities.

Although there were benefits to our use of synthetic LFR data, these networks are ultimately an approximation for real data. As discussed previously, the use of real data for this analysis would have been tricky due to the lack of large datasets of networks with consistent structure over which we could draw statistically significant conclusions. Additionally, we would not have been able to vary parameters such as the mixing parameter μ to observe their effect on the results. However, even in the case where the metrics of greatest importance are heavily dependent on the dataset, the methodology presented here could beneficially be applied to new settings in order to gain insight into complex networks relevant to different applications, such as social, biological, and financial [8, 53] networks.

Consensus clustering [85] is a way of dealing with nodes that are difficult to classify: run the community finding algorithm many times and determine the average result of these runs. Given that this study indicates that the metrics to determine if nodes are easy or hard to cluster by community finding algorithms are consistent across algorithm, the results of consensus clustering approaches could be augmented with these metrics to help determine which community these nodes should be clustered into. Also, values for these metrics could be used to seed the stable core of a community and then find other nodes that are less easy to cluster. This approach could lead to other partitioning algorithms or potentially overlapping community finding algorithms where “hard to cluster" nodes are partially contained by multiple communities. However, the effectiveness of such an approach would still need to be evaluated.

The studies that we present here suggest metrics used in social network analysis can be used to explain partitioning algorithms, though it could be of interest to determine if this extends to overlapping community finding [87,122], where algorithms can assign a single node to multiple communities.

3.6 Conclusion

In this chapter, we have presented a methodology and the results of an experiment to determine features that can be used to explain detected community structure in networks. These features were specifically chosen to be interpretable to a domain expert with some understanding of network analysis. The features introduced in this chapter have been defined for individual nodes or pairs of nodes, to understand their membership to specific communities.

In designing these experiments, we were motivated by problems in public health applications, where these social network analysis metrics may be used to understand phenomena such as social contagion [20, 164] and to plan interventions [163]. We envisage these results could be used with a visualisation system where the communities assigned by an algorithm can be explored by selecting individual nodes or pairs of nodes to understand their community assignment. When a node flips between different communities on consecutive runs of the community finding algorithm, important feature values such as those identified in this work could be visually reported and compared relative to other nodes in the same or in different communities for further study by an expert, contributing to an explanation for this node's community membership.

To extend the study presented in this chapter, we next consider the use of "community features" as another "level" of graph feature, where metrics are calculated for a whole community rather than an individual node or pair of nodes. The details of the chosen features, and results of experiments, are laid out in Chapter 4, which follows this one.

Chapter 4

Explainable Community Finding with Community Features

In this chapter, we further explore the application of explainable AI techniques to the field of social network analysis, continuing with our exploration of community finding algorithms. Our work in this area is once again motivated by applications in the public health domain, where community detection can be an important step in understanding structure in networks where social contagion occurs [20, 164]. Being able to explore this structure through easily understood social network analysis metrics allows experts to develop appropriate interventions [163]. While we have explored interpretable features calculated on nodes and pairs of nodes in the previous chapter, we now turn to features which can be defined on whole sets of nodes, such as those which comprise a community. When understanding social structure in the public health domain, information on both the individual level and the whole network level are highly relevant; the work of the previous chapter allows us to gain information about individuals in the network, in relation to their communities, while exploring metrics defined on larger groups of individuals would allow public health researchers to learn more about the bigger picture within the network.

Here, we adapt our methodology from the previous chapter to a new classification problem. The goal of this methodology is to identify predictive features that relate to the community membership according to an algorithm of interest, such that these can be used in future explanatory visual analytics systems. The methodology does not itself explain the outputs of the algorithms, but can be applied to any community finding algorithm to identify features that may be valuable in understanding the community membership. Although the informative features may depend on the data used in the experiments, we have chosen to use a large dataset of LFR data so that we can find the most predictive features across a wide number of graphs. The LFR data is designed to produce a similar structure to real world graphs, so we hope that the important features will translate to real applications, though verifying these results on individual real world graphs remains future work. We offer a brief recap of the steps in this methodology, which are as follows:

1. We first select a longlist of social network analysis metrics which we believe are inherently interpretable to users with social network analysis expertise. These are metrics which are applied to a specific type of network entity; in the previous chapter, we did this for individual nodes, and for pairs of nodes.
2. We then define a binary classification problem which distinguishes things in the network in an informative way. For individual nodes, we wanted to know which were easily assigned to a community, and which were not. For pairs of nodes, we wanted to know if they were in the same or different communities.
3. For each experiment, we then train random forests on this binary classification using our longlist of metrics as the input features, and use an importance score (in our work, this is permutation importance or a Shapley value, though others may be used) to identify which metrics from the longlist are most informative about our network entity.
4. We carry this out for a large number of networks to gain an understanding of the importance of our metrics on the classification problem in general, rather than for one particular graph.
5. Statistical tests, as well as a pilot study, allow us to identify which features are truly statistically significantly more informative than the others in our longlist.

In this new setting we define features which are evaluated on groups of nodes, which we refer to as the *community-level*. We then use these features to classify sets of nodes as either "real communities" or "fake communities". Our goal remains unchanged: to identify those features which are most informative in explaining the results of a given community detection problem, while being familiar and easily understood to social network analysis experts. These interpretable features can then be used in future work to develop visual analytics systems that enable a practitioner to understand the performance of the algorithms being applied to their problems.

Once again, we focus on the application of our methodology to explaining the outputs of the Louvain [13], Infomap [137] and label propagation [130] algorithms, and use synthetic graphs generated by the LFR benchmark [86] for evaluation. As before, the use of the LFR generator allows us to vary the value of a mixing parameter, μ, such that we can explore the effect of community separation on the importance of the various features. Low values of μ correspond to well-separated communities, while higher values lead to communities that are harder to identify. In all these ways, the experiments described herein are a direct extension of the ones in the previous chapter.

In making this extension to our experiments, we envisage that a future visualisation system which makes use of these insights could provide beneficial understanding not only for specific nodes, but also at a higher level, aiding insight into the reasoning behind global community structure. Augmenting the same system with this additional information will allow the contrasting of information about the whole network with information about communities, and how specific nodes interact with those communities. This contrasting of global and local information has been a focus of previous research applied to machine learning [113].

As with our previous experiments, we conclude with an analysis of the features which were most predictive in our classification task. We once again found that there were similar results across all algorithms. In this case, the cut ratio and internal-external metric, defined in section 4.1.3, were consistently the most informative. However, as the number of inter-community edges increases, relative betweenness becomes increasingly important. This chapter is based on work presented at the Springer conference Complex Networks & their Applications (CNA) 2021 [139].

4.1 Methodology

In order to identify the interpretable features which can distinguish between real communities and other sets of nodes, we propose the following methodology, adapted from the one used in the previous chapter. We perform a comprehensive set of experiments in conjunction with different community finding algorithms, applied to networks with increasing levels of community mixing. Specifically, we once again make use of the LFR benchmark generator, using different network μ values to vary the level of mixing. For each μ value, a large set of synthetic graphs are generated. Then for each experiment, we run the chosen community finding algorithm on the set of graphs at the current μ value. For each graph, we perform 1000 runs of the algorithm, and obtain the set of unique communities found across these 1000 runs. One node may appear in many different communities within this set, as the community structure may have been identified differently across different runs of the algorithm. However, each community will appear only once in the dataset as we discard duplicates of the same set of nodes. We then calculate our longlist of features for these communities. This gives us our set of candidate features for our examples labelled as “real communities".

Then, we use a rewiring process to adjust the original network structure for each synthetic graph. The feature values are recalculated for each community on the rewired graph, giving us a set of features for our examples labelled as “fake communities". We call these “fake communities” as they are sets of nodes which were unlikely to have been clustered into a community by the algorithm, given the adjusted structure of the graph. Although the set of nodes in the community remains the same, the structure of the community has changed in the rewiring, resulting in new values for the features. However, due to the one-to-one mapping between the “real community" examples and the “fake community" examples, we can guarantee balanced classes for the classification problem.

Given our labelled examples of “real communities" and “fake communities" generated as described above, in the next step we train a random forest classifier to distinguish between the two classes. The inputs to the classifier are the feature values as previously calculated. From the resulting random forest model, we can extract permutation importance values [19] and Shapley values for each of the input features and perform a statistical analysis to identify which are the most informative. As before, we use the high-speed estimation approach, TreeSHAP [98], to approximate the Shapley values due to their computational complexity in the original closed-form definition [148]. Similarly to our previous experiments, permutation importance values were preregistered in our statistical analysis report and used for our pilot study (described in the later section 4.1.6), so these are reported as the main results and we perform further statistical analysis on these alone. However, we include results using Shapley values as well to verify the results from the permutation importance experiments, showing that they are largely invariant to the explainability metric used. Henceforth we will refer to both permutation importance and Shapley values as “importance scores”.

4.1.1 Graph Generation

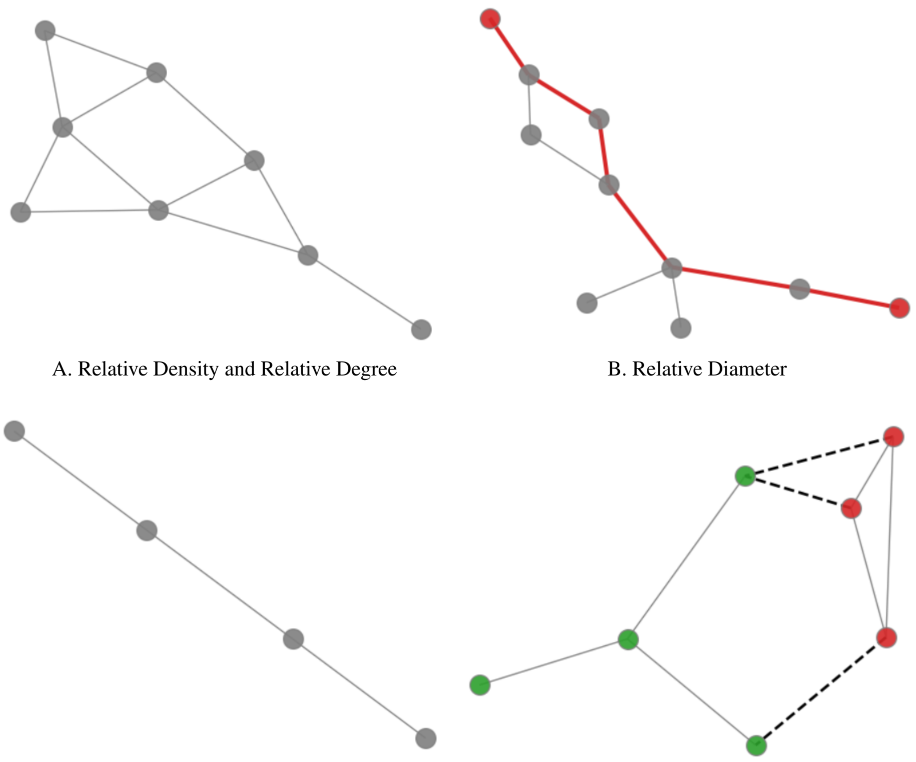
As in the previous chapter, we have chosen to use the LFR generator in NetworkX [55] as this allows us to create a large dataset of networks which mimic the structure of real data, which is itself hard to come by. Ideally, it would be beneficial to carry out similar experiments on real data in future work. An additional benefit of the LFR generator is that we can generate networks of differing levels of community mixing by varying the mixing parameter, μ, so that we can assess how the most informative metrics are dependent on this factor. When generating the graphs, we use the following hyperparameters: numbers of nodes 1000; τ_1 and τ_2 3 and 2; average degree 20; maximum degree 50. τ_1 and τ_2 correspond to the power law exponent for the degree distribution and the community size distribution in the created graph, respectively. These values are chosen to match those used in the original LFR benchmark paper [86]. Power analysis determined the number of graphs needed at each of the three μ values (0.2, 0.3 and 0.4) to be 119, which we round up to 120, requiring 360 graphs in total. Graphs used for the experiments of this chapter were different from those of the previous chapter.

4.1.2 Community Detection

A key goal of this work is to develop an approach to produce feature-based explanations for communities using a methodology which is agnostic to the community finding algorithm we wish to explain. We apply our methodology to a number of popular stochastic community detection algorithms, which are designed for detecting partitions (i.e. non-overlapping communities). Specifically, we employ: Louvain [13], Infomap [137], and Label Propagation (LPA) [130]. We select these algorithms as they have been widely employed in the literature and also since each algorithm differs considerably in terms of the objective which it attempts to optimize, as dicussed in the previous Chapter 3. In total, this gives nine experiments (i.e. three algorithms on three μ values). However, we omit LPA on $\mu = 0.4$ since it consistently classifies all nodes in each graph as belonging to a single community.

4.1.3 Network Features

In this chapter, we make use of features which are defined on sets of nodes, rather than individual or pairs of nodes. This will allow us to analyse information on a different resolution, in particular, about whole communities. Therefore, while some of these features draw on the values of features from the previous chapter applied to individual nodes, for example, the *relative degree* which takes a mean of the node degree across all nodes in the set, other features defined here are not directly derived from those of the previous chapter. As before, however, the features have been chosen so as to be interpretable to domain experts who have some previous experience with network analysis. Therefore, while these features may not be familiar to a layperson, they should be considerably easier for a social network analysis expert to under-


A. Relative Density and Relative Degree

B. Relative Diameter

C. Relative Pathlength

D. Cut Ratio and the Internal-External metric

Figure 4.1: Illustrative examples to aid with intuition behind community feature definitions. Reference nodes and edges are highlighted in red.

stand, relative to the specific internal details of the community detection algorithm itself. We consider the simplest features possible, omitting, for example, modularity in favour of those which are easier to understand. Figure 4.1 provides illustrative examples to assist with intuition of these definitions. Definitions for the full longlist of features chosen for these experiments are as follows:

- *Relative Density*: For a set of nodes V connected by a set of edges E, the density of this set is defined as: $\frac{2|E|}{(|V|(|V|-1))}$. In Figure 4.1A, there are 11 edges and 8 nodes. Therefore, this set of nodes has a density of $\frac{2\times 11}{8\times 7} = 0.393$. Then, the *relative density* is the density of the community divided by the density of the whole graph. A high value indicates that the community has a higher frequency of edges than the graph as a whole, while a lower value indicates that the community has a lower frequency of edges than the graph as a whole.

- *Relative Diameter*: For a set of nodes V, the diameter is the maximum length of the shortest path between any two nodes. In Figure 4.1B, the shortest path between the two red nodes is longer than that between any other two nodes, and has a length of 6. Therefore, the diameter of this graph is 6. Then the *relative diameter* is the diameter of the community divided by the diameter of the whole graph. A low value indicates that the most distantly connected nodes of the community are still relatively close together when compared to the whole graph, while a high value indicates that there are distantly connected nodes within the community.

- *Relative Pathlength*: For a set of nodes V, the average shortest path length is defined as: $\sum_{s,t \in V} d(s,t)/(|V|(|V|-1))$ where $d(s,t)$ is the length of the shortest path from s to t. In Figure 4.1C, the average shortest path length is $\frac{1+1+1+2+2+3}{4\times3} = \frac{10}{12} = \frac{5}{6}$. Then the *relative pathlength* is the average shortest path length of the community divided by the average shortest path length of the whole graph. A low value indicates that nodes of the community are on average relatively close together when compared to the whole graph, while a high value indicates that there are distantly connected nodes within the community.

- *Relative Degree*: For a set of nodes V, the average degree centrality is defined as: $\sum_{i \in V} deg(i)/(|V|(|V|-1))$ where $deg(i)$ is the number of edges adjacent to node i. In Figure 4.1A, the average degree centrality is $\frac{2+4+3+2+4+3+3+1}{8\times7} = \frac{22}{56} = 0.393$. Then the *relative degree* is the average degree centrality of the community divided by the average degree centrality of the whole graph. A high value indicates that on average, members of the community have a large number of connections when compared to the graph as a whole, while a low value indicates that they have a low number of connections on average when compared to the graph as a whole.

- *Relative Betweenness*: Let V be a set of nodes of which i, j and k are members. Let $\sigma(j,k)$ be the number of shortest (j,k) paths, and $\sigma(j,k|i)$ be the number of those paths that pass through i. Then the betweenness centrality of node i is given by: $\sum_{j,k \in V} \sigma(j,k|i)/\sigma(j,k)$ Note that if $j = k$, $\sigma(j,k) = 1$ and if either j or $k = i$, then $\sigma(j,k|i) = 0$. Note also that this definition of betweenness centrality is as introduced in section 3.1, and therefore as illustrated in Figure 3.1E in that section, where the red node has high betweenness centrality. Then the relative betweenness is the mean betweenness centrality of the nodes in the community divided by the mean betweenness centrality of the nodes in the whole graph. A high value indicates that the average node of the graph has a higher betweenness centrality than the graph as a whole, meaning that the nodes of the community are well connected. Meanwhile, a low value implies the opposite, i.e. that some nodes of the community are not well connected to other members of the community.

- *Relative Closeness*: Let V be a set of nodes of which i and j are members, and let $d(i,j)$ be the length of the shortest path between nodes i and j. Then the closeness centrality of node i is given by: $(|V|-1)/\sum_{j \neq i} d(j,i)$. Note that this definition of closeness centrality is as introduced in section 3.1, and therefore as illustrated in Figure 3.1G in that section, where the red node has high closeness centrality. Then the relative closeness is the mean closeness centrality of the nodes in the community divided by the mean closeness centrality of nodes in the whole graph. A high value indicates that the average node of the graph has a higher closeess centrality than the graph as a whole, meaning that the nodes of the community are well connected.

- *Cut Ratio*: Let $w_{ij} = 1$ if nodes i and j share an edge, and $w_{ij} = 0$ if they do not. Then we calculate a cut ratio where set A is the set of nodes in the community, and set B is the set of nodes in the graph *not* in the community, as follows: $1/|A||B|\sum_{i \in A, j \in B} w_{ij}$. In Figure 4.1D, the red nodes represent those within the community, while the green nodes represent all those in the graph outside of the community. Within the community, there are 3 nodes, while outside there are 4, and there are 3 edges (the ones represented by dashed lines) which connect nodes in the community to those outside. Therefore, the cut ratio is given by $\frac{1}{3 \times 4} \times 3 = 0.25$. Intuitively, this represents the proportion of the possible edges connecting the two communities which are actually present. Therefore, the highest value is 1 if every node in the community is connected to every node outside the community. The lower this value, the fewer of these possible connections exist. Note that this definition of cut ratio is based on similar principles to the one adapted for individual nodes in Chapter 3, but is not the same metric as it is defined here for sets of nodes.

- *Internal-External*: Let $w_{ij} = 1$ if nodes i and j share an edge, and $w_{ij} = 0$ if they do not. Then we calculate a measure of internal-external where set A is the set of nodes in the community, set E is the set of edges in the community, and set B is the set of nodes in the graph *not* in the community, as follows: $|E|/(|E| + \sum_{i \in A, j \in B} w_{ij})$. Once again, we refer to Figure 4.1D, where the red nodes represent those within the community, and the green nodes represent all those in the graph outside of the community. As before, there are 3 edges (the ones represented by dashed lines) which connect nodes in the community to those outside, and there are 3 edges within the community itself. Therefore, the internal-external metric is given by $\frac{3}{3+3} = 0.5$. Intuitively, this represents the proportion of edges connected to nodes of the community which remain within the community. Therefore, at the highest value of 1, the community is completely disconnected from the rest of the graph. At lower values, it indicates that there are a large number of connections between the community and the rest of the graph.

4.1.4 Graph Rewiring

The longlist of features above is used to calculate features for our set of “real communities", generated using the same three algorithms discussed in Chapter 3: Infomap, Louvain and LPA. We subsequently calculate feature values for “fake communities" using a rewiring approach as follows. To rewire each graph, pairs of edges are selected and at random and their endpoints are swapped using the double_edge_swap method in NetworkX which allows many of these swaps to be made at once. For example, for edges (i, j) and (a, b), after swapping the edges will be (i, a) and (j, b), with the original edges removed. A noise level value is multiplied by the total number of nodes in the graph to determine the number of edge swaps. For our experiments, the noise level is set to 0.5. A swapping probability of 0.5 represents a balance between a uniform structure and a random structure in the original discussion of graph rewiring by Watts and Strogatz [177].

4.1.5 Community Classification

Given our labelled set of “real" and “fake" community feature values generated as per above, we apply a random forest classifier to distinguish between the classes using the following evaluation methodology. We first split the examples into 80% training data and 20% test data. We then train the classifier 50 times on the training data, comprised of 10 repeats of 5-fold cross-validation. An importance score is calculated for each node feature after the 50 runs, using the held-out test data. These 50 values then provide us with a distribution of importance score values for that feature.

4.1.6 Statistical Methodology and Pilot Study

In order to confirm the suitability of the above methodology, we ran an initial pilot study on 20 LFR generated networks at each μ value (i.e. 60 networks total). This pilot study was used to determine the behaviour of the underlying phenomena so that we could conduct a proper power analysis. Note that the pilot study networks are not included in the final analysis, and that for this pilot study, we used permutation scores alone and not Shapley values.

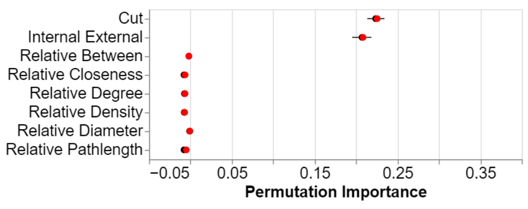
Distributions of each permutation importance for all features were created. We then ran a Shapiro-Wilk test to determine the normality of the phenomena. As the majority of the distributions in the pilot study followed a normal distribution (77%), we took the assumption that the underlying phenomena were normal for our power analysis.

Our statistical methodology was also set before our experiments. Bonferroni-Holm corrections are used in our analysis. The normality of the final permutation importance values on the main experiment are confirmed using a repeat of the Shapiro-Wilk tests, and thus t-tests are used to compare the features. Power analysis was conducted with the following parameters: Cohen's effect size of 0.3, significance level of 0.05, and a power of 0.9. We treat each of the three community finding algorithms independently for analysis and use the pairwise t-tests to identify significantly different pairs of features.

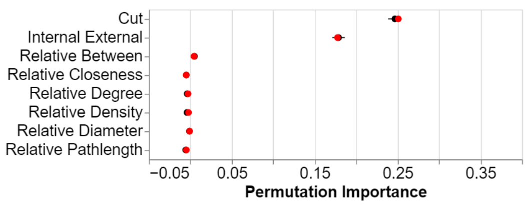
4.2 Results

Following the methodology described in the last section, we now detail the results of the experiment performed on the complete collection of 120 LFR graphs. As discussed, our aim is to identify which of the features listed in section 4.1.3 have a significantly greater importance than the others in predicting whether a set of nodes represents a “real" or “fake" community. Distributions of permutation importance for the features on each experiment are displayed below in Figure 4.2. These distributions are constructed using the 50 permutation importance values calculated during training, as described in section 3.5.

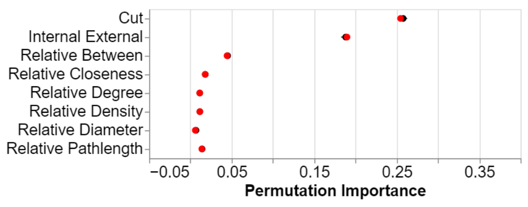
From our experimental results, we see qualitatively that the cut ratio and internal-external metrics are consistently the most important features in distinguishing the “real communities" from the “fake communities". However, as the μ value increases, there is evidence to suggest that the relative betweenness may also have some importance. The Bonferroni-Holm corrected t-tests identified significant differences between these three “important" metrics and all other metrics in the study with the following exceptions:



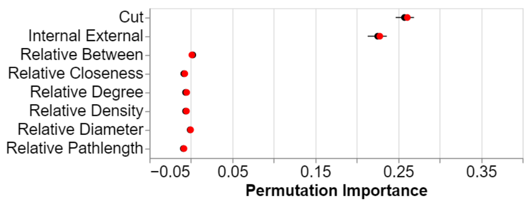
A. Infomap, $\mu = 0.2$



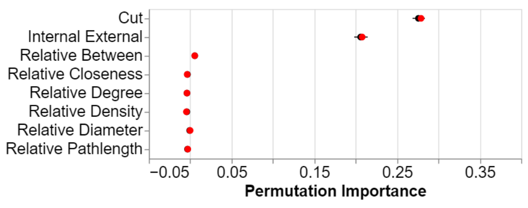
B. Infomap, $\mu = 0.3$



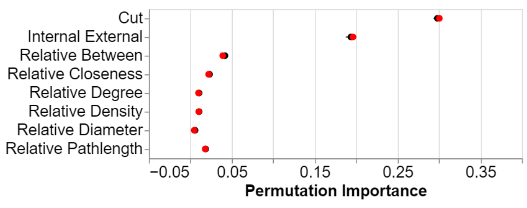
C. Infomap, $\mu = 0.4$



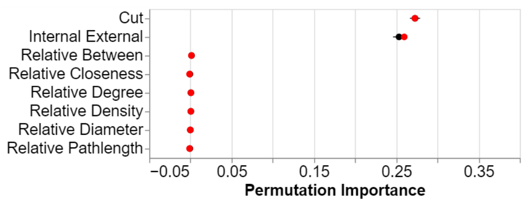
D. Louvain, $\mu = 0.2$



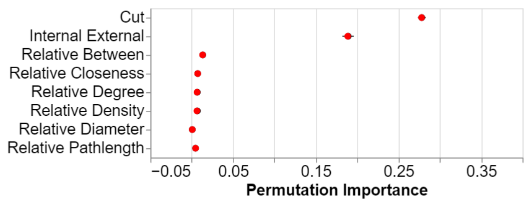
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$



G. LPA, $\mu = 0.2$



H. LPA, $\mu = 0.3$

Figure 4.2: Results of the community feature experiments. Plots show the permutation importance of the features, across three community detection algorithms and graphs with different levels of community mixing (μ). A mean value is indicated as a black dot and median as a red dot. Lines indicate 95% bootstrapped confidence intervals.

- Infomap $\mu = 0.2$: relative betweenness compared with relative diameter
- Louvain $\mu = 0.2$: relative betweenness compared with relative diameter
- LPA $\mu = 0.2$: relative betweenness compared with relative degree and relative density

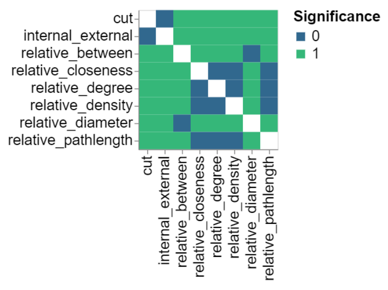
The full set of statistical results is provided in Fig. 4.3.

In addition to the permutation importance experiments, we also generated results using Shapley values, as these are well-known among the explainability community and are known to have mathematically desirable properties. The results are shown in Figure 4.4. As with the permutation importance results, we see qualitatively that the cut ratio and internal-external metrics are consistently the most important features. The greater importance of relative betweenness is also visible, though other features (relative degree and relative density) also appear to have slightly more importance as well. Relative diameter is consistently the least important.

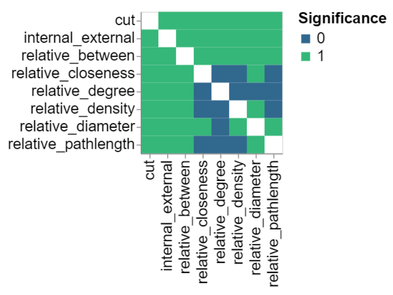
4.3 Discussion

The Infomap, Louvain, and LPA algorithms all performed similarly in our permutation importance experiments, with three features being consistently important for distinguishing the real communities from the fake communities. These were: cut ratio and internal-external for all experiments, with relative betweenness becoming increasingly important with increasing mixing parameter. Both cut ratio and the internal-external metric relate to proportions of edges which connect two communities. Thus, when the communities are more clearly defined, or when μ has a low value, more agglomerative features are important for explaining community structure for the set of nodes considered. However, as μ increases and the communities become less well defined, divisive features, such as relative betweenness, become of increasing importance. A possible explanation for this finding is that local connections and neighbours to the node set can be used to understand the community structure when there are few cross-community edges, but as these edges increase in prevalence, more global features, such as relative betweenness, become important in explaining community structure.

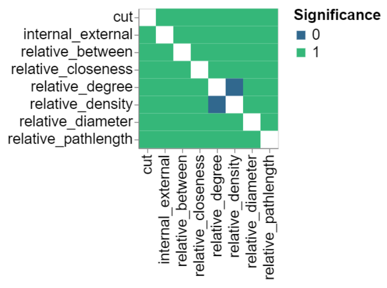
When we swapped out the permutation importance score for Shapley values, we once again found that cut ratio and the internal-external metric were consistently most important for all algorithms and at all μ levels. The relative betweenness also grows in importance as the μ value increases, as with permutation importance, however particularly so for the Infomap algorithm. Additionally, we see that some of the other less important features have more variation in their



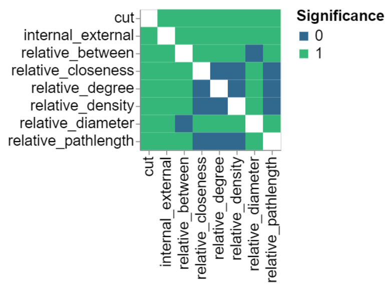
A. Infomap, $\mu = 0.2$



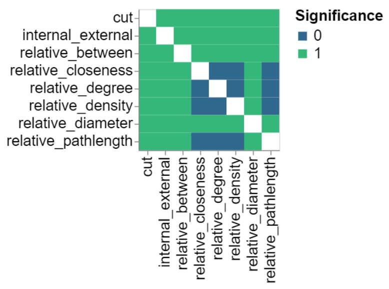
B. Infomap, $\mu = 0.3$



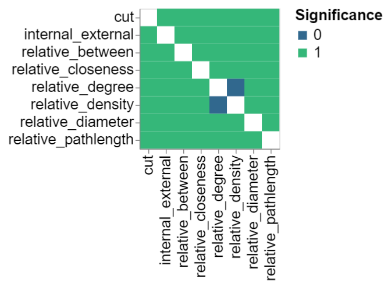
C. Infomap, $\mu = 0.4$



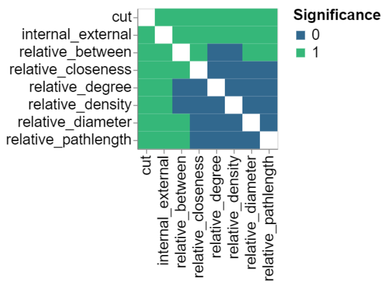
D. Louvain, $\mu = 0.2$



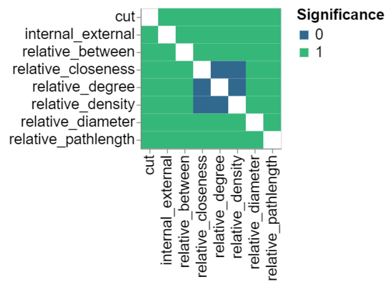
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$

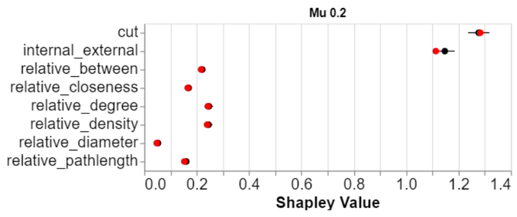


G. LPA, $\mu = 0.2$

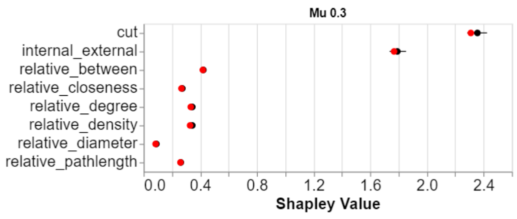


H. LPA, $\mu = 0.3$

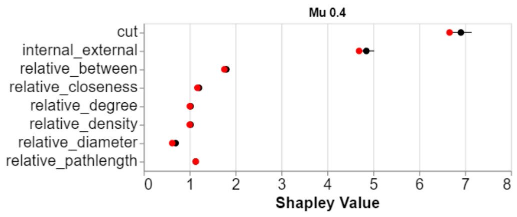
Figure 4.3: Results of Bonferroni-Holm corrected pairwise t-tests for different detection algorithms and μ values. Significant is represented with colour, with a key in the upper right. A significance of 0 represents no significant difference; a significance of 1 represents a significant difference.



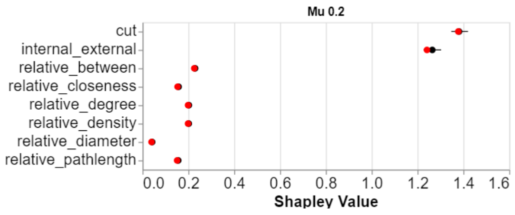
A. Infomap, $\mu = 0.2$



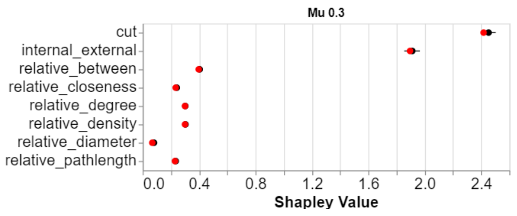
B. Infomap, $\mu = 0.3$



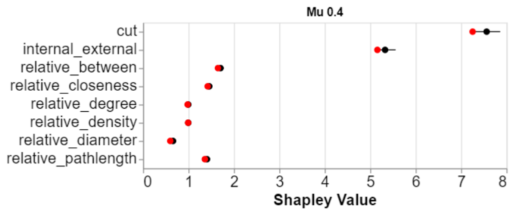
C. Infomap, $\mu = 0.4$



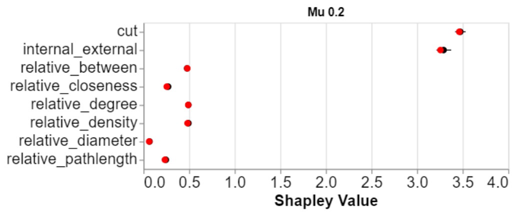
D. Louvain, $\mu = 0.2$



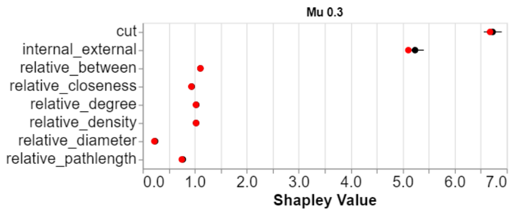
E. Louvain, $\mu = 0.3$



F. Louvain, $\mu = 0.4$



G. LPA, $\mu = 0.2$



H. LPA, $\mu = 0.3$

Figure 4.4: Results of the community feature experiments. Plots show the Shapley values of the features, across three community detection algorithms and graphs with different levels of community mixing (μ). A mean value is indicated as a black dot and median as a red dot. Lines indicate 95% bootstrapped confidence intervals.

importance according to the Shapley value. For example, the relative pathlength appears more important at higher μ values, particularly for Louvain. This slightly diminishes the perceived importance of the relative betweenness when visually scanning our results plots.

4.3.1 Relation to Results on Node and Node-Pair Features

Surprisingly, despite the fact that the community detection algorithms use different objective functions and optimisation strategies, the most informative features were once again the same for all. This mirrors our results in Chapter 3, where we found similar results for the three community detection algorithms for node and node-pair features. This provides evidence that post-hoc explainable network analysis is feasible, independent of the choice of algorithm.

Furthermore, our finding here that relative betweenness grows in importance as the μ value increases also relates to the results of our node-pair experiment in Chapter 3, where maximum edge centrality increased in importance with a growing value of μ. In both cases, this is a divisive feature, which we hypothesised was due to the need for more global information in explaining community structure when communities are more well-mixed. This contrasts with the agglomerative features that are consistently important in all experiments at lower amounts of community mixing.

4.4 Conclusion

In this chapter, we have adapted our methodology presented in the previous chapter to a new level of social network analysis feature: those evaluated on sets of nodes, rather than individual nodes or pairs. Our aim in doing so was to identify metrics which could be used to improve understanding global community structure of a network, for incorporation into a visual analytics system that could be used by public health researchers with social network analysis expertise for understanding social networks. The metrics identified in the previous chapter could also be included to provide local information about individuals within the network.

Beyond the development of this visual analytics system, additional areas for future work include: extending this study to real-world network data; further exploration of the effects of the graph rewiring process on the results, e.g. by varying the number of edge swaps; incorporation of localised measures, such as betweenness centrality, into the longlist of features; and a qualitative assessment of the explainable features using the feedback of domain experts.

These possible adaptations of our presented experiments are beyond the scope of this thesis. Instead, we next examine the application of predictive uncertainty quantification to a setting with real social network data. Uncertainty quantification differs in many ways from the identification of interpretable features as presented in this chapter and the last. Nevertheless, it is another approach which aids greater understanding in the presence of black-box algorithms. We explore these ideas further in the next chapter, Chapter 5.

Chapter 5

Uncertainty Quantification in Social Network Problems

Further to the work in the previous chapters on applying explainable AI techniques to social network analysis, we continue by exploring how uncertainty quantification can be applied to a prevalent problem in applications developed for social networks. Unlike the work of the previous chapters, the uncertainty quantification approaches we explore are applied to deep learning techniques, however our focus is still on social networks. Uncertainty quantification is not usually regarded as an explainable AI approach in itself, as the methods used can themselves be black-boxes [47, 83, 124]. Nevertheless, uncertainty quantification provides additional insight in settings where predictions can be misleading or biased, and has been considered closely related to explainable AI in its intentions [145]. The work of this chapter was carried out in collaboration with Meta Platforms, who own the Facebook platform and associated data, firstly during a research internship between June 2021 and September 2021, and then continued through a research collaboration agreement between October 2021 and February 2022. Therefore, the specific prediction problem we address is one encountered by those who manage large social media platforms such as Facebook. A workshop paper on the work of this chapter is in preparation.

On such platforms as Facebook, content moderation is a growing problem due to the consistently increasing scale of the user base and the quantity of uploaded content. In particular, Facebook has over 2.9 billion monthly users as of 2021. At the present, machine learning systems are not a complete replacement for human reviewers in identifying content which breaches platform policies. Conversely, it is increasingly difficult to rely solely on human reviewers, as the scale of online content continues to grow. A good compromise is to incorporate the insight from both machine learning and human reviewers. One way to do this is by first using machine learning to filter the large pool of content, before passing this filtered selection to the human reviewers [168]. During the filtering process, the aim is to identify content which is likely to cause the greatest harm. At Facebook, more than 95% of content violating hate speech standards is removed by AI [21], though hate speech constitutes only one type of policy-breaching content. Such AI systems [144] are trained to automatically classify content as violating community standards and remove harmful content as soon as it is created. Nonetheless, Facebook continues to use human reviewers in addition to their machine learning approaches.

The criteria for prioritisation when passing content for human review are two-fold: firstly, the degree of harm which could be caused must be assessed (for example, content promoting physical violence takes precedence over nuisance spam). However, all other things being equal, we can also take into consideration the reach of the content. It is more important to remove that which will be viewed by thousands of users than that which no one will see. Thus, for a given degree of harm, we use the number of views as a proxy for the overall reach of the harm of content. It follows that predicting the number of views a piece of content will receive is an important problem in this space. In particular, it is important to predict so-called 'viral' content, since if it is harmful, it can lead to a large number of users being exposed to inappropriate or dangerous content.

Ideally, we would like to predict the popularity of the content as early as possible, to prevent avoidable harm. Unfortunately, predicting content popularity at the time of its creation (known as a "cold start" prediction), in contrast to predicting its popularity after the initial cascade of engagement has been observed, is known to be challenging [7, 102, 111]. In the work by Martin et al. [102], the authors attribute this to the unpredictability of complex social systems and argue that high accuracy would effectively require a perfect "ex-ante" knowledge of the world. The highly-skewed nature of popularity distribution, in which a large proportion of content receives no engagement, is an additional challenge. Previous studies, such as the work by Arapakis et al. [7], focused on content with non-zero popularity, which artificially simplifies the problem and hides the fact that models fail to generalise in the presence of a highly-inflated proportion of unpopular content. Though we have no statistical evidence for the reasons content might receive no views, we can intelligently speculate based on our understanding of how platforms such as Facebook operate. Such content may include: individual photos in large photo albums; content posted by spam accounts with few or no "friends"; or content which is deleted very quickly after first being posted. In this chapter, we evaluate the performance of approaches in the face of a large proportion of content receiving zero views.

Thankfully, predicting the exact number of views that will be received by a piece of content is not only challenging, but unnecessary. As stated, we are particularly interested in highly viral content which is likely to reach many users, and less interested in distinguishing content which will receive very few views from that which will receive none. Therefore, instead of making a prediction of the exact number of views, we focus on the prediction of narrow, high-probability intervals, under the assumption that this is a simplification of the problem and still allows us to identify content which will receive significantly more views than others. In order to do this, we employ uncertainty quantification approaches.

Uncertainty quantification as we apply it here differs from the explainability methodology introduced in the previous two chapters, in that the predictive problem is fundamentally changed rather than understood with the additional information gained by post-hoc explanations. Nevertheless, it has philosophical similarities; in both cases, we are exploring ways to improve user understanding of algorithmic outputs in a setting where black-box approaches provide insufficient information with which to reason about the solution. Here, we replace a "point prediction", i.e. a single number representing the expected number of views, with an upper and lower bound. Bounds not only provide an estimate of the true value, as point predictions do, but their range implies something about the degree of uncertainty in the prediction. For example, if there is a small gap between the upper and lower bounds then we might be more convinced that we can make a close estimate of the true value than if there is a large gap. In this way, having both an upper and lower bound provides an additional dimension of information than a single point prediction.

We perform our analysis using a large real-life dataset of content collected at Facebook, for which features are recorded at the time of its creation, making it a "cold-start" prediction problem, where the estimation of exact popularity is highly challenging. The data used for this analysis is proprietary and was made available to us only for the duration of the collaboration agreement. The purpose of our work is to evaluate the performance of well-known existing methods for uncertainty quantification at generating high-quality intervals in the face of the presence of a highly skewed distribution, where much of the content receives zero views, and to identify whether there is an approach which can work well in this setting. To our knowledge, this specific domain problem is a novel application of existing uncertainty quantification approaches, and we hypothesise that there will be poor performance due to the assumptions made about the data distribution by existing approaches. In our experiments, we compare well-known methods MC Dropout [47] and quantile regression [77] to some statistical baselines, observing that, while quantile regression is a minor improvement over these, MC Dropout fails to perform well in the setting with the zero-skewed distribution. We chose these two methods as they are commonly used approaches to uncertainty quantification, and represent modelling of the two uncertainty types - aleatoric and epistemic uncertainty. MC Dropout was chosen over other more computationally demanding approaches such as deep ensembles [83], which are less scalable in storage. Preliminary experiments were also done with quality-driven prediction intervals introduced by Pearce et. al [124], though we did not develop this approach as it performed poorly with our dataset. Recent advances have led to new methods for prediction interval generation [57, 72, 143] which could be considered in future work.

We then propose a novel methodology through incorporation of a binary classifier with MC Dropout to separate out the content which receives zero views from that which doesn't, as the large proportion of zeros in the dataset causes a skewed distribution which we hypothesise causes the poor performance of MC Dropout. This allows improvement over the performance of all other methods in some regions. In particular, though it performs worse than other methods when prediction intervals are small, it eventually improves over other methods as the size of performance intervals grows. However, the size of prediction intervals using this method has an upper bound, meaning it is unable to compete at all at the largest prediction interval sizes. We further explore the limitations of this approach in this chapter.

5.1 The Cold-Start Prediction Problem

Our goal is to identify whether there is an uncertainty quantification approach which can perform well at the challenging cold-start prediction problem, i.e. the prediction of content popularity at the moment of its creation, when no engagement has yet been observed. In particular, we are interested in predicting the popularity in the form of number of views over a given time horizon after features are recorded, which in this case is 7 days. In this section, we will first demonstrate the challenge of predicting content popularity at its time of creation, thereby motivating our interest in uncertainty quantification.

For our research, we consider a dataset composed of content shared on Facebook, comprised of several types, including text posts, photos, videos, comments, and other similar variations on these. A breakdown is shown in Table 5.1. The total proportion of content which received zero views is 59%, which reaffirms a similar observation reported in the work by Arapakis et al. [7] on news resharing activities on Twitter.

Content Types		
Content Type	Count	0s Count
Photo	25271	18113
Post	25809	17013
Video	10439	8601
Comment	18815	3495
Total	80334	47222

Table 5.1: A breakdown of content types contained in our cold start dataset. For each content type, we show the number in our dataset and the number of those receiving 0 views.

The full distribution of the number of views is displayed in Figure 5.1. View count in this distribution is normalised by the mean number of views received across all content, and plotted with a log scale due to the long-tale distribution shape. The curve begins just below 0.6, reflecting the fact that 59% of content receives no views. There is initially a steep increase in the curve, reflecting the fact that a large proportion of the content receives only a small number of views. The steepness begins to decrease at a CDF of around 0.9, indicating that almost 90% of the content receives a small number of views in this region. From here, the curve gradually plateaus, 99% of the content receiving a moderate number of views or less. The long asymptote indicates that a very small amount of content receives some very high number of views.

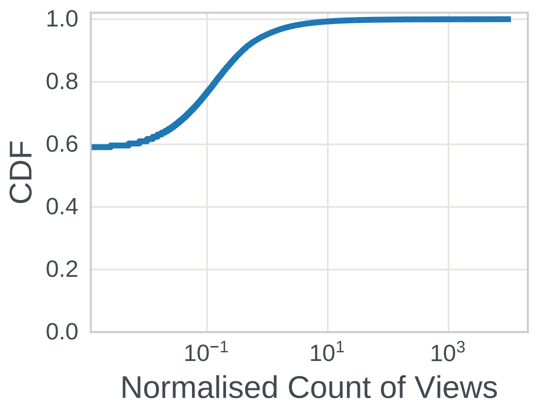


Figure 5.1: The distribution of log-transformed counts of views received by content items within 7 days of their creation.

The dataset was collected from content produced over the course of a month (from 26th August 2021 to 18th September 2021). For each content item, we record over 1000 features at the beginning of the time horizon, including several anonymised features describing the user who posted it, the page it was posted on and features of the content itself. The number of views accumulated on it in the next 7 days is then recorded as the label. With reproducibility in mind, we restrict our attention to content posted on public pages, therefore sampling from a distribution which other researchers also have access to.

To illustrate the challenge of predicting popularity at “cold start”, i.e. at the moment of content creation, we gather a second dataset from the same timeframe, which we call the “cascade” dataset as it has been gathered after the intial cascade of views has been observed, and conduct an experiment to compare the difficulty of the prediction problem on the two datasets. The second “cascade” dataset is composed of content items sampled once the content is over a minute old, allowing us to observe the beginning of the engagement cascade which we have not yet observed for the “cold start” data. In contrast with our “cold start” dataset, we are able to record a set of features which characterise this observed social engagement, in addition to the original features used in our “cold start” dataset. For example, these features include those about the number of views received over different time horizons, such as 10 seconds or 30 seconds, and the rate at which views are being received. We then train two regressors to predict popularity on each of the two datasets. For the “cascade” dataset, the additional features are included for training. For our regressors, we use gradient boosting with 500 estimators with a maximum depth of 5, and train for 500 epochs.

We then evaluate performance of these two regressors, using three metrics which are defined as follows:

- *p80 Ratio Error:* The factor of 10 for which our prediction is within this distance from the true value 80% of the time. For example, if the p80 ratio error is 2, it means that our predicted number of views is within $10^2 = 100$ of the true number of views 80% of the time. A lower value is better, as it indicates that our prediction is within a smaller margin of the true value. This metric is a version of the *relative error.*

- *Non-View Weighted Magnitude Accuracy:* The proportion of data points for which the magnitude of the prediction is correct (i.e. log_{10}(prediction) $\approx$ log_{10}(true views)) A higher value for this accuracy indicates that the model is able to predict the magnitude of views correctly across the whole dataset, regardless of the number of views.

- *View Weighted Magnitude Accuracy:* The proportion of data points for which the magnitude of the prediction is correct, weighted by number of views. A higher value for this

Point Prediction Metrics		
Metric	Cold Start Data	Cascade Data
p80 Ratio Error	8.82	3.44
Proportion Predictions Magnitude Accuracy	0.823	0.942
Proportion True Views Magnitude Accuracy	0.0475	0.942

Table 5.2: Comparison of point prediction accuracy for cascade data vs. cold start data using gradient boosting.

accuracy indicates that the model is not only able to predict the magnitude of views accurately, but in particular, performs well on content receiving a higher number of views, which is the region of the dataset which is most important for content moderation.

These metrics were chosen as they allow for better comparison of results from two very long-tailed distributions than standard metrics such as RMSE and MAE, which are not informative in this setting. The results in Table 5.2 quantitatively highlight the magnitude of the problem: whereas the popularity of content on the "cascade" dataset can be predicted with a very high proportion of 94% of the true views matched in order of magnitude by the prediction when weighted by number of views, a similar exercise on the "cold start" dataset achieves only 5% on this metric. This difficulty in making accurate predictions at content creation time motivates our experiments to identify an uncertainty quantification approach which can provide high-quality prediction intervals. The prediction of upper and lower bounds requires less certainty about the exact number of views, while providing additional information which can be used to prioritise potentially harmful content for human review.

5.2 Problem Formulation

Due to the difficulty in predicting the exact number of views a piece of content will receive at the time of its creation, we therefore consider instead predicting upper and lower bounds on the number of views the content will receive. This provides an additional axis of information; not only can we get a rough idea of the expected number of views, but also an estimation of the certainty of this prediction. Thus, we are less likely to miss content which ultimately receives a large number of views, i.e. "goes viral" just because our prediction accuracy is low. By including both an upper and lower bound, we can also account for content which has a middling probability of receiving many views, in addition to any content we would've picked up using a single "point prediction" of a very high number of views.

However, we first need to quantify how to evaluate prediction intervals. When training a model to predict upper and lower bounds, the first metric of interest is the frequency with which the true value falls inside the bounds, which is essentially a measure of accuracy. If the true value falls between the upper and lower bound, we can consider this a correct prediction. However, it is very easy to achieve close to 100% accuracy using this metric alone: by simply setting all lower bounds to be an arbitrarily low number, and all higher bounds to be an arbitrarily high number, almost all true values will fall inside them. In this case, the prediction of bounds is accurate, but highly uninformative, as we don't learn anything about the likely number of views. Therefore, we are interested in constraining the prediction of bounds by minimising the distance between them. By considering both metrics, we encounter a trade-off: we would like the interval to be as small as possible, while containing the true value with the highest possible probability. We formally define these two metrics, the coverage and prediction interval width, as follows.

5.2.1 Coverage and Prediction Interval Width

Let $x \in \mathcal{X} \subseteq \mathbb{R}^d$ be the feature vector of a content item and $y \in \mathcal{Y} \subseteq \mathbb{R}$ be the number of views of the content item over a fixed time horizon.

Suppose that we have a model $x \to (g(x), h(x))$, where for every given feature vector x, $[g(x), h(x)]$ is a prediction interval such that $y \in [g(x), h(x)]$ if the interval is accurate. Here $g(x)$ and $h(x)$ are lower and upper boundaries of the prediction interval.

We define the *expected coverage* C as follows:

$$C = \mathbb{P}[y \in [g(x), h(x)]] \tag{5.1}$$

Thus coverage represents the proportion of prediction intervals for which y falls within the bounds $[g(x), h(x)]$, which can intuitively be thought of as the interval accuracy metric.

We define the *expected prediction interval width* W as:

$$W = \mathbb{E}[|h(x) - g(x)|]. \tag{5.2}$$

As explored previously, larger intervals enable higher coverage. In our experiments, we would therefore like for the prediction interval width $|h(x) - g(x)|$ to be as small as possible for a given coverage. For this reason, we will generate intervals of several different sizes and evaluate approaches by plotting the interval width against coverage. It may be the case that some approaches perform better in different regions of this plot.

5.2.2 Baseline for interval predictions

Having established these metrics, we now bring our attention to the definition of a simple baseline model for calculating prediction intervals. The purpose of this model is to give us something against which to compare the uncertainty quantification approaches. The input for this model, as with all the uncertainty quantification approaches, are the features of the content, as discussed previously. However, for the baseline, we don't make use of all of these features. The output will be the predicted upper and lower bound, creating a prediction interval.

For our first baseline, we don't use any of the features and instead compute symmetrical pairs of quantiles from the empirical distribution of views observed in the training data. The empirical distribution is an estimate of the cumulative distribution that generated the sample, and is a step function which jumps up by $\frac{1}{n}$ at each of the n datapoints. Thus, the function is given by:

$$F_n(t) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{X_i \leq t}$$

The symmetrical quantiles we calculate are $(\frac{1}{2}(1-\alpha), \frac{1}{2}(1+\alpha))$. Quantiles are points which divide the range of a probability distribution into continuous intervals with equal probabilities, so in this case, $\frac{1}{2}(1-\alpha)$ of the content items would fall below the lower bound, for example. We then use these constant values as the upper and lower bounds of the prediction interval for *every* item in the test set. Here, α is a value we can vary to adjust the size of the prediction intervals. By the definition of a quantile, we can hope to achieve an observed coverage of α for the prediction intervals we have generated ($\frac{1}{2}(1+\alpha) - \frac{1}{2}(1-\alpha) = \alpha$).

This baseline produces a constant interval for all data in the test set, i.e. it does not condition on the input features. To iterate on this initial approach, we also calculate symmetrical quantiles in the same way from distributions of item views conditioned on content type (i.e. text, photo, video etc.) in the training set. For each item in the test set, we then use as a prediction interval the upper and lower quantiles calculated from the corresponding training set distribution of the same content-type. In this sense, we are conditioning on a single feature. With this approach, prediction intervals will still be constant for content items within each content type, so it remains a relatively uninformed approach.

Against these baselines, we will compare two popular existing approaches on our "cold-start" dataset, to assess whether either perform well on this challenging and unusual data distribution. We also propose our own approach, which we call HZIM, which incorporates a binary classifier to identify that content which receives zero views, in an attempt to avoid the issues this high proportion of data introduces. We will describe each of these approaches in the coming subsections.

5.2.3 MC Dropout

The first of the existing approaches we evaluate is the Monte Carlo (MC) Dropout method introduced in the work by Gal et al. [47]. MC Dropout makes use of dropout, which is normally used as a regularisation technique to prevent overfitting. When dropout is used, some units of the neural network are randomly dropped (along with their connections) during the training process, to prevent their co-adaptation. In the case of MC Dropout, this dropout regularisation is instead used to calculate prediction intervals. The network is trained in the same way as normal, with the main difference occurring at inference time. Rather than passing the input data through the network once, the data is passed through several times with random activation of the dropout layers. Intuitively, this gives several estimates of the true value using several perturbed versions of the trained neural network. This selection of sample predictions can then be used as a sample distribution, from which an expected value can be calculated, as well as quantiles to represent upper and lower bounds of the prediction interval. We formalise this as follows.

During training with dropout, units are randomly dropped with probabilities p_k for every weight layer $k \in [1..K]$, where K is the number of layers in the neural network. With a regularisation term included, the loss is given by:

$$L = \frac{1}{N}\sum_{i=1}^{N} l(y_i, \hat{y}_i) + \lambda \sum_{k=1}^{K} (||W_k||_2^2 + ||b_k||_2^2)$$

where N is the number of data points, each having an output $\hat{y}_i$ and true label y_i, and W_k and b_k denote the weight matrix and bias vector of the kth layer. l is the unregularised loss function, which in our case is the mean-squared error. λ is a hyperparameter which determines the strength of the regularisation.

To apply the MC Dropout approach to calculate prediction intervals, we then also make multiple forward passes using this network at test time. On each forward pass, weight $w_{j,k}$ (i.e. the $j-$th weight in the k-th layer) is set to 0 with probability p_k, the dropout probability defined for layer k during training. We denote this weight at test time, which is 0 with probability p_k or the original weight $w_{j,k}$ with probability $1 - p_k$, as $\tilde{w}_{j,k}$. A forward pass of the network then results in:

$$\tilde{y}_i = f(x_i, \tilde{w}_{j,k})$$

Thus, each forward pass is dependent on the resulting random variables $\tilde{w}_{j,k}$. We sample $T = 100$ such forward passes, resulting in an output $\tilde{y}_i^t$ for each $t < T$. Finally, to generate prediction intervals for content item i, we estimate a symmetric pair of quantiles $(\frac{\alpha}{2}, 1 - \frac{\alpha}{2})$ from the sample distribution $\{\tilde{y}_i^t\}_{t=1}^T$.

5.2.4 Quantile Regression

The second existing method we evaluate is quantile regression [75, 76]. In this case, we train two neural networks to predict each of the upper and lower bound respectively. Each network is trained as a regression model, where the loss function is designed such that it predicts a given quantile, α, conditional on the input data, rather than an expected value. Intuitively, the value it predicts should be such that the probability of the true value being below this is α.

Specifically, this is conditional quantile regression, where we estimate a quantile of Y conditional on X. Formally, the conditional distribution function of Y given $X = x$ is:

$$F(Y|X = x) := \mathbb{P}\{Y \leq y|X = x\}$$

and the αth conditional quantile function is:

$$q_a(x) := \inf\{y \in \mathbb{R} : F(y|X = x) \geq \alpha\}$$

In the machine learning setting where we work with a dataset, X_i describes the features of data point i while Y_i is the label. With our dataset (X_i, Y_i) we aim to estimate a conditional quantile function q_α by using a neural network for approximation. Each network is trained using a pinball loss [153], which aims to under-predict against the true value with probability τ, a hyperparameter representing the desired quantile. Let y be the true value (in this case, the number of views the content receives), τ be the target quantile, and z be the output of the neural network. Then the pinball loss is given by:

$$L_\tau = \begin{cases} (y - z)\tau & y \geq z \\ (z - y)(1 - \tau) & z > y \end{cases}$$

This loss function penalises under-prediction (z < y) or over-prediction (z > y) relative to the parameter τ. By varying τ, the model can be trained to predict different quantiles of the target distribution.

For this approach, we train two neural networks in parallel to generate the lower and upper bounds respectively. These are once again generated symmetrically by setting $\tau = \frac{\alpha}{2}$ for the first network and $\tau = 1 - \frac{\alpha}{2}$ for the second to obtain an interval $(\frac{\alpha}{2}, 1 - \frac{\alpha}{2})$.

5.2.5 Hierarchical Zero-Inflated Method (HZIM)

In initial experiments, MC Dropout was vastly outperformed on our metrics by the baselines and quantile regression, demonstrating that it is poorly adapted for the setting with a zero-inflated distribution. We aim to improve its performance, and ideally outperform the other methods, by incorporating a binary classifier to form a hierarchical model. Intuitively, the idea is to separate out the content which receives zero views using the binary classifier, and to use MC Dropout on the remainder of the data only. In this way, we hope that the poor adaptation of the MC Dropout approach to a high proportion of zero-receiving content can be circumvented. Specifically, we incorporate the binary classifier model as follows, and call this new model HZIM (Hierarchical Zero-Inflated Method).

The MC Dropout neural network is identical in architecture and methodology to the one used previously, but is instead trained on a dataset composed only of the non-zero data points. In parallel, we train a simple feedforward two-layer neural network to perform binary classification separating content receiving zero views from that receiving a non-zero number of views. I.e. these two models are trained separately. We then generate prediction intervals as follows.

We first use the binary classifier to predict whether the piece of content will receive zero or a non-zero number of views. In the former case, we immediately set the prediction interval to $[0,0]$. In the latter case, we then use the trained MC Dropout model to make an interval prediction as described previously.

By incorporating the binary classifier, we hope that we can improve performance over plain MC Dropout in two ways:

- Reducing the mean size of prediction intervals by reducing many of them to a width of 0 where the model is confident the content will receive no views.
- Improving the performance of the MC Dropout model on non-zero data points by discarding the consideration of a large peak at zero in the distribution.

5.3 Experimental Setup

For our main experiment, we compare the performance of our baselines, MC Dropout, quantile regression, and HZIM at generating prediction intervals for our "cold start" dataset. Our aim is to discover whether any of them performs well at generating high-quality intervals (i.e. intervals with a high coverage relative to the interval width) in the face of our unusual data distribution. A high coverage can easily be achieved by setting intervals arbitrarily wide, as setting an extremely low lower bound and high upper bound will likely catch most of the data, which we demonstrate with the baselines. We hope to discover a method which can improve over these baselines, thus being well-adapted for use on this “cold start” problem. In this context, one model outperforms another if it achieves the same coverage using smaller intervals, or better coverage using intervals of the same size.

In order to assess coverage in relation to the interval width, we use a range of α values for each method and recalculate the coverage and mean interval width for each value of α. These pairs of values for coverage and mean interval width are then used to generate a curve, which are displayed in our plots. Since better performance corresponds to a higher coverage for the same mean interval width, a method is performing better than another in any region where it has a larger area under the curve (AUC).

For MC Dropout, quantile regression and the MC Dropout model in HZIM, we train neural networks with 3 linear layers, dropout layers with dropout probability 0.1, and ReLU activations. For the HZIM method, we also separately train a binary classifier with just 2 linear layers, between which there is a ReLU activation, and a sigmoid function applied to the output. For both MC Dropout and HZIM, these trained models are used for all values of α; for quantile regression, new models must be trained for new values of α, as this is a hyperparameter used in the loss function.

In addition to evaluating coverage (the proportion of data points for which the true value falls inside the prediction interval), we also evaluate the proportion of data points for which the true value falls above the upper bound, and below the lower bound, and generate similar plots for each of these. A different type of error is represented by each of these metrics (i.e. overestimation or underestimation), so in this case larger AUC corresponds to more errors made in the corresponding region.

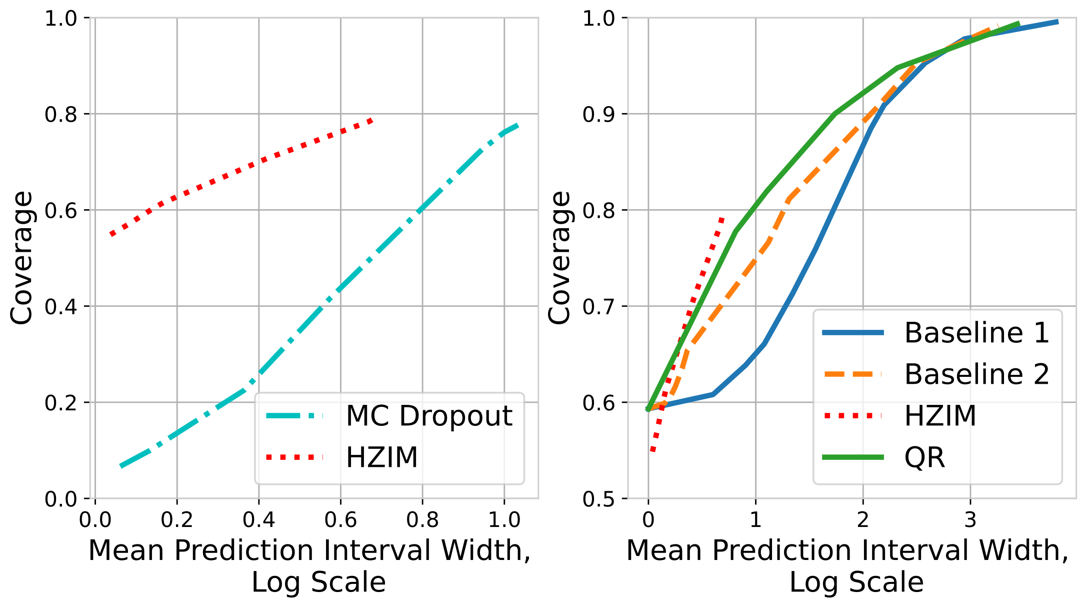


Figure 5.2: Coverage vs confidence interval length on "cold start" data. Acronyms correspond to the following methods: BL1 - Baseline 1 (quantiles taken from the empirical baseline); BL2 - Baseline 2 (as BL1, but conditioned on the content type); HZIM - our Hierarchical Zero-Inflated Model.

5.4 Results

We observe the performance of the methods in Figure 5.2, where we compare coverage to mean prediction interval width. We separate MC Dropout into a distinct subplot as it is unable to achieve performance on the scale displayed for the other methods. With this separate subplot we demonstrate the improvement HZIM makes by the inclusion of a binary classifier.

For both the baselines and quantile regression, the lowest interval width of zero achieves a fixed coverage of 59%, representing the total proportion of content receiving zero views, while HZIM performs worse at this point due to errors introduced by the binary classifier. While the other methods predict that everything receives zero views for an interval width of zero, i.e. a point prediction, this is not the case for HZIM. Instead, the MC Dropout model (which doesn't see any content receiving zero views in training) is used for those datapoints which are classified by the binary classifier as receiving more than zero views. Therefore, when the binary classifier misclassifies content as receiving a non-zero number of views, the coverage drops, causing this lower performance for an interval width of zero.

As mean prediction interval width increases, all methods achieve increasing coverage. HZIM quickly outperforms other methods, though stops abruptly at a coverage of 0.8. This is caused by the MC Dropout model component of HZIM not generating any intervals larger than a certain width. Beyond this point, quantile regression continues to outperform the baselines.

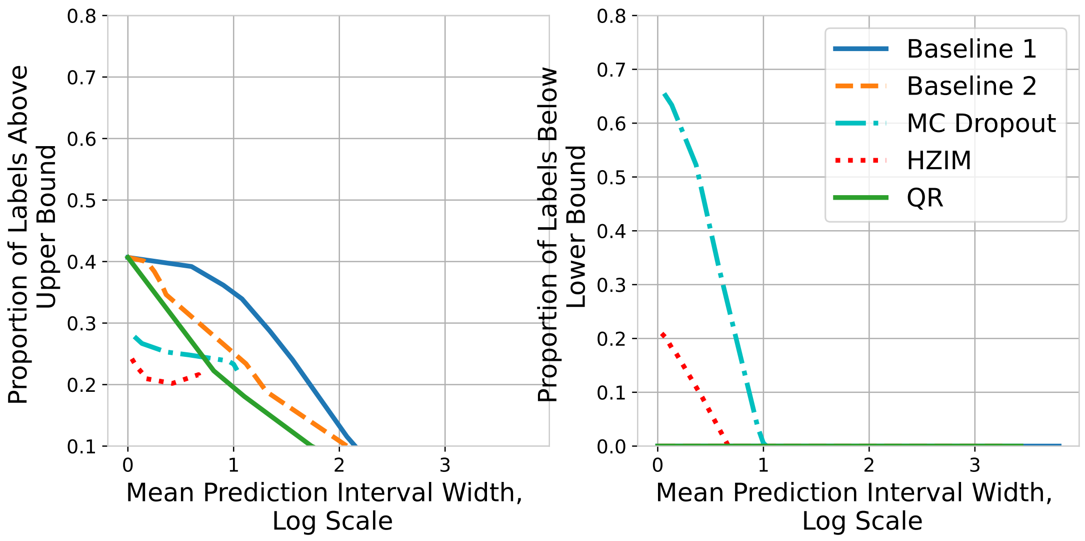


Figure 5.3: Two error types vs confidence interval length on “cold start” data. On the left are the proportion of data points for which the true value lies above the upper bound of the prediction interval (i.e. underestimation has occurred) and on the right are the proportion for which the true value lies below the lower bound (i.e. overestimation has occurred).

In Figure 5.3 we see the proportion of ground truth labels falling above and below the upper and lower bounds. For the baselines and quantile regression, all of the errors fall above the upper bound, but decrease as intervals grow in size. In contrast, HZIM and MC Dropout incur some lower-bound errors, though these decrease to zero when the intervals are large enough.

Performance of binary classification Metrics outlining the performance of the binary classifier are shown in Table 5.3 and in Figure 5.4. We trained the binary classifier with all available data, as there is only a slight data imbalance. The binary classifier achieves a good performance, with a precision around 80% on both zero and non-zero view content. The recall is slightly better on zero view content. This is unsurprising, since there is more zero view content in the dataset. Despite the good performance, binary classifier errors lead to some downsides of the HZIM method. These are reflected in some of the results shared above, and are discussed further in section 5.4.1

Binary Classifier Performance	
Metric	Metric Value
Precision, Zeros	0.80
Recall, Zeros	0.89
Precision, Non-Zeros	0.81
Recall, Non-Zeros	0.68

Table 5.3: Binary classifier performance metrics.

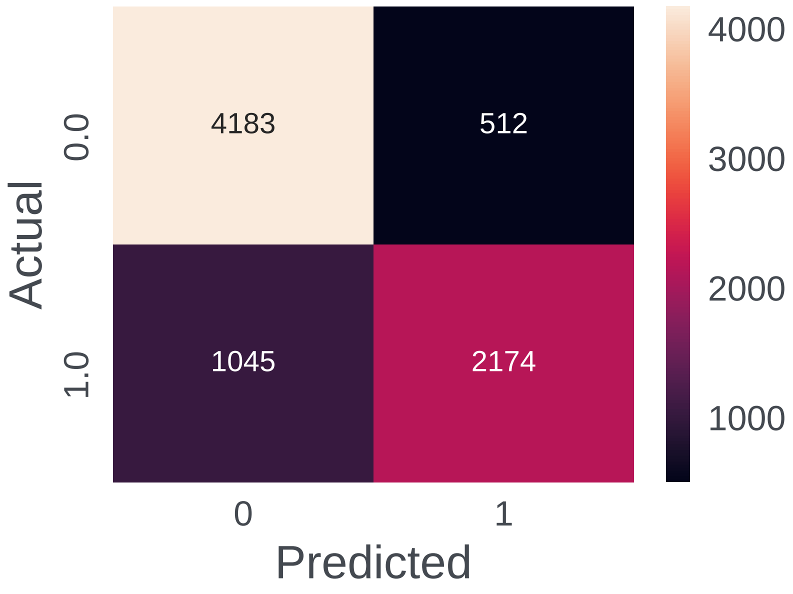


Figure 5.4: Binary classifier confusion matrix.

5.4.1 Discussion

These results demonstrate that quantile regression is able to outperform the baselines, while MC Dropout performs very poorly, and HZIM, despite marking an improvement over MC Dropout, performs better than quantile regression and the baselines in only some regions. It is worth exploring several of these details individually and in greater detail, especially those which contain surprising or counter-intuitive elements, before summarising a full interpretation of the results. Some elements of particular interest are as follows:

5.4.1.1 Poor performance of MC Dropout

Of all the methods, MC Dropout performs particularly poorly, starting with a low coverage of less than even 0.1 for the lowest interval width. The interval widths produced by MC Dropout are also severely restricted, reaching a maximum of 1.0 on our log scale. Unlike other methods, MC Dropout is not designed to directly calculate or predict quantiles of the label distribution, but instead calculates quantiles on bootstrapped predictions of the mean value. Therefore, the size of the interval width is determined by the variation in the sample distribution, and in all cases here, this appears to be very low, preventing large interval widths being reached.

Though MC Dropout was proposed with some theoretical guarantees which would allow this sample distribution to represent the true distribution, it may be the case that our highly irregular distribution does not fulfil the assumptions made in the original MC Dropout paper, preventing a representative sample distribution being generated. This could account for the inability to produce intervals larger than 1.0 on our log scale, and therefore a poor performance unable to exceed 0.8. These limitations are the motivation for developing an improved hierarchical method. However, this inability to produce large intervals carries over to HZIM, despite its improved performance.

5.4.1.2 Truncation of HZIM

In the results plots for HZIM compared against the baselines and quantile regression, the curve is truncated at a mean interval width of roughly 1.0. Although this intuitively seems as though it could be extended, in fact the size of the intervals is limited by the MC Dropout model, as discussed above. For quantile regression and baseline methods, the interval widths can be increased to the order of magnitude of the variance in the dataset. For MC Dropout and HZIM, the interval widths are constrained to the order of magnitude of the variance in the predictions made by the different versions of the model after random dropout, which is far less. Therefore, it is not possible to manually extend this curve by setting larger intervals as it is with the other methods, despite its promising trajectory.

5.4.1.3 Coverage for a mean interval width of zero

For both of the baselines and quantile regression, the coverage achieved at a mean interval width of zero is a fixed value at 59%. This occurs since an interval width of zero corresponds to setting both the upper and lower quantile to 0.5, which results with both an upper and lower bound of zero. Therefore, the coverage is exactly the proportion of the dataset which receives zero views, which in our dataset, is 59%.

As discussed above, MC Dropout calculates quantiles on a bootstrapped distribution which fails to capture this large presence of zeros. We therefore introduce a binary classifier in HZIM to improve the performance of MC Dropout, however the coverage at zero still falls short of 59%. This is due to errors introduced by the binary classifier. For some content which receives zero views, the binary classifier incorrectly identifies it as a non-zero data point. Therefore, the predicted interval is generated from an MC Dropout model which has been trained on the dataset with all the zeros removed, and the interval will not include zero. Therefore, HZIM cannot achieve 59% as some proportion of these 59% zeros will have been misclassified and coverage will not have been achieved.

5.4.1.4 Overall Performance of HZIM

From the above results and discussion, we can conclude that HZIM is able to improve vastly on the performance of MC Dropout on its own (for example, achieving a much higher coverage of around 55% at an interval width of zero where MC Dropout achieves less than even 10%). In one region of the curve, HZIM is even able to outperform the baselines and quantile regression (for a mean interval width between roughly 0.2 and 0.8). However, the performance of HZIM is restricted by both of its hierarchical components. At the lower end (mean interval width < 0.2) errors introduced by the binary classifier limit its performance in comparison to the baselines and quantile regression. At the higher end, the MC Dropout component is unable to produce large interval widths to extend the curve to higher coverage values. We conclude that HZIM is an improvement to MC Dropout, but is only better than other methods for a specific coverage range.

5.4.1.5 Final Results Discussion

Overall, we find that quantile regression is an improvement over the baselines, while MC Dropout is highly restricted by a seeming inability to adapt to the setting of a high proportion of content receiving zero views. In particular, MC Dropout generates only relatively small prediction intervals, and additionally achieves poor coverage relative to their size. By incorporating a binary classifier to separate out the content receiving zero views, as we do with our approach, HZIM, we are able to improve upon the coverage achieved by MC Dropout alone. However, HZIM is still restricted by the inability of MC Dropout to produce large confidence intervals. Despite performing better than the other approaches for an interval width greater than roughly 0.2, the performance of HZIM is also impeded at interval widths lower than this by errors introduced by the binary classifier, where content receiving zero views are incorrectly classified as receiving at least one view.

These results indicate that quantile regression, and to some degree HZIM, could be suitable for generating prediction intervals in the “cold start” prediction problem. In the case of HZIM, a strong performance is demonstrated for intervals of a certain size, but further work would be required to explore how intervals of a larger size could be generated if necessary. MC Dropout appears to be poorly adapted and therefore inappropriate for this problem, where simple statistical baselines are able to outperform it.

5.5 Conclusion

In this chapter, we have explored how uncertainty quantification, a technique providing additional information to a regular “point prediction” for improved user understanding, can be applied to predictive problems in a setting with social networks. In doing so, we have presented a preliminary evaluation of existing methods against baselines for generating prediction intervals for social media content popularity prediction. We then improved the performance of MC Dropout under some conditions by incorporating a binary classifier into a hierarchical model. Nevertheless, we have shown that the errors caused by the binary classifier introduce limitations. Based on these results, we have demonstrated that quantile regression and HZIM may be appropriate methods for generating prediction intervals in the “cold start” prediction problem.

Recent advances in parallel work have led to new methods for prediction interval generation which could be considered in future work. Among these are NOMU (Neural Optimization-based Model Uncertainty) [57] and Quality-Driven Deep Ensembles [143]. The first of these uses two separate neural networks, one of which makes the usual model prediction, and the other of which predicts the *model uncertainty*. Some multiple of this model uncertainty, determined by a calibration parameter, c, is both subtracted and added to the model prediction to generate lower and upper bounds. The second approach, Quality-Driven Deep Ensembles, builds on the work of Pearce et. al [124], which uses a single neural network architecture but generates three outputs: the model prediction, and the upper and lower bounds. An aggregation method is applied over several such neural networks. The quality-driven prediction intervals approach introduced by Pearce et. al was left out of our study due to its poor performance in initial experiments, and other more computationally demanding approaches such as deep ensembles [83] were also omitted. We prioritised MC Dropout over the deep ensembles approach as it is more scalable in storage, requiring only one copy of the model to be stored where the deep ensembles approach requires several.

For all methods in our study, we have calculated symmetrical quantile-pairs. However, other pairs of quantiles can be used to obtain a coverage of α, for example, $(0, \alpha)$ or $(1-\alpha, 1)$. Preliminary results suggested that attempting to optimise the coverage-interval width trade-off by varying the chosen quantiles did not have a significant effect on the results, despite our skewed distribution. Nevertheless, a full experimental exploration has been omitted and could be the subject of future work.

We now shift our attention from the application of explainability and uncertainty quantification techniques to social network analysis, to the application of such techniques to a new field. In particular, we are interested in optimisation and specifically, fitness landscape analysis. Like social network analysis, optimisation is a field where complex, stochastic algorithms are frequently employed, making it suitable for the application of explainability ideas found in the field of machine learning. Interestingly, this is also a field where graph (i.e. network) structure has been extracted and used in visual explanations for fitness landscape analysis, linking it in another way to our previous work on explainability for social network analysis. A key difference is that these graphs do not represent social interactions. In the next chapter, Chapter 6, we develop a novel visualisation technique to aid user understanding for fitness landscape analysis through the use of such graphs.

Chapter 6

Extrema Graphs for Fitness Landscape Analysis

Having explored the application of explainable AI and uncertainty quantification approaches to social network analysis, we now consider another domain outside of traditional deep learning: optimisation. As with social network analysis, optimisation is a field where highly complex algorithms are applied to solve challenging problems, often including the incorporation of stochastic processes. Thus, this makes optimisation another field highly relevant for the application of explainability approaches more commonly explored in machine learning.

One growing area of optimisation is fitness landscape analysis. A *fitness landscape* is a functional mapping between solutions to an optimisation problem, and their corresponding fitness values. In essence, this is a topological representation of the search space for the optimisation algorithm, which lends itself well to visual analysis. Since features of a fitness landscape can have considerable impact on the performance of a metaheuristic when applied to the optimisation problem, it can be highly beneficial to visualise these before developing problem-specific algorithms. For example, one common issue occurs when the algorithm becomes trapped in a local optimum, i.e. a location in the fitness landscape which is optimal in relation to all closely neighbouring regions, but not the sought globally optimal solution.

A prominent existing technique for visualising the fitness landscape is the Local Optima Network (LON) [118], for which an example can be seen in Figure 6.1. A LON represents the fitness landscape using a directed graph, where nodes are minima and edges represent a likely transition taken by the optimisation algorithm between these minima. The graphical representation allows for the use of network algorithms for visualisation, such as force-directed

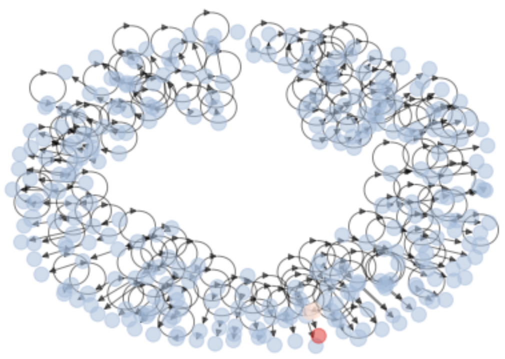


Figure 6.1: An example image of a LON, a prominent visualisation technique for fitness landscape analysis. This LON represents the Rastrigin function with 5-dimensional decision space using the default settings from Adair et al. [2]. The dark red node represents the global minimum. The series of red nodes show the basin of attraction for the global minimum, while other blue nodes represent basins for local minima. Arrows are also included showing the direction of travel in the basin-hopping algorithm used to construct the LON.

approaches. LONs were first proposed for the analysis of *NK* landscapes [118]. *NK* landscapes are mathematical models used to represent fitness landscapes in optimisation and evolutionary computation. In an *NK* landscape, each point represents a potential solution, which are all possible strings of length *N* in a given alphabet. The parameter *K* determines the landscape ruggedness. LONs have since gone on to be used in a wide range of combinatorial settings [119], as well in the continuous [2] and multi-objective [40] domains. A key concept in the construction of a LON is that of *neighbourhood*, and a number of approaches to this have been taken depending on the specific representation at hand. However, as with any visualisation, LONs are limited to the incorporation of certain information; for example, only the minima are included. Therefore, in this chapter we aim to develop a complementary visualisation approach that can be used alongside LONs to provide additional information. We focus on a method similar to this rather than on other topology-based visualisations (for example, those from Morse theory or those developed for volume multifield data) as it requires minimal information, i.e. the locations of extrema, rather than knowledge of the full manifold. It is also not required to visualise the entire space. In particular, we aim to *develop a novel visualisation approach to capture the general characteristics of function landscapes with input dimensions of three or higher (similar to contour plots for two-dimensional input spaces).* We expect that this will assist the design process of problem-specific optimisation algorithms by providing additional information not found in other visualisation methods.

To this end, we apply extrema graphs. These are a network representation of relationships between the extrema (i.e. the minima and maxima) of a function applied to a specific range of datapoints. Extrema graphs were originally used for isosurface extraction in volume visualisation, where transitions are captured between both maxima and minima embedded in two dimensions through dimensionality reduction techniques. For dimensionality reduction, we use multidimensional scaling in our prototype for fitness landscape visualisation. The inclusion of both minima and maxima means that regions in which the distance between the minima and maxima is small can be identified, such that the ensuing small headroom makes passing between them difficult for solutions. As well as including the extrema as nodes in our network visualisation, we also include regularly spaced points on the lines directly connecting these extrema, which we refer to as *edge nodes*. This allows us to visualise features of the landscape between the extrema themselves. We then visualise the resulting network by projecting it into two dimensions with multidimensional scaling (MDS), which allows for preservation of the distances between extrema, a characteristic which is lost in other layouts such as force directed graphs. Additional visual encodings are used to supply information, such as the colouring of both extrema nodes and edge nodes to represent fitness value, and the discrimination of extrema nodes from edge nodes through size.

We assess our proposed extrema graph technique on a number of well-known benchmark problems, which we analyse to understand the information that can be gleaned from their usage. We then further evaluate our approach through an expert interview, for which we gained ethical approval. We report the qualitative comments, both positive and negative, of this expert, as well as performing a quantitative test of the expert's understanding. The expert was shown to have a strong ability to match extrema graphs to the corresponding fitness landscape which they represent. This chapter is based on work from a paper presented at the LAHS workshop at ACM's Genetic and Evolutionary Computation Conference (GECCO) 2023 [141].

6.1 Extrema Graphs

Similarly to a LON, extrema graphs use a graphical representation where nodes consist of extrema (in the case of LONs, only the minima) and edges represent some metric of distance between these extrema. However, extrema graphs differ from LONs in both the construction and the visualisation of this graph, for example, in the inclusion of both maxima and minima. In this section, we will provide a detailed description of the extrema graph approach as applied to a given fitness landscape. The incorporation of all extrema is infeasible for large landscapes with many extrema due to the computational expense in identifying them. To generate our extrema graphs, we have therefore used the *Niching Migratory Multi-Swarm Optimiser* (NMMSO) [39] to sample extrema to include in the graph. NMMSO is an algorithm which identifies multiple maxima of multi-modal problems by using a swarm of individual optimisers which work in different regions of the search space. Swarm elements are able to migrate away from their parent swarm if they are identified as being in the vicinity of a separate peak, and swarms are able to merge if they are identified as being concerned with the same peak. Though NMMSO is a good multimodal optimiser capable of finding many, if not all, of the extrema in our benchmark problems, it could be substituted for another approach. In this chapter, we focus on developing a novel visualisation technique containing information beneficial for the design process of algorithms which cannot found in other visualisation approaches, rather than optimising the process for identifying extrema.

6.1.1 Graph Construction

The graph is then constructed from these extrema in the following steps. A full overview of the graph construction process is contained in Algorithm 6.1. The process begins by running NMMSO $2N$ times on our function of interest, $f : \mathbb{R}^m \to \mathbb{R}$, where there are N runs for the minima and N for the maxima. This identifies the extrema, which we then initialise as the nodes for our graph, G. Initially, we do not have any edges.

Next, an edge is added between two nodes where the Euclidean distance between the corresponding extrema is less than a threshold, whether these extrema are minima or maxima. The threshold t is given by a hyperparameter ρ multiplied by the Euclidean distance between the bounds of the search space, such that $t = \rho \|b_u - b_l\|$ where b_u and b_l vectors are the upper and lower bounds of the search space. We call the hyperparameter ρ the *radius percentage*. This hyperparameter can take values between 0 and 1, thus determining the proportion of edges which are included, and can be chosen by the problem owner based on their requirements when designing an optimiser (for example, if it depends on a certain neighbourhood value) as well as their own personal perception of the visualisation.

Once the edges have been determined, they are then replaced with a line of nodes, which we call *edge nodes*. In the visualisation there are therefore no edges in the graph, however the extrema nodes have a 10x larger visual representation than the edge nodes, such that these take the appearance of edges between the extrema nodes. This is demonstrated in the example extrema graph of Figure 6.2. The benefits of drawing the edges as a sequence of nodes

Algorithm 6.1 Extrema Graph Construction[1]

```
Input: For a problem with dimensionality $m$: function $f : \mathbb{R}^m \to \mathbb{R}$; upper and lower bounds
    on the search region, $b_u, b_l \in \mathbb{R}^m$; number of fitness evaluations, $2N$; radius percentage, $\rho$;
    number of edge nodes per edge, $e$
Output: Extrema graph and 2D node locations
 1: Run NMMSO algorithm for $N$ fitness evaluations to obtain $n_{max}$ maxima
 2: Run NMMSO algorithm for $N$ fitness evaluations to obtain $n_{min}$ minima
 3: Initialise graph $G = (V,E)$ with $|V| = n_{max} + n_{min}$ and $E = \emptyset$
 4: Set the threshold, $t = \rho \|b_u - b_l\|$
 5: Initialise location matrix $L \in \mathbb{R}^{n \times |V|}$ such that for every node $i \in V$, $L_{\_,i} = o_i$ where $o_i$ is
    the location of optimum $i$ in the landscape
 6: for pairs of extrema, $o_i, o_j$ do
 7:     If $\|o_i - o_j\| < t$, add edge $(i, j)$ to $E$
 8: end for
 9: for edges $(i, j)$ in $G$ do
10:     Add $e$ nodes to $V$ and corresponding locations in the
        search landscape to $L$ such that these locations are evenly
        spaced along the line connecting $o_i$ and $o_j$
11:     Remove $(i, j)$ from $E$
12: end for
13: Perform MDS on L to reduce to 2 dimensions
14: Output $G = (V, \emptyset)$ and $L$
15: Note: $\|\cdot\|$ denotes the Euclidean norm in $n$ dimensions
```

[1]An implementation of the algorithm in Python is available at https://github.com/sophiefsadler/extrema_graphs.

are twofold: firstly, this allows for easy variation of the colour along the connecting line between extrema to visualise changes in fitness value, corresponding to specific locations in the landscape. Secondly, it provides a visual indication of how the search space is folded by the dimensionality reduction. These benefits offset the downside of additional visual clutter. Each of these edge nodes also has an associated location in the search landscape, such that they are linearly spaced along the line connecting the two relevant extrema. The associated locations in the search landscape of both the extrema nodes and the new edge nodes are stored in a location matrix, L, as described in lines 5 and 10 of Algorithm 6.1.

This process produces the final graph topology, where we have a number of extrema nodes and a number of edge nodes, each having a corresponding location in the search landscape. The final step before we can visualise the extrema graph is to translate these locations in high-dimensional space to locations in 2-dimensional space. In our extrema graph prototype, we use MDS [81] to perform this translation as it aims to preserve relative distances, though there

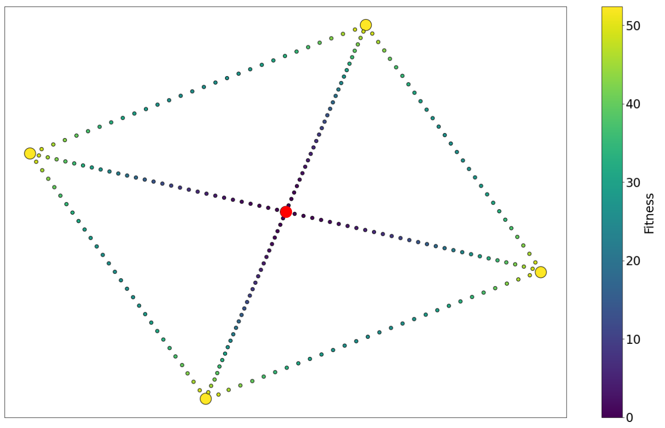


Figure 6.2: Example Extrema Graph (for the Sphere function in 2D)

are many other relevant dimensionality-reduction techniques which could be applied, including Landmark MDS [32] or t-SNE [165], which has previously been applied for visualising LONs [167]. MDS performs well with our choice of graphical structure as its preservation of relative distances increases the chance that edge nodes lie in a sequence that can be easily understood to follow a path between two extrema nodes, thereby making it clear which two nodes the edge is connecting. MDS is also a good choice when the structure of the manifold is unclear. The chosen dimensionality-reduction method should be applied to all nodes - both extrema nodes, and edge nodes, to determine their location in the extrema graph visualisation. It should be noted that, while MDS aims to preserve pairwise distances when projecting to lower-dimensional space, it is impossible to do so perfectly and therefore there may be some unintuitive distortion of the space. This motivates the use of edge nodes to aid understanding.

6.1.2 Visual Encoding

The MDS drawing of the extrema graph is visualised directly. However, in order to distinguish minima, maxima, and edge nodes, as well as to provide other useful information about the search landscape topology, we use additional visual channels in our encoding.

Firstly, extrema nodes are larger than edge nodes to distinguish them as local extrema. All nodes are then coloured according to an isoluminant colour scheme representing the fitness value at the corresponding location in the search landscape. As we know the locations of all nodes in the high-dimensional search space (these are stored in matrix L in Algorithm 6.1), we can evaluate the function on these locations to obtain these fitness values. In our prototype, we use Matplotlib's Viridis colour scheme, which varies from yellow for the highest values through green, blue and finally purple to the lowest values. The Viridis colour scheme is designed to improve graph readability for readers with common forms of color blindness, making it widely accessible for all audiences. In addition, it spans many colours, making differences easy to see, and is perceptually uniform, meaning that values close to each other have similar colours and values far away from each other have very different colours. This works well in the extrema graph setting to show smooth transitions from areas of high function value to low function value.

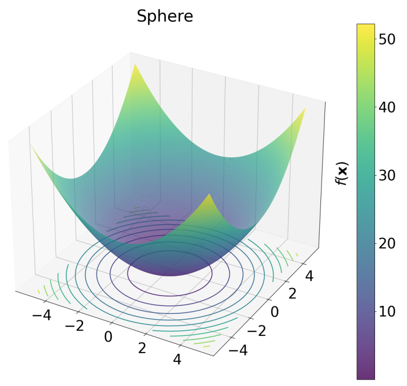
When using this colour scheme, minima are usually represented in purple (low values) and maxima in yellow (high values), though this is not guaranteed as local minima or maxima may not have fitness values as low or high as the global minima and maxima. Edge nodes are also coloured according to this scheme, allowing for interpretation of the shape of the landscape between two extrema. The global minima are marked separately in red to distinguish them from other minima, which may be globally non-optimal despite appearing similar in colour when using the Viridis colour scheme. Red was chosen as it is a good highlighting colour [114] which differs considerably from those which appear in the Viridis colour scheme.

An example extrema graph is displayed in Figure 6.2, to demonstrate these visual encodings. The scale on the right-hand side of the plot displays the fitness values which the Viridis colour scheme represents: 0 at the lowest, purple end, and 50 at the highest, yellow end. Thus, we can see in the extrema graph itself that there are four maxima in the corners of the graph, which are connected by smaller edge nodes to the global minimum at the centre, represented in red. None of the visual features described here are present in LONs, and are therefore a key differentiator of the two methods. In the case of a LON, we would see only a single node for the sphere function in 2D, represented without colour.

6.2 Experimental Results and Illustrations

In order to demonstrate the efficacy of extrema graphs for understanding search landscapes, we demonstrate them as applied to a number of well-known, continuous benchmark functions: Sphere, Rastrigin, Schwefel, Ackley, Griewank and Rosenbrock. These are defined as follows.

Sphere Function



The Sphere function is defined in n dimensions as:

$$f(x) = \sum_{i=1}^{n} x_i^2$$

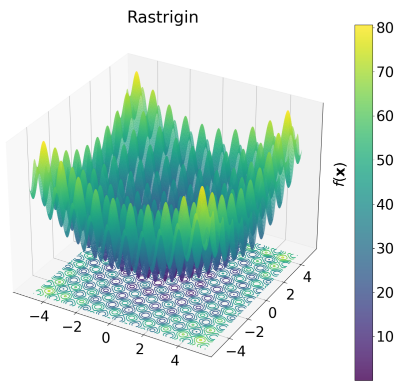
Our search bounds were defined by the hypercube $x_i \in [-5.12, 5.12]$, and in this region there is one global minimum at $x = 0$ with a function value of $f(0) = 0$.

Rastrigin Function

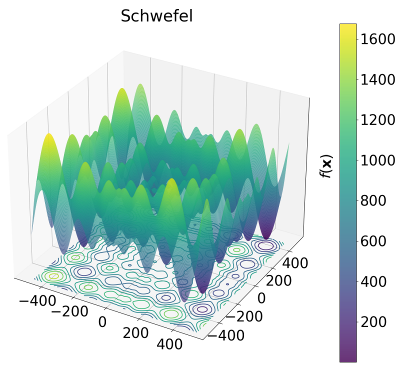
The Rastrigin function is defined in n dimensions as:

$$f(x) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right)$$

Our search bounds were defined by the hypercube $x_i \in [-5.12, 5.12]$, and in this region there is one global minimum at $x = 0$ with a function value of $f(0) = 0$.



Schwefel Function



The Schwefel function is defined in n dimensions as:

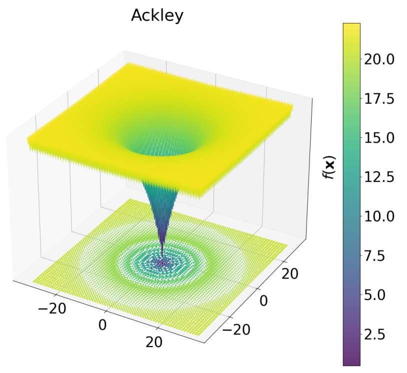
$$f(x) = 418.9829n - \sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|})$$

Our search bounds were defined by the hypercube $x_i \in [-500, 500]$, and in this region there is one global minimum at $x = 420.9687 = (420.9687, \ldots, 420.9687)$ with a function value of $f(420.9687) = 0$.

Ackley Function

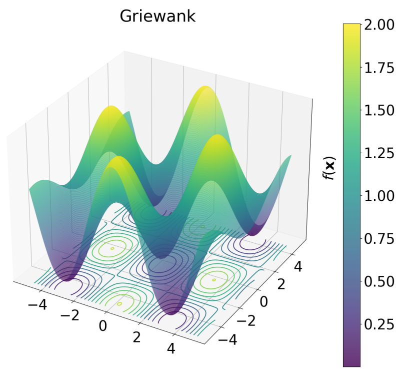
The Ackley function is defined in n dimensions as:

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1)$$



Our search bounds were defined by the hypercube $x_i \in [-32.768, 32,768]$, and in this region there is one global minimum at $x = 0$ with a function value of $f(0) = 0$.

Griewank Function



The Griewank function is defined in n dimensions as:

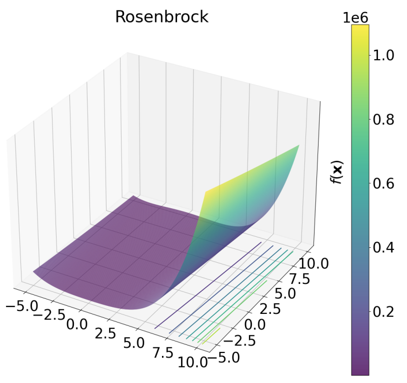
$$f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

Our search bounds were defined by the hypercube $x_i \in [-5, 5]$, and in this region there is one global minimum at $x = 0$ with a function value of $f(0) = 0$.

Rosenbrock Function

The Rosenbrock function is defined in n dimensions as:

$$f(x) = \sum_{i=1}^{n-1} \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$$



Our search bounds were defined by the hypercube $x_i \in [-5, 10]$, and in this region there is one global minimum at $x = 1 = (1, \ldots, 1)$ with a function value of $f(1) = 0$.

Radius Percentage Values		
	ρ	Search Bounds
Sphere	0.75	$[-5.12, 5.12]^D$
Rastrigin	0.08	$[-5.12, 5.12]^D$
Schwefel	0.1	$[-500, 500]^D$
Ackley	0.05	$[-32.768, 32.768]^D$
Griewank	0.25	$[-5, 5]^D$
Rosenbrock	0.75	$[-5, 10]^D$

Table 6.1: The value of the "radius percentage" hyperparameter ($\rho \in (0,1]$) used during graph construction, alongside the bounds of the search region explored.

We have produced an extrema graph for each function in $2, 3$ and 5 dimensions to demonstrate the performance and variation across these differing numbers of dimensions. Additionally, the extrema graph for the Sphere function in 10 dimensions can be found in the corresponding section below, as an example of how the approach scales to larger dimensional spaces. In the graph construction, we used $20,000$, $30,000$ and $50,000$ fitness evaluations respectively to generate extrema graphs in $2, 3$ and 5 dimensions. The radius percentage hyperparameter, ρ was chosen for each function individually; these values are as in Table 1. As this value controls the number of edges included in the extrema graph, using a larger value of this hyperparameter allows more information about the fitness landscape to be captured and included. However, for functions with large numbers of maxima and minima, a large radius percentage can lead to an extrema graph with too much visual clutter to be understood, and also drastically increases computation time. For our evaluation, we chose this value experimentally, by testing a range of values and selecting the one for which the resulting extrema graph contained the best quantity of valuable information in the eyes of our interpretation. These values were then fixed across all dimensions so that all extrema graphs for a given function could be directly compared. Note that the NMMSO runs need not be rerun in order to test new values of ρ, so these were performed only once.

6.2.1 Benchmark Visualisations and Interpretation

For each of the benchmark functions, we present a figure containing four subplots: the conventional surface plot already introduced in the previous subsection, and an extrema graph for the function in 2D, 3D and 5D.

Sphere In Figure 6.3, we see the results for the Sphere problem – the problem with the simplest fitness landscape considered herein. Figure 6.3B shows an extrema graph of a 2D

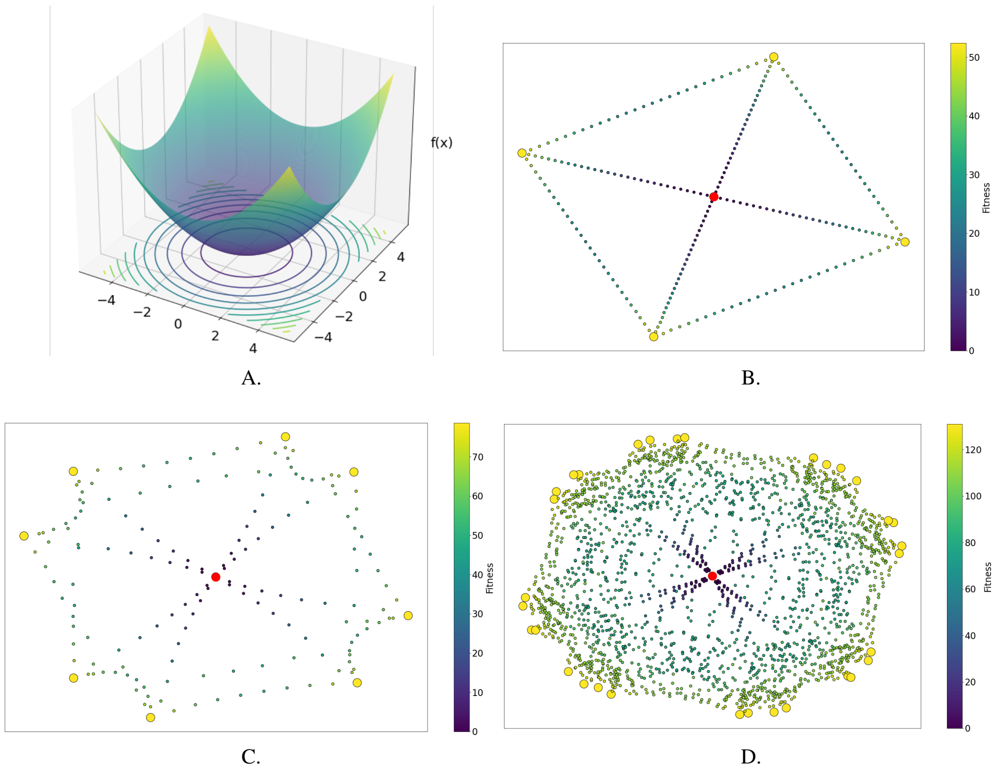


Figure 6.3: A surface plot for the Sphere function (subfigure A), in addition to extrema graphs for the Sphere function in 2D, 3D and 5D (subfigures B, C and D respectively).

landscape. The graph consists of five extrema nodes: four nodes shown in yellow, with the central node being shown in red. As described in Section 6.1.2, the red node corresponds to the global minimum, while yellow nodes have high fitness value (so are maxima). The larger extrema nodes are connected by smaller edge nodes, also coloured according to fitness. By contrast, a LON for this case would comprise of a single node (representing the global minimum) with no edges. Here it is possible to observe the maxima, which correspond to the corners of the landscape shown in Figure 6.3A; the scale of the fitness in those nodes is determined by the bounding box selected when instantiating the problem. The colour of the internal edges, connecting the maxima to the minimum, show a steady decrease in fitness (the colour gradually progresses from yellow to purple). This represents the smooth slope of the landscape, which has no local minima. The outer edges decrease to a trough before climbing back towards the adjoining maxima. This is a graphical representation of how the surface cuts the landscape's bounding box, which is the U-shape that can be seen at the edge of the landscape in Figure 6.3A.

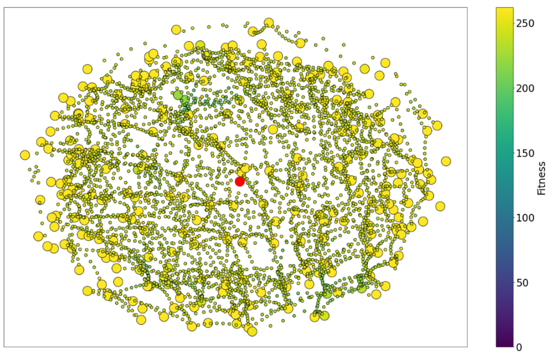


Figure 6.4: Extrema Graph for the Sphere function in 10D

Figure 6.3C shows the corresponding 3D Sphere landscape. The characteristics highlighted in the 2D case can be seen here. In this case, there are eight nodes representing maxima. This is to be expected, as any instance of this problem will feature a hypercube comprising 2^D corners, which form the maxima. Each maximum is again directly connected to the minimum via an edge, which again exhibits the steady decrease we would expect from the smooth landscape. Each adjacent maximum is again connected by an edge that shows the U-shape formed by the bounding box, however in this case there are additional edge nodes. This is a result of the compression of a 3D landscape into two dimensions, and shows connections between maxima in the third dimension. The same topology is visible in the 5D graph shown in Figure 6.3D, though the number of edge connections in this case makes it difficult to observe direct connections between them. It is, however, still possible to identify the smooth progression from maxima to minimum by following the colour of the edges as they move toward the centre of the graph. There are again $2^5 = 32$ nodes representing maxima.

For the Sphere function, we also generated an example of the extrema graph in 10 dimensions, to show how the approach scales to higher dimensions. This can be seen in Figure 6.4. As with the other plots, there is a single minimum surrounded by maxima, though there are now too many of these to easily count, and the number of edge nodes renders it infeasible to identify direct connections. Due to distortion of the space, we no longer see the smooth progression to the centre of the graph. From this we can see the beginnings of extrema graph limitations as the number of dimensions grows, though some key elements (the presence of a single minima surrounded by many maxima) can be identified regardless.

Rastrigin Figure 6.5 illustrates the Rastrigin examples. Again, we display the conventional landscape view in Figure 6.5A and the corresponding 2D extrema graph in Figure 6.5B. In

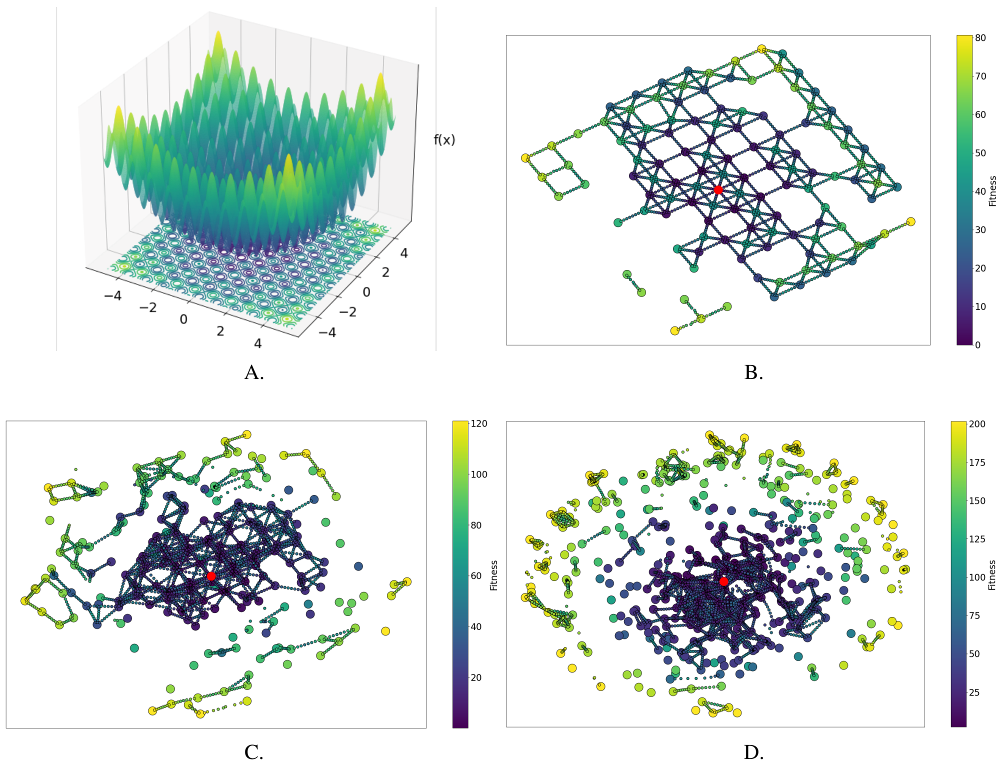


Figure 6.5: A surface plot for the Rastrigin function (subfigure A), in addition to extrema graphs for the Rastrigin function in 2D, 3D and 5D (subfigures B, C and D respectively).

some ways, the visualisation is not dissimilar to the Sphere graph shown in Figure 6.3B. There are four maxima, again shown in yellow, though we note that the scale is different as it depends on the fitness values in a specific graph. The principal difference is the presence of a number of local minima, represented as nodes within the graph. The Rastrigin objective function is formed of the sum of squared decision variables (accounting for the similarities to the Sphere case) which is offset by a cosine term introducing regular local minima (accounting for the additional extrema nodes). The regularity of the minima can be seen by the way in which the nodes have been placed into a grid arrangement. Clearly, the graph is missing regions of the landscape. In this case, we know this is because of the stochastic nature of the multimodal optimiser that has been used to sample the space as the cosine term guarantees regular local minima, as visualised in Figure 6.5A. One option that could be used to address this and provide a more complete map would be to run multiple repeats of the optimiser in the way that a LON is constructed, however, this would increase the computational cost which is currently minimised by the use of only two optimiser runs, one for the maxima and one for the minima.

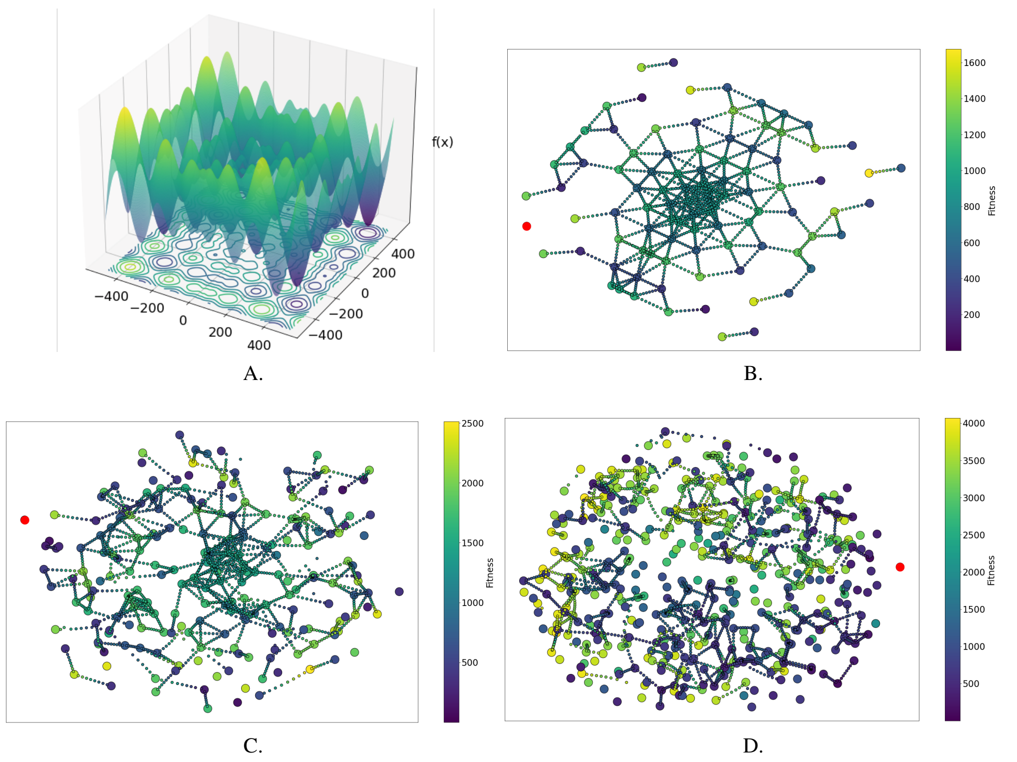


Figure 6.6: A surface plot for the Schwefel function (subfigure A), in addition to extrema graphs for the Schwefel function in 2D, 3D and 5D (subfigures B, C and D respectively).

An interesting characteristic visible from the edge nodes is that the problem is non-convex. This is shown by the regular progression of colour between nodes.

The 3D case shown in Figure 6.5C exhibits similar characteristics to the 2D case, albeit in a less uniform layout because of the distorted projection of the additional dimension. The minimum is still shown at the centre of the graph; however, the graph shows fewer connections between the inner core and the outer maxima. This again is likely to be an artefact of the way in which the multimodal optimiser has traversed the space while sampling. This effect is even more pronounced in the 5D case (Figure 6.5D).

Schwefel The Schwefel problem is similar in construction to Rastrigin. Figure 6.6B indicates this in that the graph has a similar regular structure. Noise in this case is introduced by a sine term rather than the cosine, and taking the square root of the absolute value of each decision variable introduces local minima at varying heights, rather than the steadily decreasing minima of Rastrigin. In Figure 6.5B, the 2D Rastrigin extrema graph, the further a minimum is toward the global minimum the lower the fitness value. In Figure 6.6B, for Schwefel, the minima

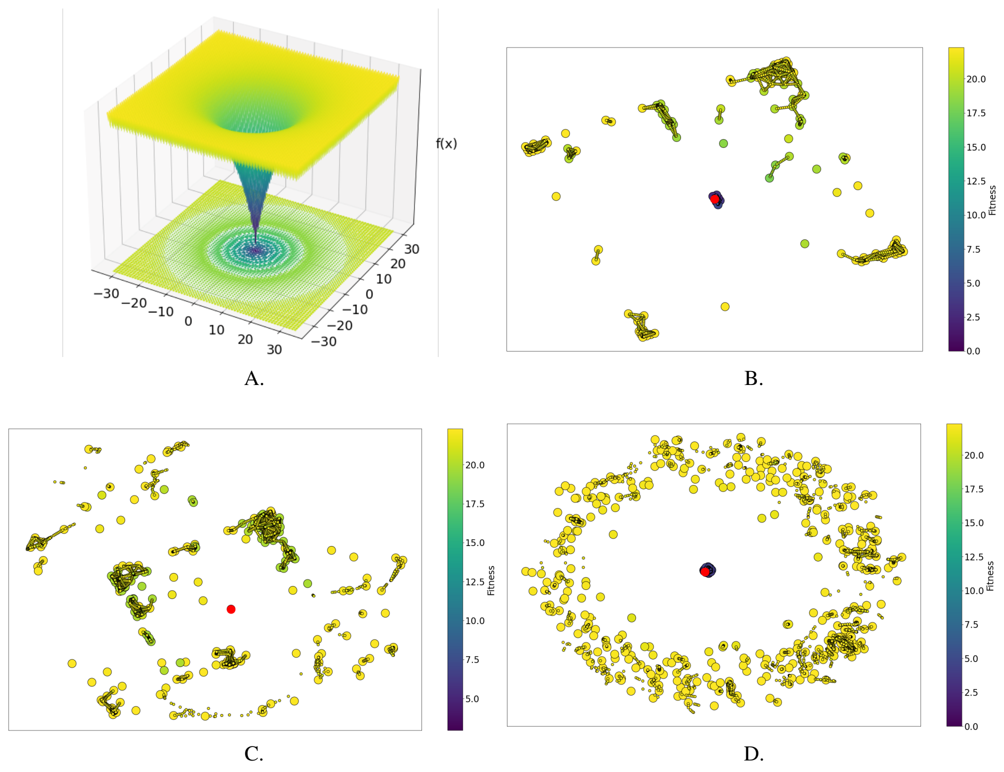


Figure 6.7: A surface plot for the Ackley function (subfigure A), in addition to extrema graphs for the Ackley function in 2D, 3D and 5D (subfigures B, C and D respectively).

increase and decrease as they progress to the global minimum. Another interesting aspect of this problem compared to Sphere and Rastrigin is that it only has one maximum. This is placed at the opposite end of the graph from the minimum, which is again shown in red. While these characteristics are, as has been the case in the other problems, visible in the 3D case (Figure 6.6C), they are more difficult to observe in the 5D case (Figure 6.6D). We theorise that this is because of the effect of the more rugged landscape on the sampling process, leading to fewer connected edges which in turn leads to a less precise embedding with MDS.

Ackley Figure 6.7 illustrates extrema graphs for the Ackley problem. The 2D graph in 6.7B follows the typical layout we have seen in previous cases, with maxima arranged around a central minimum (again, shown in red). The regions between the external maxima and the minimum are sparse. This, in combination with the high concentration of minima identified near the global minimum, indicates that the optimiser has not struggled to locate the problem's central funnel. Those maxima and local minima that have been found are arranged in a some-

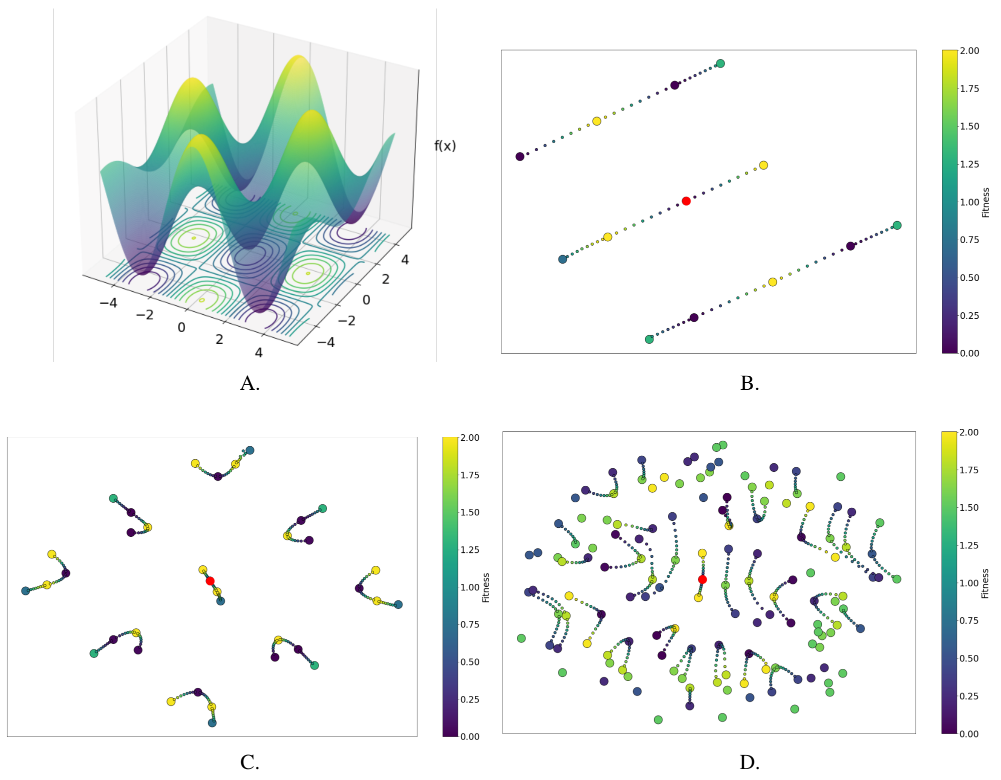


Figure 6.8: A surface plot for the Griewank function (subfigure A), in addition to extrema graphs for the Griewank function in 2D, 3D and 5D (subfigures B, C and D respectively).

what regular fashion in a square shape around the central global minimum, which is as would be expected for the problem. When considering the 3D case (Figure 6.7C) the number of local minima nodes found near the global minimum has reduced considerably. This indicates that the funnel is becoming steeper, and the optimiser is less likely to be trapped by local minima once it locates solutions within the funnel. Having increased to 5D in Figure 6.7D, the distance between the funnel and the maxima nodes is increased. This is due to the MDS projection, which has placed nodes towards the corners of the projection as it seeks to keep nodes that are distant in the 5D space distant in the corresponding 2D embedding.

Griewank As can be seen in Figure 6.8A, the Griewank landscape is highly regular. This regularity has resulted in a distinct extrema graph (Figure 6.8B). As seen previously, the global minimum is placed toward the centre of the graph. The local minima and maxima are placed around it, organised into lines. These lines cut across the fitness landscape, and the peaks and troughs visible in the 2D landscape (Figure 6.8A). This effect is visible in the 3D case, shown

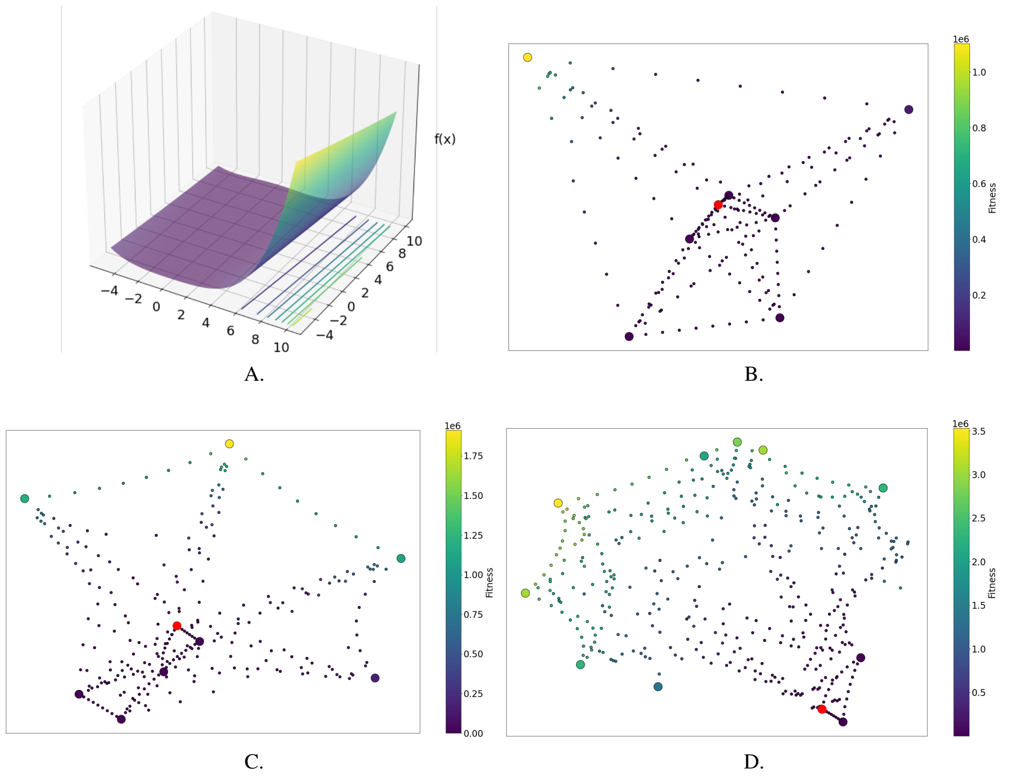


Figure 6.9: A surface plot for the Rosenbrock function (subfigure A), in addition to extrema graphs for the Rosenbrock function in 2D, 3D and 5D (subfigures B, C and D respectively).

in Figure 6.8C, and the 5D case, shown in Figure 6.8D. In both cases, compressing the extra dimensions into a two-dimensional extrema graph has resulted in curved edge nodes. While in the 2D and 3D cases all of the maxima and local minima have been located, the optimiser has not sampled the entire 5D landscape.

Rosenbrock The final problem we consider is Rosenbrock. As shown in Figure 6.9A, this problem is characterised by a deep trench of reasonable fitness, only a specific portion of which actually achieves the global minimum. Figure 6.9B illustrates the 2D extrema graph. Here, there is a single maximum, due to a slight tilt in the fitness landscape that biases the value of one of the maxima to be marginally higher than the other corners (however, by expanding the bounds of the problem the opposite side of the landscape in the x_2 axis becomes visible). In the lower right-hand corner of the graph are a set of connected local minima at values close to the global minimum, as well as the global minimum itself. This indicates the trench, and this characteristic can be seen in Figures 6.9C for the 3D case and 6.9D for the 5D case. In both the 3D and 5D cases the optimiser has located maxima in other corners, and in both cases these are connected to the trench (and, in some cases, the global minimum itself) by decreasing edge nodes as was the case in the Sphere example.

6.3 Expert Interview

We also conducted an interview with an FLA expert to ascertain whether extrema graphs as we have presented them can be easily understood, and whether they fulfil the intended purpose of capturing the general characteristics of function landscapes in high dimensions, so that they can be easily recognised. Expert interviews are a common assessment of visualisation techniques, usually consisting of studies with a small number of experts. Five or fewer experts can find the majority of issues with a visualisation [160]. In line with standard ethical practice, anonymity of participants/experts is protected. In this case, we perform such an evaluation with one expert and will expand this evaluation in future work.

Experimental Design The interview took the following format. We began by introducing the premise of our approach and the aims of our research. This was followed by some technical discussion of the methodology, and finally by provision of examples in the form of the Sphere function extrema graphs in 2D, 3D, 5D and 10D.

In the second stage, we presented randomly ordered extrema graphs and randomly ordered surface plots for the remaining benchmark functions in 2D (Ackley, Griewank, Rastrigin, Rosenbrock and Schwefel), and asked the expert to match each extrema plot with one of the surface plots. At this stage, we did *not* tell the expert whether the answers were correct, but immediately presented randomly ordered extrema graphs for the benchmark functions in 3D and asked the expert to again match these to the surface plots. Although the expert was *not* informed of the correct answers for the 2D plots, we cannot rule out that a learning effect could take place to improve the expert's performance on the subsequent 3D plots. This approach was chosen to assess the expert's understanding as it gives us a quantitative evaluation of performance, in contrast to other qualitative elements of the expert interview. The interview was concluded with an exploration of the benefits and drawbacks of our approach as perceived by the expert.

Participant Response For the 2D functions, the expert correctly matched the Ackley, Rastrigin and Schwefel graphs to their surface plots without difficulty, but deliberated over the matching of the Griewank and Rosenbrock functions and ultimately matched these incorrectly. For the 3D functions, the expert once again deliberated over the matching of the Griewank and Rosenbrock functions, but this time got the matching of all 5 functions correct. This indicates that the expert was able to gain enough insight from the extrema graphs to identify the corresponding functions. Increased exposure may have led to the improved performance in the second round, despite the expert not being told the correct answers for the 2D functions inbetween the two rounds.

During the discussion in the final stage of the interview, the expert identified the ability of extrema graphs to capture the ruggedness of the landscape, which is derived from the concise representation of the spatial relationship of the extrema, as their main benefit. The expert believed that extrema graphs can provide an overview of the functions in higher dimensions, in particular with respect to their function variability. Furthermore, the expert highlighted that this method may be useful in visualising paths of optimisers for analysing their performance on specific problems.

Regarding drawbacks, the expert noted that the appearance of the extrema graphs may be sensitive to methodological details. The expert highlighted with the following comments some examples of possible sensitivities: the use of edge nodes may introduce bias into the MDS procedure towards regions where there are more edges (i.e. where there are more maxima and minima closer together in Euclidean space); interpolating edge nodes between extrema “as the crow flies” restricts the view of the landscape to the regions directly connecting extrema; and finally, the NMMSO algorithm expends some proportion of its work towards finding the global optimum, so it may ignore certain optima.

We agree that our method is highly dependent on the efficacy of the multimodal optimiser in finding distinct extrema. Nonetheless, we believe the sensitivity due to this is minimal in our demonstration, especially due to the high number of function evaluations used for the known synthetic problems explored in this chapter, and focus here on evaluation of the visualisation technique itself.

During this informal discussion of benefits and drawbacks, the expert provided insights that may lead to potential further work, detailed in section 6.4 below. We conclude based on the expert’s feedback that extrema graphs provide the benefit of capturing a realistic idea of landscape ruggedness, while potentially being sensitive to details in the methodology used to generate these visualisations.

6.4 Discussion

The extrema graphs of the benchmark functions and descriptions of their visual features in Section 6.2.1 demonstrate how they can be used to represent fitness landscapes with a 2-dimensional visualisation. Through these examples, we encounter some specific points for further discussion and exploration. For instance, the 10D extrema graph for the sphere function, presented in Figure 6.4, indicates some limitations of this approach as the number of dimensions grow, which represents one direction of exploration for future work. It would be beneficial to identify further whether extrema graphs experience limitations for other benchmark functions as the number of dimensions grows, and to identify ways to circumvent this problem. In this setting with a high number of dimensions, the trade-off between including a large amount of information while avoiding visual clutter becomes more important, motivating further investigation into how we can quantify an optimal value of ρ to balance this trade-off. We also discovered with the Schwefel function that some characteristics became harder to observe in the 5D case, which we hypothesised was due to the effect of the more rugged landscape on the sampling process. The presence of fewer connected edges in this case could also be addressed if we were able to quantitatively identify an optimal value for ρ. Interestingly, the identification of a methodology for automatically determining the value of the radius percentage hyperparameter was an avenue of future work suggested by the expert during the expert interview. One approach for doing so could be to identify an information-content metric with which to measure the performance of the extrema graph visualisations.

In addition to this suggestion, the expert also offered visual adaptations which could be considered for future work. Specifically, the expert suggested replacing the edge nodes with edges consisting of coloured line segments. These segments could represent fitness value through use of colour as in our version of the extrema graph visualisation, while providing alternative information to the current implementation during the embedding stage. In this case, rather than including points along the edges in the MDS process, only extrema would be embedded, leaving the connecting edges to overlap in the final visualisation. In this case, intersecting edges with different colours would provide information about the folding of the search space during dimensionality reduction.

Some limitations of extrema graphs are accepted here as a trade-off for associated benefits. For example, the extrema included in the graph are limited to those identified by NMMSO, however the use of just two runs of this optimiser keeps the computational complexity low. Similarly, the visual clutter introduced by edge nodes is accepted for the benefit of the information gained about the general characteristics of the landscape, and how these change for a given value of ρ.

One of the commonly cited reasons for a visualisation, especially in the field of *explainable AI* (XAI), is that we seek to represent a complex concept to a lay audience. The expert interview conducted in this study has shown that extrema graphs can convey useful information characterising problem landscapes to experts, however it would be beneficial to scale the human-centred evaluation by interviewing more experts. Since access to experts is limited, this is an inherent limitation to expert interviews. Additionally, while we have not tested whether our visualisations can be understood by a non-expert, it would likely be more difficult for a non-expert to perform such analysis. Two avenues of future work are therefore to explore the extent to which these are accessible to non-expert users, and – in the event that they are not intuitive – to perform an investigation into approaches by which the technical barrier to entry could be reduced.

Faster methods for multidimensional scaling could be used to draw our extrema graphs, similar to stressed-based layout methods in graph drawing [69] which have been accelerated via many methods [48, 74, 80, 190]. Using a more scalable MDS algorithm would allow extrema graphs to scale to larger data sets. For our prototype presented in this chapter, we have chosen MDS as it preserves relative distances and we have found that this often ensures that the connectivity of the edge nodes is retained, enabling us to see which extrema have been connected by these. It is also a good choice when the structure of the manifold is unclear.

Beyond enhancing the visualisation, there are several ways this study could be extended. A larger range of problems should be explored, including those with discrete representations, in contrast to the continuous benchmarks considered here. A further area of exploration is the use of extrema graphs for visualising multi-objective search spaces. A key consideration will be how to represent solution quality, and whether trade-offs between solutions that define such problems are usefully visualised in an extrema graph.

6.5 Conclusion

In this chapter, we have presented a novel methodology for visualising fitness landscapes through the use of extrema graphs. We have shown that the incorporation of both minima and maxima can provide valuable information, and that appropriate use of dimensionality reduction is key for visualising ruggedness of the landscape. The use of sampled “edge nodes” replacing edges assist with giving a complete picture of this ruggedness. Through an expert interview, we have investigated the interpretation of these graphs by an expert, and showed that they were able to match the extrema graphs to the corresponding landscape in most cases.

Though we have explored a number of improvements that could be made in the discussion section, we will describe avenues for future work in greater detail in the next chapter, Chapter 7, where we will further cover areas of future research relating to all chapters of this thesis.

Chapter 7

Future Work

In this thesis, we have applied a set of explainable AI and uncertainty quantification techniques to a specific selection of domains outside the realm of traditional deep learning, with a focus on social systems and networks. However, a whole world of additional techniques and application domains exist beyond these, and there are many directions for work which builds upon this including the following broad areas:

- The application of a greater number of explainable AI or uncertainty quantification techniques to the problems we have presented.
- The exploration of explainability in domains which bridge network science with deep learning, such as graph neural networks (GNNs).
- The application of explainable AI to other domains beyond either deep learning or those explored in this thesis.
- Further novel visualisation developments which build upon ours.
- The application of our methodology for identifying interpretable features to new domains outside of network science.

In this chapter we will identify future work which is most closely relevant to the work we have presented, covering several of these areas. We will first discuss some direct extensions of the experiments and ideas in each of the chapters, before exploring some broader areas which may relate to the work of more than one of the earlier chapters.

7.1 Extensions to the Work on Explainable Community Finding

In Chapters 3 and 4, we performed experiments to identify which features from a longlist of interpretable social network analysis metrics were best able to explain the outputs of community finding algorithms. In both chapters, we applied a similar methodology, using permutation importance scores and Shapley values to quantify the contribution of features to the predictions on a binary classification problem related to community membership. We also performed these experiments on the same three community detection algorithms in each chapter: the Louvain algorithm, the Infomap algorithm, and the label propagation algorithm. For these reasons, we will address direct extensions to the work of these chapters together, as they share some ideas and limitations.

Additional Community Detection Algorithms In both cases, a first avenue for further experimentation would be to apply the methodology to more community detection algorithms. In our experiments on all feature-levels, i.e. node features, node-pair features and community features, we found that the most informative were consistent across the three algorithms we experimented on. It would be worth investigating whether this trend continues for other community detection approaches, or if not, to identify the differences between algorithms which lead to alternative important features. It would also be interesting to extend the work of this thesis by applying a similar methodology to overlapping community detection algorithms, as we focused only on those which partition the nodes. Future work would need to determine suitable adaptations to our methodology to allow for this. Extending even further, we note that several community detection approaches reliant on deep learning have now been proposed [66, 92, 149, 182]. Although the work of this thesis was more concerned with exploring the application of explainable AI techniques to problems outside the scope of traditional machine learning, it would also be interesting to explore how the outputs of these approaches based on deep learning could be explained.

Alternative Datasets Another area of future work mentioned in both chapters is the application of this methodology to real data. In both cases, our experiments were performed on graphs generated by the LFR algorithm, which offered us some benefits in that it allowed us to intentionally vary the value of the mixing parameter, μ, and to perform our experiments on a large dataset. However, it would be beneficial to perform some case study experiments on specific datasets of interest, to identify whether the same features are important for real data, which the

LFR graphs are designed to mimic. It would also be possible to calculate an estimation of the value of μ for this real data, so that the results could be directly compared to those from the experiments on LFR data.

Alternative Explainability Approaches Beyond the algorithms themselves, and the data, another element of the experiments which could be adjusted is the choice of explainability approach. Any metric which allows calculation of some form of importance score could be used, though we chose permutation importance and Shapley values as they are both well-known, while permutation importance scores are simple to implement and understand, and Shapley values come with desirable theoretical guarantees. One appropriate replacement could be the Shapley-Lorenz approach [52], as it is well-suited to settings such as ours where the response variable is categorical.

Visual Analytics System Aside from extending the experiments with new algorithms, data, or explainability approaches, the main avenue for future work which directly follows the experiments of Chapters 3 and 4 is the development of a visual analytics system whose design draws from the results of these experiments. One motivating application of these experiments was the ability for community structure to inform research of public health researchers studying social contagion [20, 164] and planning interventions [163] for preventing the spread of harmful behaviour on social media platforms. Initially, it would be beneficial to perform a qualitative assessment of the chosen features to assess their ability to inform public health experts who work with social network data. Beyond this, we then envisage designing a system which allows the user to highlight specific nodes, pairs of nodes, or sets of nodes in a social network and to understand the community structure of these nodes on the basis of the chosen informative features. For example, it may assist public health researchers in identifying key players in a social network who are most able to disseminate information to other actors. The incorporation of features on node and node-pair levels, as well as the community level, would allow for analysis on both a local and global scale within the network. The design of this system constitutes a significant portion of the future work which directly follows from the experiments presented in these chapters.

7.2 Extensions to the Work on Uncertainty Quantification in Social Network Problems

In Chapter 5, we then applied uncertainty quantification approaches to a regression problem faced by social media platforms dealing with large social networks. In particular, this regression task involved predicting the number of views a piece of content would receive over a 7-day window. By using uncertainty quantification approaches, we adapted the problem to predict an upper and lower bound on this number, showing that quantile regression was able to outperform our statistical baselines, and that MC Dropout could outperform both the statistical baselines and quantile regression when combined with a binary classifier into a hierarchical model, which we called HZIM.

Additional Uncertainty Quantification Approaches As with the work of the previous two chapters, an obvious area for future work would be to expand this study to include more uncertainty quantification approaches, of which there are many. For example, a simple technique similar to MC Dropout is the Deep Ensembles approach [83], thought we chose to focus on MC Dropout as it is more scalable in storage. Meanwhile, there have been several new approaches [57, 72, 143] proposed since we began this work, such as that developed by Pearce et al. [124] where the trade-off between coverage and prediction interval width is explicitly encoded in the loss function of a neural network. We performed preliminary experiments using this method by training it on our dataset, however this showed very poor performance on our zero-inflated distribution, as hyperparameter tuning did not produce suitable trade-offs between coverage and interval size. Formal experiments would be required to validate this finding. The Lower Upper Bound Estimation (LUBE) method [72] uses a similar approach, optimising prediction intervals through the construction of a specific loss function. By contrast, the Neural Optimization-based Model Uncertainty (NOMU) method [57] proposes a network architecture consisting of two connected neural networks, where one performs the usual prediction and the other predicts the uncertainty in the first.

Conformal Methods Conformal methods make use of an additional held-out calibration set to calculate a set, fixed adjustment which will be applied to all interval predictions, based on the rate of errors in the calibration set relative to the desired coverage. As well as the various uncertainty quantification approaches mentioned above, it could also be beneficial to incorporate conformal methods into those we already evaluated. Conformalisation has previously been combined with quantile regression [136], however could also benefit HZIM, which was unable to generate arbitrarily large prediction intervals due to the limitations of MC Dropout. Adjusting these predictions using conformalisation could allow us to compare HZIM with our other methods at larger interval sizes.

Asymmetrical Quantiles Another area for further experimentation could be the use of asymmetrical quantile pairs to generate prediction intervals. For all methods in our experiments, we calculated prediction intervals using a hyperparameter α such that the intervals were symmetrical, i.e. the upper and lower bounds corresponded to quantiles (α, $1-\alpha$). However, other pairs of quantiles can also be used to approximate a coverage of α. For example, we could consider setting the lower bound at 0 for all intervals, and using the uncertainty quantification approach to estimate an upper bound of α, since we expect a large proportion of content to receive zero views regardless. This approach would be equivalent to attempting to ignore the large skew in the distribution, and rank content based on the estimated upper quantile of the number of views it may receive. We also performed preliminary experiments to assess whether we could identify optimal intervals based on coverage when varying the quantile prediction for the lower bound in increments, and setting the upper bound to be a quantile α greater than this. However, these preliminary results indicated that there was little gain in performance over using symmetrical intervals. Nevertheless, a full analysis of the effect of the chosen quantiles could be considered for future work.

Selection of Content Items Finally, we have discussed in detail the application for this work: the selection of potentially-harmful content items for human review; however, we have not discussed how we can use the prediction intervals in order to select content. In a paper by Mehta et. al [104], upper quantiles are shown to perform better than mean values in top-k selection problems. In these top-k selection problems, k items are chosen from a larger set with the objective to maximise the highest value found amongst the k items. We would like to use the upper quantile predictions found by the methods evaluated in our study to identify whether these perform better at this task than mean value predictions in the setting with our unusual distribution. Additionally, it would be interesting to extend this problem to consider evaluating the methods on the sum of the top h highest values found amongst the k items, rather than the value of the single highest. The previous work by Mehta et. al indicates that upper quantiles could be more useful for this setting, however it would be worth evaluating performance using both the upper and lower quantiles as some aspects of our data distribution breaches the assumptions made in their work. These studies could indicate whether predicted quantiles would be beneficial for identifying viral content for human review.

7.3 Extensions to the Work on Extrema Graphs for Fitness Landscape Analysis

Chapter 6 introduced our novel visualisation technique for fitness landscape analysis, where we applied extrema graphs to generate a graphical representation of the problem landscape. These extrema graphs represent both minima and maxima of the search space as nodes in a graph, and connects with edges those whose Euclidean distance is less than a certain threshold, which is determined by a hyperparameter which we call ρ. These edges are then replaced with a sequence of nodes in the final visualisation, which represent evenly spaced points on the line connecting the two extrema. Our visualisation encodes fitness value using colour, and is generated using MDS to reduce the number of dimensions to two so that the graph can be visualised. We evaluated our technique by generating extrema graphs for a series of benchmark functions, and performing an interview with a fitness landscape analysis expert to gather feedback and assess how easily these benchmark visualisations could be interpreted.

Extensions to Evaluation One immediate area of future work would be to extend this evaluation by interviewing more experts, as well as to explore how extrema graphs in this context can be understood by non-experts, who may be working in domains where fitness landscape analysis is beneficial, but is not their core expertise. We expect that non-experts may not be able to interpret information from these visualisations as well as experts, and it may also be beneficial to investigate how our extrema graph visualisation can be adapted to make it easier for non-experts to understand.

Alternative Benchmark Functions Our original work could also be easily extended by generating example visualisations for more benchmark functions and incorporating into further expert interviews an exploration into how good the extrema graph visualisation is at representing various landscape features. Beyond generating more visualisations similar to the ones we have already included, future work could also include the application of extrema graphs to other types of problems, such as discrete problems or problems with multi-objective search spaces. Some examples include those introduced by Kumar et. al [82], and those introduced by Tanabe et. al. [158] To adapt our visualisation for multi-objective problems, we will need to consider how to represent solution quality, and it remains to be seen whether trade-offs between solutions that define such problems are usefully visualised in an extrema graph.

Adaptations to Visualisation Some elements of the extrema graph visualisation could also be adapted or changed, and it would be beneficial to explore through user studies which types of representation are most useful for different problem settings. For example, the expert we interviewed in our work suggested that “edge nodes” could be replaced with coloured line segments, and the multidimensional scaling (MDS) algorithm applied to extrema only in the dimensionality reduction. Therefore, points along the connecting edges of the extrema would not contribute to the distortion of the space during dimensionality reduction, but the colouring of line segments along the edges would still indicate the fitness value at these points, offering an alternative view which would allow intersecting edges to indicate how the space has been folded. Additionally, faster versions of the MDS algorithm could be used, which would also allow us to scale our approach to larger datasets, which we could include in further benchmark function exploration.

Automatic Detection of ρ Another area which could accelerate practical usage of extrema graphs is the automatic selection of an appropriate value of ρ for the given dataset. In generating our benchmark visualisations, this hyperparameter was set experimentally, by trying several values and selecting the one which generated extrema graphs with a high information content, while minimising visual clutter. Although it may be beneficial to users in many cases to be able to select the value of ρ according to the type of problem on which they are working, it is also time-consuming to test several values if unsure about which will generate a visualisation which is most valuable. Therefore, it would be highly beneficial to find a way to automate this, based on some metric of information content of the final visualisation.

Parameter Identifiability Problem Finally, one of the primary motivations behind developing the method was to investigate the fitness landscape of a maximum likelihood estimation (MLE) function. When fitting a mathematical model to real-world observations, we often query whether the parameters are identifiable [78]. In particular, our work was motivated by the use of a compartmental model for disease progression in COVID-19. Compartmental models segment population members into different buckets, such as those who are infected and those who are recovered, and model transitions between these states using differential equations [71]. Specifically, parameter identifiability relates to the identification of the convexity of the relevant MLE function, and there are common statistical procedures in biomathematics for this purpose. Current work focuses on deploying extrema graphs to the parameter identifiability problem, to complement standard parameter identifiability approaches.

7.4 Explainability for Other Social Network Analysis Problems

In the previous sections of this chapter, we have explored how the work of this thesis can be directly extended in further experiments or related applications, however there are also other areas and domains in which we can apply the techniques we have introduced. The work of this thesis has explored how explainability techniques can be applied to problems in social networks and systems, specifically community detection problems, and the prediction of content popularity on social media websites. However, approaches that we have used could also be applied to other such network problems. In particular, the work of Chapters 3 and 4 could be adapted for any network problem where we can identify a list of interpretable features which may be of interest, and represent one element of the problem we wish to understand with a binary classification task.

Identification of Key Players One such network problem could be the identification of key players in a network, for which several algorithms exist, such as the work by Borgatti et al. [15]. In their paper, they motivate the development of their KPP-Pos algorithm by demonstrating that off-the-shelf centrality metrics of individual nodes are not sufficient to perform well at identifying key players; nevertheless, it could be beneficial to apply a similar method to the one we propose in Chapters 3 and 4 to identify a set of metrics which are most closely relevant to the outcome of the KPP-Pos algorithm which can be visualised together. This approach could be even more beneficial when dealing with more complex approaches to identifying key players, such as the one proposed by Fan et al. [38] which makes use of deep reinforcement learning.

Graph Sampling Outside of identification of key players, there are also other types of algorithm we could apply our important features approach to, such as graph sampling algorithms. These algorithms aim to provide a sample of nodes and edges from a large graph while capturing its properties, so that further social network analysis can be performed on a graph of smaller scale. Some previous studies showed that certain graph sampling algorithms can be biased towards nodes of high degree [54, 90]. Applying our methodology to identify important features would potentially also have been able to identify this limitation, as degree centrality was included in our list of features and would've been highlighted as one of importance when using these algorithms. This demonstrates the benefit that explainability can bring to network problems such as these, where understanding features of importance can highlight potential biases in the sampling process. Therefore, it could be beneficial to apply a similar methodology to ours for identifying important network features to an array of graph sampling algorithms, of which there are many [5, 54, 90, 134].

Other Network Problems Beyond these, there are many other new and emerging approaches in social network analysis which could be approached with explainability to improve understanding. Some of these directly relate to community detection, but incorporate novel or more complex approaches designed to deal with large or dynamic networks, such as the work by Ma et al. [101] which combines problems in identifying key players with community detection, or the work by Panizo-LLedot et al. [123] which uses a multi-objective genetic algorithm approach to apply community detection to dynamic graphs.

The Incorporation of Uncertainty Quantification Finally, future work could also consider combining some aspects of the work of this thesis on explainability with the uncertainty quantification approaches introduced in Chapter 5. Recent work by Watson et al. [176] has shown how Shapley values can be used to explain conditional entropy in machine learning models, such as when calculated by MC Dropout, demonstrating how explainability approaches can in fact be used on the black-box models which are themselves designed to estimate uncertainty. However, this area remains relatively unexplored.

7.5 Explainability for Graph Neural Networks

In addition to the traditional social network analysis problems mentioned in the previous section, the domain of graph neural networks (GNNs) could also greatly benefit from the development of relevant explainability approaches. The work of this thesis has primarily focused on domains outside of deep learning, though our focus on network data relates closely to the problems tackled by GNN approaches. For example, existing work has used GNNs for community detection [149, 151], which we explored in Chapters 3 and 4.

Though several approaches already exist for explaining the outputs of GNNs [100, 183, 184, 186], including those which adapt and apply Shapley values for this setting [34, 126], this is still a relatively new area of work. For example, existing approaches focus primarily on instance-specific explanations, while only one approach, XGNN [184], provides model-level explanations. Several of these approaches identify important nodes or edges, for example by removing them and examining the effect on the prediction. Future work could focus on trying the reverse: adding nodes or edges, studying the effect on the prediction, and then using this to develop model-level explanations. In particular, future work could focus on the use of visualisation to aid explanation of GNN outputs by exploring how predictions vary across similar graphs. So far, little work has focused on visualisation for explainability of GNNs, aside from the work by Agarwal et al. [3], which introduces the GraphXAI library which includes some functionality to visualise important nodes or edges of a network. Nevertheless, it remains future work to explore how this can be done in a dynamic way for large networks, or to explore predictions across many networks at once.

Additionally, we could consider combining existing explainability approaches for GNNs with insight gained by using social network analysis metrics as we used in our experiments of Chapters 3 and 4, for example by visualising important nodes and edges identified by another approach, and then incorporating into a visual analytics system some exploration of the values of relevant metrics of these important elements, for example betweenness centrality.

7.6 Applications Beyond Network and Graph Problems

Finally, we can consider applications that go even beyond work which is closely related to that of this thesis, including those which deal with data outside of networks and graphs. Some of the methodologies we have introduced here may be applied in other domains, including the identification of important features in non-deep learning algorithms using a proxy binary classification problem, as in Chapters 3 and 4, or an adaptation of our extrema graph visualisation of Chapter 6 which allows us to visualise other types of data which has undergone dimensionality reduction.

For example, there are many other stochastic algorithms which do not fall under the field of deep learning and may be applied to various problems outside of social network analysis or those which apply to network data, including stochastic optimisation approaches such as simulated annealing [73] and quantum annealing [31], swarm algorithms, and evolutionary algorithms (including genetic approaches). In some of these fields explainability is a growing topic of interest [6, 9, 115], while in others it remains under-explored, and these could present opportunities to apply a similar methodology to ours for identifying interpretable features.

Separately, the zero-skewed distribution we encountered in the popularity prediction problem of Chapter 5 resembles a type of Tweedie distribution, which is normally characterised by a large proportion of data labelled zero. There are several domains where datasets distributed as a Tweedie distribution are known to be common, for example in ecology, cancer metastasis, organ blood flow, genomic structure, and actuarial studies. Thus, any prediction problem in these domains which deals with such a distribution could be another area where our uncertainty quantification results would also apply, and could be used to gather insight.

Finally, our extrema graph approach of Chapter 6 depends on the use of dimensionality reduction to enable visualisation, suggesting that visualisation techniques making use of MDS or similar algorithms could be another avenue for future work. Indeed, this could even be combined with work on uncertainty quantification, as a previous paper by Hägele et al. [60] explored how MDS could be used to visualise uncertain data. Dimensionality reduction has also commonly been used for visualisation of deep learning methods to aid with understanding. For example, in the work by Rauber et al. [132] they project the learned representations of hidden layers of neural networks for visualisation. Thus, further exploration can be done to discover how visualisation techniques combined with dimensionality reduction can aid explainability for algorithms both within and outside the domain of deep learning.

7.7 Conclusion

In this chapter, we have summarised some broad directions for future work which expands upon that proposed in this thesis, though there likely exist even more than we have detailed here. Many of the summarised ideas are immediate extensions to the experiments we have performed within in each of the respective chapters, though we have also explored some possible future work in other social network analysis problems and beyond.

Following the work of this thesis, the first angle of future work which we will explore is further adaptations of the extrema graphs visualisation introduced in Chapter 6, specifically by applying our technique to visualise loss landscapes of neural networks to gain insight into the effects of the chosen neural architecture. Having explored avenues for future work, we will now conclude this thesis with closing remarks in the final chapter, which follows this one.

Chapter 8

Conclusion

In this thesis, we have explored the application of explainable AI methodologies to new problems in domains where network data is of primary consideration, namely social network analysis and optimisation. In particular, we have focused on the community detection and fitness landscape analysis problems. We have also applied existing approaches in uncertainty quantification to a problem faced by social media companies handling large social systems, highlighting the shared motivation of explainability and uncertainty quantification to improve decision-making on the basis of outputs from black-box algorithms. We summarise these contributions in the following list, before expanding with further details:

- The development of a methodology for identifying the most informative features for community detection algorithms, which may also be adapted to other types of algorithm outside the space of traditional deep learning in the future.
- The application of this methodology to 3 algorithms, and therefore the identification of node, node-pair and community features which may be relevant for visual analysis of community structure in networks.
- A study evaluating the performance of uncertainty quantification approaches in the face of a highly-skewed distribution type.
- The development of a technique for improving the performance of MC Dropout on such a dataset.
- The development of a novel visualisation technique (the extrema graph) to capture the general characteristics of function landscapes for fitness landscape analysis.

- An analysis of the insight gained from this extrema graph approach on six benchmark problems.

In Chapter 3 we developed a methodology for identifying features that are highly predictive for explaining the results of community detection algorithms. We included only those features which can easily be understood by a social network analysis expert, and in this case, created two longlists of features defined on single nodes or pairs of nodes respectively. Our methodology requires that a binary classification problem be defined which makes use of these features. In the case of the single nodes, we distinguished those which are easy to assign to a community from those which are hard, on the basis of how much variation there is in their community membership over many runs of the community detection algorithm. In the case of pairs of nodes, we distinguished those which are in the same community from those which are in different communities. We then ranked features using a score of their importance in these classification problems; specifically, we use permutation importance and Shapley values, though other importance scores could also be used. We concluded from our experiments that there are a small number of features which are consistently important across the three community detection algorithms we tested, which included clustering coefficient, triangle participation, eigenvector centrality, expansion, cosine similarity and the Jaccard coefficient. In future work, we envisage the development of a visual analytics system to explain community membership of nodes in a given network which includes the ability to explore these metrics for given nodes in a network, and relates their values to the node's community membership.

We then extended these experiments to a third longlist of features and a third binary classification task in Chapter 4. In this case, we defined features on whole sets of nodes, and classified them as real communities or fake communities. We first calculated our features on real communities of nodes, before a rewiring process on the network adjusted the values of the features and changed the structure such that the same set of nodes no longer represented a real community. Once again, we found that the most important features were invariant across the community detection algorithms we experimented with, indicating the possibility of including them in the aforementioned visual analytics system. In this case, the primary features of interest were the cut ratio, the internal-external metric, and relative betweenness. In the visual analytics system, we envisage being able to relate values of these features for sets of nodes to their identity as a community, explaining why nodes have been clustered in a certain way.

These experiments made use of existing post-hoc explainability metrics, in this case permutation importance scores and Shapley values, in the domain of social network analysis, demonstrating the efficacy of using approaches from explainable AI in domains outside of traditional machine learning. To further address problems important in social network interactions, we next explored how uncertainty quantification approaches could be applied in this domain, though we turned our attention to a regression task making use of neural networks in this case. This work is the content of Chapter 5. In particular, we studied the "cold-start" prediction problem, where social media platforms aim to quantify how many views a piece of content will receive over a given time horizon. Though this problem is challenging, it is also of high importance for social media platforms, who aim to prioritise content efficiently for review to ensure that minimal harm is caused by inappropriate posts. To our knowledge, this is not a problem where uncertainty quantification had been previously applied.

This problem is difficult not only due to the relative lack of informative features when compared to content collected even a few minutes after its creation, but also due to the highly skewed nature of the content popularity distribution, where many items of content receive no views at all. This skewed distribution presented a novel challenge against which we evaluated the performance of existing uncertainty quantification approaches MC Dropout and Quantile Regression, finding that, although Quantile Regression improved marginally over our statistical baselines, MC Dropout was poorly adapted for this setting. To remedy this issue, we developed a hierarchical model using a binary classifier alongside MC Dropout to improve its performance. Our results indicated that this model, which we called HZIM, outperformed Quantile Regression in some regions, but was still limited by an inability to generate arbitrarily large prediction intervals. Thus, either HZIM or Quantile Regression could be appropriate methods to use for this particular problem, depending on the context.

Finally, in Chapter 6, we investigated how explainability could benefit the work of another field: fitness landscape analysis. We employed the use of visualisation as a tool for explainability, while also using a network representation, thus linking this work to our experiments in social network analysis. In particular, we developed a novel visualisation approach for representing fitness landscapes using extrema graphs, which had previously been used for isosurface extraction in volume visualisation. In the fitness landscape setting, extrema graphs as we applied them represent both minima and maxima as nodes of a network, while also incorporating points along the lines connecting the extrema as additional "edge nodes". All the included points in the fitness landscape are projected to 2D using MDS for visualisation, with "edge nodes" receiving a visually smaller representation than the extrema nodes themselves. Colour is also used for all nodes to encode fitness value. To demonstrate the information gained from visualising a fitness landscape in this way, we generated example visualisations for six benchmark functions, and observed in an expert interview that a fitness landscape analysis expert was largely able to match these representations to the surface plot of the 2D landscape. This indicates that this approach could be used in the future to help design optimisers for new types of problem, or even, as the expert suggested, to analysis the behaviour of existing optimisers.

At the beginning of this thesis, we introduced the motivating research question with which we approached this work:

> *How can we apply or adapt existing techniques from explainable AI and uncertainty quantification to new problems in network applications and social systems?*

Through the work of our contributions, we have addressed this central research question in several ways, to show that existing approaches from explainable AI and uncertainty quantification can be applied in other domains outside the realm of deep learning and in particular, those which deal with networks. The application of work from one domain to problems faced in another will continue to be an area of relevance in the coming years, especially in the face of rapid developments in the field of machine learning, a field which can both inform and learn from other computer science disciplines such as visualisation, human-computer interaction and evolutionary computation. The insight gained from all of these domains, and the further development of explainability approaches which benefit all fields, will be essential to ensure that artificial intelligence is developed to serve humanity effectively that we can thrive in the coming decades and beyond.

Bibliography

[1] A. Adadi and M. Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access*, 6:52138–52160, 2018.

[2] J. Adair, G. Ochoa, and K. M. Malan. Local Optima Networks for Continuous Fitness Landscapes. In *Proc. Genetic and Evolutionary Computation Conference (GECCO '19 Companion)*, pages 1407–1414, 2019.

[3] C. Agarwal, O. Queen, H. Lakkaraju, and M. Zitnik. Evaluating Explainability for Graph Neural Networks. *Scientific Data*, 10(144), 2023.

[4] N. Agarwal and S. Das. Interpretable Machine Learning Tools: A Survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1528–1534, 2020.

[5] Y.-Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of Topological Characteristics of Huge Online Social Networking Services. In *Proceedings of the 16th International Conference on World Wide Web*, page 835–844, 2007.

[6] M. Anneken, M. Veerappa, and N. Burkart. Anomaly Detection and XAI Concepts in Swarm Intelligence. In *NATO Symposium*, 2021.

[7] I. Arapakis, B. B. Cambazoglu, and M. Lalmas. On the feasibility of predicting news popularity at cold start. In *Proceedings of the International Conference on Social Informatics*, pages 290–299. Springer, 2014.

[8] S. Avdjiev, P. Giudici, and A. Spelta. Measuring contagion risk in international banking. *Journal of Financial Stability*, 42, 2019.

[9] J. Bacardit, A. E. I. Brownlee, S. Cagnoni, G. Iacca, J. McCall, and D. Walker. The Intersection of Evolutionary Computation and Explainable AI. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, page 1757–1762, 2022.

[10] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, 10(7):1–46, 2015.

[11] O. Bastani, C. Kim, and H. Bastani. Interpreting Blackbox Models via Model Extraction. *arXiv preprint arXiv:1705.08504*, 2019.

[12] S. Ben-David, D. Pál, and H. U. Simon. Stability of k-Means Clustering. In *Proceedings of the 20th Annual Conference on Learning Theory*, pages 20–34, 2007.

[13] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10), 2008.

[14] P. Bonacich. Power and Centrality: A Family of Measures. *American Journal of Sociology*, 92(5):1170–1182, 1986.

[15] S. Borgatti. Identifying Sets of Key Players in a Social Network. *Computational & Mathematical Organization Theory*, 12:21–34, 2006.

[16] C. Bothorel, L. Brisson, and I. Lyubareva. How to Choose Community Detection Methods in Complex Networks: The Case Study of Ulule Crowdfunding Platform. *Series Computational Social Science*, 2020.

[17] U. Brandes. A Faster Algorithm for Betweenness Centrality. *Journal of Mathematical Sociology*, 25(2), 2001.

[18] U. Brandes and C. Pich. Eigensolver Methods for Progressive Multidimensional Scaling of Large Data. In *Graph Drawing*, pages 42–53, 2007.

[19] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[20] R. C. Brown, T. Fischer, A. D. Goldwich, F. Keller, R. Young, and P. L. Plener. #cutting: Non-suicidal self-injury (NSSI) on Instagram. *Psychological Medicine*, 48(2):337–346, 2017.

[21] K. Canales. Mark Zuckerberg said content moderation requires 'nuances' that consider the intent behind a post, but also highlighted Facebook's reliance on AI to do that job, 2021. [Online] [Accessed 19-July-2024] Avail-

able from: https://www.businessinsider.com/zuckerberg-nuances-content-moderation-ai-misinformation-hearing-2021-3.

[22] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso. Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics*, 8(8):832, 2019.

[23] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly. Metrics for Community Analysis: A Survey. 50(4), 2017.

[24] T. Chakraborty, S. Srinivasan, N. Ganguly, S. Bhowmick, and A. Mukherjee. Constant Communities in Complex Networks. *Scientific reports*, 3:1825, 2013.

[25] M. C. A. Clare, M. Sonnewald, R. Lguensat, J. Deshayes, and V. Balaji. Explainable Artificial Intelligence for Bayesian Neural Networks: Toward Trustworthy Predictions of Ocean Dynamics. *Journal of Advances in Modeling Earth Systems*, 14(11), 2022.

[26] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111, 2004.

[27] M. A. A. Cox and T. F. Cox. *Multidimensional Scaling*, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[28] L. Danon, A. Dí az-Guilera, and A. Arenas. The effect of size heterogeneity on community identification in complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(11):P11010–P11010, 2006.

[29] V. L. Dao, C. Bothorel, and P. Lenca. Community structures evaluation in complex networks: A descriptive approach. In *Proceedings of the International Conference and School on Network Science (NetSci-X 2017)*, pages 11–19, 2017.

[30] V. L. Dao, C. Bothorel, and P. Lenca. Community structure: A comparative evaluation of community detection methods. *Network Science*, 8(1):1–41, 2020.

[31] D. de Falco, B. Apolloni, and N. Cesa-Bianchi. A numerical implementation of quantum annealing. In *Stochastic Processes, Physics and Geometry: Proceedings of the Ascona-Locarno Conference*, pages 97–111, 1990.

[32] V. De Silva and J. B. Tenenbaum. Sparse multidimensional scaling using landmark points. Technical report, technical report, Stanford University, 2004.

[33] H. Du, M. W. Feldman, S. Li, and X. Jin. An Algorithm for Detecting Community Structure of Social Networks Based on Prior Knowledge and Modularity: Research Articles. *Complex.*, 12(3):53–60, 2007.

[34] A. Duval and F. D. Malliaros. GraphSVX: Shapley Value Explanations for Graph Neural Networks. In *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II*, page 302–318, 2021.

[35] R. Dwivedi, D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, and R. Ranjan. Explainable AI (XAI): Core Ideas, Techniques, and Solutions. *ACM Comput. Surv.*, 55(9):1–33, 2023.

[36] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *Bernoulli*, (1341):1–13, 2009.

[37] M. Espadoto, N. S. T. Hirata, and A. C. Telea. Deep learning multidimensional projections. *Information Visualization*, 19(3):247–269, 2020.

[38] C. Fan, Z. li, Y. Sun, and Y.-Y. Liu. Finding key players in complex networks through deep reinforcement learning. *Nature Machine Intelligence*, 2:1–8, 2020.

[39] J. E. Fieldsend. Running Up Those Hills: Multi-modal search with the niching migratory multi-swarm optimiser. In *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 2593–2600, 2014.

[40] J. E. Fieldsend and K. Alyahya. Visualising the Landscape of Multi-Objective Problems using Local Optima Networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '19 Companion)*, pages 1421–1429, 2019.

[41] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient Identification of Web Communities. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00)*, pages 150–160, 2000.

[42] R. Fong and A. Vedaldi. Interpretable Explanations of Black Boxes by Meaningful Perturbation. *CoRR*, abs/1704.03296, 2017.

[43] S. Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[44] A. P. Francisco and A. L. Oliveira. On Community Detection in Very Large Networks. In L. da F. Costa, A. Evsukoff, G. Mangioni, and R. Menezes, editors, *Complex Networks*, pages 208–216, 2011.

[45] L. C. Freeman. Centrality in networks: I. Conceptual clarification. *Social Networks*, 1:215–239, 1979.

[46] N. Frosst and G. Hinton. Distilling a Neural Network Into a Soft Decision Tree. *arXiv preprint arXiv:1711.09784*, 2017.

[47] Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, page 1050–1059, 2016.

[48] E. R. Gansner, Y. Koren, and S. North. Graph Drawing by Stress Majorization. In J. Pach, editor, *Graph Drawing*, pages 239–250, 2005.

[49] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, et al. A Survey of Uncertainty in Deep Neural Networks. *Artificial Intelligence Review*, 56(1):1513–1589, 2023.

[50] A. Ghasemian, H. Hosseinmardi, and A. Clauset. Evaluating Overfit and Underfit in Models of Network Community Structure. *IEEE Transactions on Knowledge and Data Engineering*, 32(9):1722–1735, 2019.

[51] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–7826, 2002.

[52] P. Giudici and E. Raffinetti. Shapley-Lorenz eXplainable Artificial Intelligence. *Expert Systems with Applications*, 167, 2021.

[53] P. Giudici, P. Sarlin, and A. Spelta. The interconnected nature of financial systems: Direct and common exposures. *Journal of Banking & Finance*, 112:105149, 2017.

[54] M. Gjoka, M. Kurant, C. T. Butts, and A. Markopoulou. Walking in Facebook: A Case Study of Unbiased Sampling of OSNs. In *Proceedings of IEEE INFOCOM 2010*, pages 1–9, 2010.

[55] A. Hagberg, P. Swart, and D. S Chult. Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Lab., NM, 2008.

[56] S. Harenberg, G. Bello, L. Gjeltema, S. Ranshous, J. Harlalka, R. Seay, K. Padmanabhan, and N. Samatova. Community detection in large-scale networks: a survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6):426–439, 2014.

[57] J. Heiss, J. Weissteiner, H. Wutte, S. Seuken, and J. Teichmann. NOMU: Neural Optimization-based Model Uncertainty. In *International Conference on Machine Learning*, pages 8708–8758, 2022.

[58] F. Hohman, M. Kahng, R. Pienta, and P. Chau. Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2674–2693, 2018.

[59] Q. Huang, M. Yamada, Y. Tian, D. Singh, and Y. Chang. GraphLIME: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(7):6968–6972, 2022.

[60] D. Hägele, T. Krake, and D. Weiskopf. Uncertainty-Aware Multidimensional Scaling. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):23–32, 2023.

[61] E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.

[62] T. Itoh and K. Koyamada. Isosurface generation by using extrema graphs. In *Proceedings Visualization ’94*, pages 77–83, 1994.

[63] T. Itoh and K. Koyamada. Automatic isosurface propagation using an extrema graph and sorted boundary cell lists. *IEEE Transactions on Visualization and Computer Graphics*, 1(4):319–327, 1995.

[64] P. Jaccard. The distribution of flora in the alpine zone. *New Phytologist*, 11(2):37–50, 1912.

[65] E. Jayne Bienenstock and P. Bonacich. Eigenvector centralization as a measure of structural bias in information aggregation. *The Journal of Mathematical Sociology*, pages 1–19, 2021.

[66] Y. Jia, Q. Zhang, W. Zhang, and X. Wang. CommunityGAN: Community Detection with Generative Adversarial Nets. In *Proceedings of The World Wide Web Conference 2019*, page 784–794, 2019.

[67] P. Johnsen, I. Strumke, M. Langaas, A. DeWan, and S. Riemer-Sørensen. Inferring feature importance with uncertainties with application to large genotype data. *PLoS computational biology*, 19:e1010963, 2023.

[68] M. Kahng, N. Thorat, D. Horng Chau, F. B. Ví egas, and M. Wattenberg. GAN Lab: Understanding Complex Deep Generative Models using Interactive Visual Experimentation. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):310–320, 2018.

[69] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989.

[70] M. T. Keane and E. M. Kenny. How case-based reasoning explains neural networks: A theoretical analysis of XAI using post-hoc explanation-by-example from a survey of ANN-CBR twin-systems. In *Proceedings of the International Conference on Case-Based Reasoning (ICCBR'19)*, pages 155–171. Springer, 2019.

[71] M. Keeling, P. Rohani, and B. Pourbohloul. Modeling Infectious Diseases in Humans and Animals. *Clinical infectious diseases : an official publication of the Infectious Diseases Society of America*, 47:864–865, 2008.

[72] A. Khosravi, S. Nahavandi, D. Creighton, and A. F. Atiya. Lower Upper Bound Estimation Method for Construction of Neural Network-Based Prediction Intervals. *IEEE Transactions on Neural Networks*, 22(3):337–346, 2011.

[73] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by Simulated Annealing. *Science (New York, N.Y.)*, 220:671–680, 1983.

[74] M. Klimenta and U. Brandes. Graph Drawing by Classical Multidimensional Scaling: New Perspectives. In *Graph Drawing*, pages 55–66, 2013.

[75] R. Koenker. *Quantile Regression*. Econometric Society Monographs. Cambridge University Press, 2005.

[76] R. Koenker and G. Bassett. Regression Quantiles. *Econometrica*, 46(1):33–50, 1978.

[77] R. Koenker and K. F. Hallock. Quantile regression. *Journal of economic perspectives*, 15(4):143–156, 2001.

[78] C. Kreutz, A. Raue, D. Kaschek, and J. Timmer. Profile likelihood in systems biology. *The FEBS journal*, 280(11):2564–2571, 2013.

[79] S. Krishnan and E. Wu. PALM: Machine Learning Explanations For Iterative Debugging. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*, 2017.

[80] J. F. Kruiger, P. E. Rauber, R. M. Martins, A. Kerren, S. Kobourov, and A. C. Telea. Graph Layouts by t-SNE. *Computer Graphics Forum*, 36(3):283–294, 2017.

[81] J. Kruskal. Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29:115–129, 1964.

[82] A. Kumar, G. Wu, M. Z. Ali, Q. Luo, R. Mallipeddi, P. N. Suganthan, and S. Das. A Benchmark-Suite of real-World constrained multi-objective optimization problems and some baseline results. *Swarm and Evolutionary Computation*, 67:100961, 2021.

[83] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 6405–6416, 2017.

[84] A. Lancichinetti and S. Fortunato. Community detection algorithms: A comparative analysis. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 80(5):1–12, 2009.

[85] A. Lancichinetti and S. Fortunato. Consensus clustering in complex networks. *Scientific reports*, 2(1):1–7, 2012.

[86] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 78(4):1–6, 2008.

[87] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato. Finding Statistically Significant Communities in Networks. *PLOS ONE*, 6(4):1–18, 2011.

[88] A. Lee and D. Archambault. Communities Found by Users – not Algorithms. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 2396–2400, 2016.

[89] C. Lee, F. Reid, A. McDaid, and N. Hurley. Detecting highly overlapping community structure by greedy clique expansion. In *Proceedings of the 4th International Workshop on Social Network Mining and Analysis (SNA-KDD)*, page 33–42, 2010.

[90] S. H. Lee, P.-J. Kim, and H. Jeong. Statistical properties of sampled networks. *Physical Review E*, 73(1):016102, 2006.

[91] Z. C. Lipton. The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[92] F. Liu, S. Xue, J. Wu, C. Zhou, W. Hu, C. Paris, S. Nepal, J. Yang, and P. S. Yu. Deep Learning for Community Detection: Progress, Challenges and Opportunities. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2020.

[93] M. Liu, J. Shi, K. Cao, J. Zhu, and S. Liu. Analyzing the Training Processes of Deep Generative Models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):77–87, 2018.

[94] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu. Towards Better Analysis of Deep Convolutional Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):91–100, 2017.

[95] S. Liu, X. Wang, M. Liu, and J. Zhu. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1):48–56, 2017.

[96] O. Loyola-Gonzalez, A. E. Gutierrez-Rodríguez, M. A. Medina-Pérez, R. Monroy, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and M. Garcia-Borroto. An Explainable Artificial Intelligence Model for Clustering Numerical Databases. *IEEE Access*, 8:52370–52384, 2020.

[97] Y. Lu, R. Garcia, B. Hansen, M. Gleicher, and R. Maciejewski. The State-of-the-Art in Predictive Visual Analytics. *Computer Graphics Forum*, 36(3):539–562, 2017.

[98] S. Lundberg, G. Erion, H. Chen, A. DeGrave, J. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee. From Local Explanations to Global Understanding with Explainable AI for Trees. *Nature Machine Intelligence*, 2(1):56–67, 2020.

[99] S. M. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 4768–4777, 2017.

[100] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

[101] T. Ma, Q. Liu, J. Cao, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan. LGIEM: Global and local node influence based community detection. *Future Generation Computer Systems*, 105:533–546, 2020.

[102] T. Martin, J. M. Hofman, A. Sharma, A. Anderson, and D. J. Watts. Exploring limits to prediction in complex social systems. In *Proceedings of the 25th International Conference on World Wide Web*, pages 683–694, 2016.

[103] C. Marx, Y. Park, H. Hasson, Y. Wang, S. Ermon, and L. Huan. But Are You Sure? An Uncertainty-Aware Perspective on Explainable AI. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206, pages 7375–7391, 2023.

[104] A. Mehta, U. Nadav, A. Psomas, and A. Rubinstein. Hitting the High Notes: Subset Selection for Maximizing Expected Order Statistics. In *Proceedings of the Annual Conference on Neural Information Processing Systems 2020*, 2020.

[105] K. Michalak. Low-Dimensional Euclidean Embedding for Visualization of Search Spaces in Combinatorial Optimization. *IEEE Transactions on Evolutionary Computation*, 23(2):232–246, 2019.

[106] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu. Understanding Hidden Memories of Recurrent Neural Networks. *2017 IEEE Conference on Visual Analytics Science and Technology, VAST 2017 - Proceedings*, pages 13–24, 2018.

[107] Y. Ming, H. Qu, and E. Bertini. RuleMatrix: Visualizing and Understanding Classifiers with Rules. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):342–352, 2019.

[108] A. Mobiny, P. Yuan, S. K. Moulik, N. Garg, C. C. Wu, and H. Van Nguyen. Dropconnect is effective in modeling uncertainty of Bayesian deep networks. *Scientific reports*, 11(1):1–14, 2021.

[109] S. Mohseni, N. Zarei, and E. D. Ragan. A Multidisciplinary Survey and Framework for Design and Evaluation of Explainable AI Systems. 11(3-4):1–45, 2018.

[110] C. Molnar, G. Casalicchio, and B. Bischl. Interpretable Machine Learning – A Brief History, State-of-the-Art and Challenges. (01):1–15, 2020.

[111] N. Moniz and L. Torgo. A review on web content popularity prediction: Issues and open challenges. *Online Social Networks and Media*, 12:1–20, 2019.

[112] A. Morichetta, P. Casas, and M. Mellia. EXPLAIN-IT: towards explainable AI for unsupervised network traffic analysis. In *Proceedings of the 3rd ACM CoNEXT Workshop on Big Data, Machine Learning and Artificial Intelligence for Data Communication Networks*, pages 22–28, 2019.

[113] C. Munoz, K. da Costa, B. Modenesi, and A. Koshiyama. Evaluating explainability for machine learning predictions using model-agnostic metrics. *arXiv preprint arXiv:2302.12094*, 2023.

[114] T. Munzner. *Visualization Analysis and Design*. AK Peters Visualization Series. CRC Press, 2014.

[115] M. Naiseh, M. Soorati, and S. Ramchurn. *Outlining the Design Space of eXplainable Swarm (xSwarm): Experts' Perspective*, pages 28–41. 2024.

[116] M. Newman and A. Clauset. Structure and inference in annotated networks. *Nature Communications*, 7(1):11863, 2016.

[117] G. Ochoa, K. M. Malan, and C. Blum. Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics. *Applied Soft Computing*, 109:107492, 2021.

[118] G. Ochoa, M. Tomassini, S. Vérel, and C. Darabos. A Study of NK Landscapes' Basins and Local Optima Networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008)*, pages 555–562, 2008.

[119] G. Ochoa and N. Veerapen. Additional dimensions to the study of funnels in combinatorial landscapes. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, pages 373–380, 2016.

[120] G. Orman, V. Labatut, and H. Cherifi. Comparative evaluation of community detection algorithms: A topological approach. *Journal of Statistical Mechanics: Theory and Experiment, P08001*, 2012.

[121] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford InfoLab, 1999.

[122] G. Pallaand, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.

[123] A. Panizo-LLedot, G. Bello-Orgaz, and D. Camacho. A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme. *Future Generation Computer Systems*, 110:960–975, 2020.

[124] T. Pearce, A. Brintrup, M. Zaki, and A. Neely. High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 4075–4084, 2018.

[125] L. Peel, D. B. Larremore, and A. Clauset. The ground truth about metadata and community detection in networks. *Science Advances*, 3(5):F1602548, 2017.

[126] A. Perotti, P. Bajardi, F. Bonchi, and A. Panisson. GRAPHSHAP: Explaining Identity-Aware Graph Classifiers Through the Language of Motifs. *arXiv preprint arXiv:2202.08815*, 2022.

[127] N. Puri, P. Gupta, P. Agarwal, S. Verma, and B. Krishnamurthy. Magix: Model agnostic globally interpretable explanations. *arXiv preprint arXiv:1706.07160*, 2017.

[128] Z. Qin, F. Yu, C. Liu, and X. Chen. How convolutional neural networks see the world — A survey of convolutional neural network visualization methods. *Mathematical Foundations of Computing*, 1(2):149–180, 2018.

[129] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *PNAS*, 101(9), 2004.

[130] U. N. Raghavan and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks 1 1. pages 1–12.

[131] M. J. Rattigan, M. Maier, and D. Jensen. Graph Clustering with Network Structure Indices. In *Proceedings of the 24th International Conference on Machine Learning*, page 783–790, 2007.

[132] P. E. Rauber, S. G. Fadel, A. X. Falcão, and A. C. Telea. Visualizing the Hidden Activity of Artificial Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):101–110, 2017.

[133] D. Ren, S. Amershi, B. Lee, J. Suh, and J. D. Williams. Squares: Supporting Interactive Performance Analysis for Multiclass Classifiers. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):61–70, 2017.

[134] B. Ribeiro and D. Towsley. Estimating and sampling graphs with multidimensional random walks. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 390–403, 2010.

[135] M. T. Ribeiro, S. Singh, and C. Guestrin. "Why should I trust you?" Explaining the predictions of any classifier. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 13-17-Augu:1135–1144, 2016.

[136] Y. Romano, E. Patterson, and E. Candes. Conformalized quantile regression. *Advances in Neural Information Processing Systems*, 32:3543–3553, 2019.

[137] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences of the United States of America*, 105(4):1118–1123, 2008.

[138] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[139] S. Sadler, D. Greene, and D. Archambault. Selecting Informative Features for Post-Hoc Community Explanation. In *Proceedings of the 10th International Conference on Complex Networks and Their Applications*, pages 297–308, 2021.

[140] S. Sadler, D. Greene, and D. Archambault. Towards Explainable Community Finding. *Applied Network Science*, 7(1):81, 2022.

[141] S. Sadler, A. Rahat, D. Walker, and D. Archambault. Extrema Graphs: Fitness Landscape Analysis to the Extreme! In *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, page 2081–2089, 2023.

[142] W. Saeed and C. Omlin. Explainable AI (XAI): A systematic meta-survey of current challenges and future opportunities. *Knowledge-Based Systems*, 263:110273, 2023.

[143] T. S. Salem, H. Langseth, and H. Ramampiaro. Prediction intervals: Split normal mixture from quality-driven deep ensembles. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 1179–1187, 2020.

[144] M. Schroepfer. Community Standards report, 2019. [Online] [Accessed 19-July-2024] Available from: https://ai.meta.com/blog/community-standards-report/.

[145] D. Seuß. Bridging the gap between explainable AI and uncertainty quantification to enhance trustability. *arXiv preprint arXiv:2105.11828*, 2021.

[146] G. Shafer and V. Vovk. A Tutorial on Conformal Prediction. *Journal of Machine Learning Research*, 9(3), 2008.

[147] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[148] L. S. Shapley. A Value for n-Person Games. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, Princeton, 1953.

[149] O. Shchur and S. Günnemann. Overlapping community detection with graph neural networks. *arXiv preprint arXiv:1909.12201*, 2019.

[150] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[151] S. Sobolevsky. Recurrent graph neural network algorithm for unsupervised network community detection. *arXiv preprint arXiv:2103.02520*, 2021.

[152] T. Spinner, U. Schlegel, H. Schäfer, and M. El-Assady. explAIner: A Visual Analytics Framework for Interactive and Explainable Machine Learning. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1064–1074, 2019.

[153] I. Steinwart and A. Christmann. Estimating conditional quantiles with the help of the pinball loss. *Bernoulli*, 17(1), 2011.

[154] A. Strehl. Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining. Master's thesis, The University of Texas at Austin, 2002.

[155] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. SEQ2SEQ-VIS : A Visual Debugging Tool for Sequence-to-Sequence Models. *IEEE Transactions on Visualization and Computer Graphics*, 2018.

[156] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush. LSTMVis: A Tool for Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, 2016.

[157] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 3319–3328, 2017.

[158] R. Tanabe and H. Ishibuchi. An easy-to-use real-world multi-objective optimization problem suite. *Applied Soft Computing*, 89:106078, 2020.

[159] E. Tjoa and C. Guan. A Survey on Explainable Artificial Intelligence (XAI): Toward Medical XAI . *IEEE Transactions on Neural Networks and Learning Systems*, 32(11):4793–4813, 2021.

[160] M. Tory and T. Moller. Evaluating visualizations: do expert reviews work? *IEEE Computer Graphics and Applications*, 25(5):8–11, 2005.

[161] V. A. Traag, L. Waltman, and N. J. van Eck. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1), 2019.

[162] J. R. Tyler, D. M. Wilkinson, and B. A. Huberman. Email as Spectroscopy: Automated Discovery of Community Structure within Organizations, 2003.

[163] T. W. Valente. Network Interventions. *Science*, 337(6090):49–53, 2012.

[164] T. W. Valente and G. G. V. Yon. Diffusion/Contagion Processes on Social Networks. *Health Education & Behavior*, 47(2):235–248, 2020.

[165] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[166] M. van Lent, W. Fisher, and M. Mancuso. An Explainable Artificial Intelligence System for Small-Unit Tactical Behavior. In *Proceedings of the 16th Conference on Innovative Applications of Artifical Intelligence*, page 900–907, 2004.

[167] N. Veerapen and G. Ochoa. Visualising the global structure of search landscapes: genetic improvement as a case study. *Genetic Programming and Evolvable Machines*, 19:1–33, 2018.

[168] J. Vincent. Facebook is now using AI to sort content for quicker moderation, 2020. [Online] [Accessed 19-July-2024] Available from: https://www.theverge.com/2020/11/13/21562596/facebook-ai-moderation.

[169] T. von Landesberger, M. Gorner, and T. Schreck. Visual analysis of graphs with multiple connected components. In *2009 IEEE Symposium on Visual Analytics Science and Technology*, pages 155–162, 2009.

[170] U. Von Luxburg et al. Clustering stability: an overview. *Foundations and Trends® in Machine Learning*, 2(3):235–274, 2010.

[171] M. Vu and M. T. Thai. PGM-explainer: Probabilistic graphical model explanations for graph neural networks. *Advances in neural information processing systems*, 33:12225–12235, 2020.

[172] K. Wakita and T. Tsurumi. Finding community structure in mega-scale social networks. In *Proceedings of the 16th international conference on World Wide Web*, pages 1275–1276, 2007.

[173] D. J. Walker, R. M. Everson, and J. E. Fieldsend. Visualizing Mutually Nondominating Solution Sets in Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 17(2):165–184, 2013.

[174] M. J. Walter, D. J. Walker, and M. J. Craven. Visualising evolution history in multi-and many-objective optimisation. In *Proceedings of Parallel Problem Solving from Nature (PPSN)*, pages 299–312, 2020.

[175] J. Wang, L. Gou, H. Yang, and H. W. Shen. GANViz: A Visual Analytics Approach to Understand the Adversarial Game. *IEEE Transactions on Visualization and Computer Graphics*, 24(6):1905–1917, 2018.

[176] D. Watson, J. O'Hara, N. Tax, R. Mudd, and I. Guy. Explaining predictive uncertainty with information theoretic shapley values. *Advances in Neural Information Processing Systems*, 36, 2024.

[177] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[178] F. Wenzel, J. Snoek, D. Tran, and R. Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *Advances in Neural Information Processing Systems*, 33:6514–6527, 2020.

[179] J. Xie, Y. Zhu, Z. Zhang, J. Peng, J. Yi, Y. Hu, H. Liu, and Z. Chen. A multimodal variational encoder-decoder framework for micro-video popularity prediction. In *Proceedings of The Web Conference 2020*, pages 2542–2548, 2020.

[180] X. Xu, F. Zhou, K. Zhang, S. Liu, and G. Trajcevski. CasFlow: Exploring Hierarchical Structures and Propagation Uncertainty for Cascade Prediction. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3484–3499, 2021.

[181] J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42:181–213, 2015.

[182] L. Yang, X. Cao, D. He, C. Wang, X. Wang, and W. Zhang. Modularity based community detection with deep learning. In *IJCAI*, volume 16, pages 2252–2258, 2016.

[183] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. GNNexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

[184] H. Yuan, J. Tang, X. Hu, and S. Ji. XGNN: Towards Model-Level Explanations of Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 430–438, 2020.

[185] H. Yuan, H. Yu, S. Gui, and S. Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):5782–5799, 2022.

[186] H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji. On explainability of graph neural networks via subgraph explorations. In *Proceedings of the International conference on machine learning*, pages 12241–12252, 2021.

[187] W. Zachary. An Information Flow Model for Conflict and Fission in Small Groups. *Journal of anthropological research*, 33, 1976.

[188] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science*, 8689 LNCS(PART 1):818–833, 2014.

[189] X. Zhang, F. T. Chan, and S. Mahadevan. Explainable Machine Learning in Image Classification Models: An Uncertainty Quantification Perspective. *Knowledge-Based Systems*, 243(C):108418, 2022.

[190] J. X. Zheng, S. Pawar, and D. M. Goodman. Graph Drawing by Stochastic Gradient Descent. *IEEE Transactions on Visualization and Computer Graphics*, 25(9):2738–2748, 2019.

[191] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pages 321–328, 2004.

[192] F. Zhou, X. Xu, K. Zhang, G. Trajcevski, and T. Zhong. Variational information diffusion for probabilistic cascades prediction. In *Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1618–1627. IEEE, 2020.