Gragnostics: Fast, Interpretable Features for Comparing Graphs

Robert Gove

Two Six Labs, robert.gove@twosixlabs.com

Abstract—Many analytical tasks, such as social network analysis, depend on comparing graphs. Existing methods are slow, or can be difficult to understand. To address these challenges, this paper proposes gragnostics, a set of 10 fast, layperson-understandable graph-level features. Each can be computed in linear time. To evaluate the ability of these features to discriminate different topologies and types of graphs, this paper compares a machine learning classifier using gragnostics to alternative classifiers, and the evaluation finds that the gragnostics classifier achieves higher performance. To evaluate gragnostics’ utility in interactive visualization tools, this paper presents Chiron, a graph visualization tool that enables users to explore the subgraphs of a larger graph. Example usage scenarios of Chiron demonstrate that using gragnostics in a rank-by-feature framework can be effective for finding interesting subgraphs.

Index Terms—Graph, cognostics, visualization

I. INTRODUCTION

Comparing graphs is a common task—e.g. finding successful community characteristics [1], or predicting graph layout quality [2]. Algorithms for this task are often slow, uninterpretable, or have restrictions, e.g. that graphs be the same size or only have one component. Visualization tools for this task have typically been designed to compare individual vertices or edges [3], [4]; used slower and/or less interpretable algorithms to compare graphs [5]–[7]; or may not be expressive enough for tasks like classification or clustering [5].

Human-interpretable *features*, also known as statistics or metrics, are important for a variety of reasons. Interpretability is a central topic in modern machine learning research, and it is important for such goals as fairness, failure diagnosis, etc. Human interpretability is also important for the usability of software systems that depend on these features; by using terminology already familiar to users, software system designers can reduce cognitive load, speed up learnability, and improve re-learnability. Interpretability in machine learning has many components, but one element is that humans should be able to understand the individual features used [8].

Fast runtime is important because of both the increasingly large size and number of graphs that analysts and systems need to compare. Although some existing tools and feature sets rely on linear-time features such as the number of vertices or edges, many include slower features like graph diameter [7]. Furthermore, some graph kernels require that graphs satisfy certain properties, such as being composed of a single component, and some feature sets require that all graphs be the same size.

To address these limitations, this paper proposes using a set of 10 graph-level features, which are referred to as *gragnostics*,

a portmanteau of “Graph” and “Diagnostics.” Gragnostics is a mix of two new features, five existing ones, and three other existing features that are normalized and renamed with the goal of making them more understandable by laypeople. The features can be computed in $(|V| + |E|)$ time, and they are independent of layout or visual representation, thereby supporting a wide range of applications. This paper shows that a simple machine learning classifier using gragnostics as features can achieve good classification performance.

The contributions of this paper are (1) two new graph-level features for measuring how tree-like and how line-like a graph is; (2) an evaluation of gragnostics, a set of 10 graph-level features for machine learning classifiers and rank-by-feature visualization tools; (3) a rank-by-feature approach for visual analytics on subgraphs of a larger graph; and (4) a replication of other visualization tool design research that confirms their findings and the utility of subgraph clustering.

II. BACKGROUND

Graphs: A graph $G = (V, E)$ is a set of vertices V and a set of edges E , where if $e \in E$ then $e = (v, u)$ such that $v, u \in V$. In *undirected* graphs, (v, u) is the same as (u, v) . *Simple* graphs have no duplicate edges or self-loops (e.g. (v, v)). $|V|$ and $|E|$ denote the number of vertices and edges, respectively. A graph $S = (V', E')$ is a *subgraph* of G if $V' \subseteq V$, $E' \subseteq E$, and if $(u, v) \in E'$ then $u, v \in V'$. The number of edges containing v is the *degree* and denoted $d(v)$.

A *path* in G is a distinct sequence of vertices from V such that every consecutive pair of vertices corresponds to an edge in E . A *component* of G is subset of vertices such that there is a path between every pair of vertices, but no other vertex can be added to the component and have a path connecting it to any other vertex in the component. A graph is *connected* if it has exactly one component, and a graph is *disconnected* if it has more than one component.

A *cycle* is a path where the first and last vertices are the same. A *tree* is a graph without cycles, and a *forest* is a graph with multiple components where each one is a tree. The *path graph*, denoted P_n , is a tree with n vertices such that two vertices have degree 1 and the other $n - 2$ vertices have degree 2. The *star graph*, denoted S_n , is a tree with n vertices such that one vertex has degree $n - 1$ and all other vertices have degree 1. The *complete graph*, denoted K_n , is a graph with n vertices where each vertex has an edge to the other $n - 1$ vertices. A *bridge* is an edge such that removing it would increase the number of components in the graph. Similarly, a

cut vertex is a vertex such that removing it and all its edges would increase the number of components in the graph.

Visual Quality Metrics: Seo and Shneiderman [9] proposed the rank-by-feature framework for ranking visualizations by statistical features of the underlying data, such as normality, but it uses feature names that may be unfamiliar to novices, such as least squares error. Wilkinson et al [10] introduced a novel approach to calculating scagnostics (Scatterplot Diagnostics). Scagnostics measure a scatterplot along nine interpretable characteristics, avoiding niche statistics terminology. These are useful for sorting, filtering, or clustering scatterplots to identify visualizations that may reveal interesting patterns. In contrast, Magnostics operates in the image of the graph visualization and not the underlying data itself. Magnostics uses image descriptors of matrix visualizations to rank visualizations by visual patterns, cluster or search for visually similar matrices, or compare the quality of matrix ordering algorithms. Magnostics can only be used with matrix visualizations, and could be sensitive to the row and column orderings of the matrix. In contrast, gragnostics is independent of the visual representation of the graph, and gragnostics strives to use feature names that are familiar to users.

Comparing Graphs: Graph-level statistics can capture the relationship between graph-level structure and information flow in graphs. Himelboim et al. [11] discuss research that indicates high interconnectivity corresponds to shared knowledge, better information transmission, and the rate of information spread; clusters and modularity correspond to shared characteristics between vertices within a cluster; centralization corresponds to information flow dominated by a few individuals and vulnerable to disruption; and isolation corresponds to slow information flow and weak relationships. This supports (1) the idea that characterizing a graph’s structure can lead to insights about the graph, and (2) the features should be understandable by humans so that analysts can understand, e.g., that two graphs are similar but differ in their clustering.

Vertex- and edge-level statistics (see Newman [12] for an overview) have proven quite useful for comparing vertices and edges [3]. Graph-level statistics—e.g. clustering coefficient and graph diameter—are also useful, but several of them are not readily understandable by laypeople without a background in graph theory. A second problem is that many graph-level statistics run in $O(|V|^2)$ time or slower, making them impractical for large graph datasets.

Several visualization tools use Euclidean distance on adjacency matrices to compute distances and cluster graphs [13], [14]. Although the systems appear useful, the feature extraction and distance computation is $O(|V|^2)$, and the features do not lend themselves to concise explanations of why two graphs are similar. This technique also requires graphs to have the same number of vertices, which reduces its generality.

Motif detection [6] and graph kernels [2] are related techniques for measuring similarities between graphs. However, the naive method is slow [6]. Random sampling techniques to compute motif (aka graphlet) frequencies can improve performance [15], but the features do not lend themselves to

easy human interpretability. Furthermore, many graph kernels are not designed for graphs with multiple components [2]. Nonetheless, graph kernels have been successful in cases where explanation is not required [2].

Degree Distribution Quantification and Comparison (DDQC) [16] extracts features in linear time, but it is not designed for human interpretability. Also, classification accuracy on real datasets has not been shown to be very high.

All of the above face at least one challenge with speed, interpretability, or requirements that graphs have only one component or be the same size. In contrast, gragnostics runs in $O(|V| + |E|)$, does not require familiarity with obscure graph theory terminology, and do not impose restrictions on graphs such as size or number of components.

Graph Visualization: When comparing graphs, visualization tools need analytics and interactions that support the sensemaking process, and not just fast performance.

ManyNets [5] enables users to compare multiple graphs using graph-level statistics in a sortable tabular display. Graphs can be selected to view in a node-link diagram. EgoNav [6] computes distances between ego networks using motif frequencies. Similarly, von Landesberger et al. [7] use a mix of graph-, vertex-, and edge-level statistics and motif frequencies to cluster multiple components of a graph. In contrast to these three tools, (1) Chiron uses the rank-by-feature framework to aid exploration of subgraphs within a larger graph (subgraphs are believed to be units of interest in large graphs [17]); and (2), Chiron uses a focus+context design that allows users to see the details of a subgraph while also seeing an overview of the entire large graph, which is believed to be useful even if low-level details are not visible [18].

None of the above visualization tools used only linear-time graph features, so it is unknown whether linear-time features can yield the same level of insight. Furthermore, it is not known whether combining subgraph exploration with an overview of the entire graph provides benefits. As shown later in this paper, linear-time features can be effective, and an overview of the entire graph can provide useful context.

III. TOPOLOGICAL GRAGNOSTICS

Gragnostics were designed to balance several constraints. First, gragnostics must scale to large graphs. To accomplish this, gragnostics was designed to be computed in $O(|V| + |E|)$ time. Second, the features must be comprehensible to analysts who are not experts in graph theory. To accomplish this, gragnostics correspond to topological characteristics described in plain language. This enables broad audiences to easily understand gragnostics. Third, gragnostics should not be constrained to certain types of graphs. To ensure this, gragnostics do not have restrictions on graph size or number of components.

Of the proposed gragnostic features, two gragnostics are new methods to measure graph-level features: line and tree. Three are based on existing graph-level features, but with proposals to normalize them to improve human interpretability: bridge, disconnection, and constriction. The other five are

Fig. 1. Example graphs illustrating each gragnostic (this figure excludes the nodes and links gragnostics).

existing graph-level features: Nodes (number of vertices), links (number of edges), density, isolation, and star.

A. Features

Each feature's range is $[0, 1]$, where 0 is a low gragnostic value, and 1 is a high value. The gragnostics are presented below, along with possible interpretations. These features assume simple undirected graphs, but many gragnostics can be adapted to directed graphs. 1 illustrates gragnostics with several graphs (excluding the nodes and links gragnostics).

Density: Density measures the interconnectivity of vertices within a graph. Higher density can indicate a higher rate of information flow. This uses the typical definition for undirected graphs: $\frac{2 \cdot |E|}{|V| \cdot (|V| - 1)}$. This is minimized in a graph with no edges, and maximized in a graph where an edge connects every pair of vertices. Calculating density requires a simple count of the vertices and edges, and therefore runs in $O(|V| + |E|)$ time.

Bridge: Bridges are edges whose removal will disconnect the graph, analogous to a bridge that connects two cities. Bridge measures relationships that are required for information to flow in the graph. Graphs that have a moderate values for bridge and density can indicate tight clustering. The bridge feature is calculated by $\frac{bridge(G)}{|V| - 1}$, where $bridge(G)$ is the number of bridges in graph G . There are at most $|V| - 1$ bridges in a graph, which occurs if the graph is a tree. The number of bridges can be calculated using Tarjan's algorithm, which runs in $O(|V| + |E|)$ time.

Disconnection: A high disconnection feature indicates that many vertices are disconnected from each other. This indicates little or no information flow, or broken or disrupted communication. Let C be the set of all maximally connected components of G . Then the disconnection gragnostic is defined by $\frac{|C| - 1}{|V| - 1}$. This is minimized when the graph is a single component, and maximized when there are no edges in the graph, i.e. the graph is completely disconnected. If $|C| = 1$ then disconnection is defined as 0. C can be calculated by running breadth-first searches to identify each component, which runs in $O(|V| + |E|)$ time.

Isolation: Isolation describes the fraction of vertices in the graph that have no edges connecting them to other vertices. Low isolation indicates that there is some information flow,

but there might or might not be clusters of high information flow. This is the definition given by Himelboim et al. [11]: $\frac{|\{v \in V: d(v) = 0\}|}{|V|}$. This is 0 when $|E| = 0$, and 1 when every vertex has at least one edge (i.e. when $\forall v \in V, d(v) > 0$). High isolation can indicate slow information flow or weak relationships. The degree of each vertex can be calculated with a simple iteration over each edge to count the number of edges incident on each vertex. Therefore isolation can be calculated in $O(|V| + |E|)$ time.

Constriction: The constriction gragnostic is similar to bridge. Constriction measures vertices that are required for information to flow in the graph. Graphs that have higher constriction and higher density can indicate tight clustering. Constriction is calculated by: $\frac{cut(G)}{|V| - 2}$, where $cut(G)$ is the number of cut vertices in G . If $|V| < 3$ then this gragnostic is defined as 0. Calculating $cut(G)$ is very similar to calculating $bridge(G)$, and therefore runs in $O(|V| + |E|)$ time.

Line (new): Measures how close a graph of $|V|$ vertices is to being the path graph P_n . Gragnostics uses the name "line" to avoid confusion with the other graph theory definition of path. Line measures the degree of sequential connections, such as hierarchies without branches or train stations without transfers. Let $\vec{D}$ be a vector of length $|V|$ and let $\vec{D}_i, \vec{D}_j$ be elements of $\vec{D}$ where $\vec{D}_i = d(v_i), \vec{D}_j = d(v_j)$ for $v_i, v_j \in V$ such that if $\vec{D}_i = 1$ and $\vec{D}_j > 1$ then $i < j$. For example, if G is a graph with four vertices that have degrees 1, 2, 2, and 3, then both $\langle 1, 2, 2, 3 \rangle$ and $\langle 1, 3, 2, 2 \rangle$ are valid $\vec{D}$ vectors. Then the line gragnostic is calculated by $\sum_{i=1}^{|V|} \frac{l(i)}{|V|}$, where

$$l(i) = \begin{cases} 1, & \text{if } \vec{D}_i = 1 \text{ and } i \leq 2, \text{ or if } \vec{D}_i = 2 \text{ and } i > 2 \\ 0, & \text{otherwise.} \end{cases}$$

In English, the line gragnostic is the fraction of vertices that have the correct degree of a path graph. This is derived from the fact that a path graph is a tree with two leaves, so a graph with $|V|$ vertices has 2 vertices with degree 1 and $|V| - 2$ vertices with degree 2. Creating $\vec{D}_i$ requires first calculating an array of degrees for each vertex, and then two linear-time loops over the array of degrees (one loop to find all degrees of value 1 and put them at the beginning of $\vec{D}_i$, and one loop to append the remaining degrees to $\vec{D}_i$). Therefore computing line runs in $O(|V| + |E|)$ time. If G has multiple components, then the line gragnostic is calculated as the arithmetic mean of the line feature of all components weighted by the number of vertices in each component.

Tree (new): Graphs that are tree-like can indicate hierarchy, dependency, or parent-child relationships. Trees are graphs with no cycles, and therefore a tree with $|V|$ vertices has $|V| - 1$ edges. Therefore, for a connected graph, we can calculate the tree gragnostic using the following definition: $1 - \frac{|E| - (|V| - 1)}{|V| \cdot (|V| - 1) / 2 - (|V| - 1)}$, where $|E| - (|V| - 1)$ is the number of edges needed to remove to make the graph a tree, and $|V| \cdot (|V| - 1) / 2 - (|V| - 1)$ represents the maximum possible number of vertices needed to remove to make the graph a tree (i.e. if G were a complete graph). If G is disconnected, then tree is calculated as the arithmetic mean of the tree feature of

all connected components weighted by the number of vertices in each component. If $|V| \leq 2$ then the tree gragnostic is defined as 0. This gragnostic is a simple algebraic equation involving the count of the number of vertices and edges, and therefore it runs in $O(|V| + |E|)$ time.

Star: The star gragnostic measures how much a graph with n vertices is like S_n , or the degree to which a single vertex is more central than the other vertices. Star measures hub-and-spoke relationships, or how much more central one vertex is than the others to the graph’s information flow. This is measured using Freeman’s degree centralization [19]: $\sum_{v \in V} \frac{d(v^*) - d(v)}{(|V|-1)(|V|-2)}$, where v^* indicates the vertex with the largest degree. For the purposes of gragnostics, if $|V| \leq 2$ and then star is defined as 0. The star gragnostic is minimized when all vertices have the same degree. For a connected graph G , centralization is maximized when one vertex has degree $|V|-1$ and all other vertices have degree 1, which is a star graph. If G has multiple components, then the star gragnostic is calculated as the arithmetic mean of the centralization of each component. This gragnostic is calculated by first calculating the degree of each vertex, and then computing the sum over all vertices, and therefore it runs in $O(|V| + |E|)$ time.

Nodes and Links: Because two graphs may have similar gragnostic features but be significantly different in size, the number of vertices and edges is useful for discriminating graphs or finding graphs of a particular size. Gragnostics uses the name “nodes” for vertices, and the name “links” for edges because these terms are not based on graph theory, and they are among the most commonly used words in present-day English¹. This makes the names more likely to be understood by analysts and users without a background in graph theory. The nodes gragnostic is simply $|V|$, and likewise links is $|E|$. These can be normalized to the range $[0, 1]$ for a set of graphs by calculating $|V|/|V^*|$ and $|E|/|E^*|$, where V^* and E^* indicate the largest set of vertices and edges, respectively, from the set of graphs.

B. Alternatives and Discussion

The gragnostics features were selected by surveying the literature and identifying features for measuring graph similarity to many types of fundamental graphs, such as trees, stars, and complete graphs. Only features that can be calculated in linear time were selected. Gaps in the set of features were identified by looking for common graph motifs and fundamental graphs that were not modeled by these metrics, such as trees and lines. Additional features could detect cycle graphs, but algorithms to count cycles are computationally expensive, and therefore not included. This is an avenue for future research. However, as shown below, gragnostics achieves high accuracy without including this feature.

There are other ways to calculate the above gragnostics. Line could be calculated by dividing the graph diameter by $|V| - 1$, but calculating diameter is slow (running time is

$O(|V|^2 \log |V| + |V| \cdot |E|)$). Similarly, the largest normalized betweenness centrality could be used for the star gragnostic, but this also runs in $O(|V|^2 \log |V| + |V| \cdot |E|)$ time. The star gragnostic could be calculated using the star magnostic, but that is also slower, and introduces challenges with finding a good ordering for the matrix visualization. For some existing graph types, such as stars, trees, or complete graphs, we could use binary tests to decide if the graph matches the definition, but this can create large distances in the feature space between graphs that are very similar (e.g. if removing a single edge would make it a tree). Edit distance could give a more nuanced notion of similarity, but this is also slow.

The gragnostic feature names were chosen to improve understandability by laypeople. The names minimize the use of obscure graph theoretic terminology that is primarily only understood by graph theoreticians. Most names are in the list of 10,000 most commonly used words or in Basic English. This helps ensure that the gragnostic feature names will be easily understood by users.

IV. EVALUATIONS

There are three evaluations that examine different aspects of gragnostics: (1) The independence of the features, (2) the effectiveness of gragnostics for differentiating graphs, and (3) the effectiveness of gragnostics for visual analytic applications. The first two evaluations use a dataset of 92 graphs comprising the following seven classes of graphs:

- **Artificial:** graphs generated using the Barabási-Albert [20] and Watts-Strogatz algorithms [21] (six graphs from each, with 250, 500, or 1000 vertices).
- **Character:** Fictional character interaction networks from Anna Karenina [22], The Winter King (The Arthur Books #1) [23], David Copperfield [22], The Hobbit [23], Huckleberry Finn [22], Les Misérables [22], Star Wars 1–7 [24], and Storm of Swords [23].
- **Collaboration:** Research paper co-authorship graphs [25]–[28], such as Network Science, CHI, KDD, and arXiv networks including Astrophysics and High-Energy Theory.
- **Ego:** Ego networks of users from Facebook [29].
- **Geometric:** Graphs that exhibit regular geometric structure, such as cubical, dodecahedral, grid, hypercube, icosahedral, octahedral, and S_6 graphs.
- **Software:** Software class dependency networks [30]–[34], such as junit, jmail, jung, guava, weka, etc.
- **Subway:** Graphs of subway networks from cities in 2009 [35].

In the evaluations, the nodes and links gragnostics were scaled to the range $[0, 1]$. The evaluations began by extracting the 10 gragnostic features for all 92 graphs, which took about 24 seconds on the test computer (a 2016 MacBook Pro with a 2.7 GHz Intel Core i7 processor and 16 GB of RAM).

A. Are Gragnostic Features Sufficiently Independent?

This evaluation examines theoretically and empirically whether there are redundancies in the gragnostics features.

¹<https://github.com/first20hours/google-10000-english>
<http://ogden.basic-english.org/words.html>

Graph class	Number	$ V $	$ E $
artificial	12	250 – 1,000	741 – 4,000
character	14	18 – 138	41 – 493
collaboration	12	1,001 – 20,046	2,627 – 198,110
ego	10	53 – 1,035	198 – 30,772
geometric	18	4 – 20	5 – 32
software	11	128 – 2,956	310 – 10,845
subway	15	82 – 433	85 – 475

TABLE I

SUMMARY OF THE 92 EVALUATION GRAPHS: THE NUMBER OF GRAPHS IN EACH CLASS, AND THE VERTEX AND EDGE RANGES.

Fig. 2. The correlation between each gragnostic on the 92 graphs.

Theoretically, we expect some correlation between nodes and links, since graphs with more vertices can have more edges; we expect some correlation between links and density, since graphs with more edges tend to become more dense; we expect some correlation between isolation and disconnection, since a vertex with degree 0 forms its own component; we expect some correlation between tree, star, and line, since stars and lines are special cases of trees; we expect some negative correlation between density and tree, star, and line, since trees are not dense by definition; and we expect some correlation between bridge and constriction, since a bridge edge necessarily has two cut vertices. However, each feature measures a distinct topological characteristic, and therefore they will probably not be highly correlated in practice.

Empirically, we can compute the correlation between each feature using the 92 evaluation graphs described above to examine their correlation in a practical example. Since we are primarily concerned with the degree of correlation, and not necessarily with whether the correlation is positive or negative, this evaluation computes the absolute value of the correlation. This is show in Fig. 2. The highest correlation is between the nodes and links features, which has a value of 0.787. These two features are arguably important for differentiating graphs of different sizes that otherwise have similar topologies. Since all other correlations are smaller, we argue that the other features are sufficiently independent and should be included. With regards to the other feature correlations, note that our predictions about correlation were also correct for tree and density, bridge and constriction, and disconnection and isolation. However, our predictions were wrong about correlation between links and density; tree, star, and line; and density and star and line. The absolute value of all these correlations are below 0.5.

Fig. 3. LEFT: Accuracy and precision@ k for each tested value of k in the KNN classifiers using gragnostics and DDQC. RIGHT: The 13 graphs misclassified by a KNN classifier, and the number of classifiers that misclassified each graph. The other 79 graphs were always correctly classified.

B. Can Gragnostics Effectively Differentiate Graphs?

To evaluate gragnostics in a machine learning scenario, this evaluation uses a k -nearest neighbors (KNN) classifier. The classification task is to predict a graph’s class (e.g. artificial, character, etc.) from the graph’s gragnostic features. Past research indicates different classes have different structure [16], [36]. The KNN classifier uses Euclidean distance to find the k nearest neighbor graphs, and uses inverse distance-weighted voting to predict the class from the nearest neighbors. (Note that a different classifier, such as a support vector machine with a Gaussian kernel, might perform better by identifying non-linear relationships between features. But this might require a larger dataset for training.)

The evaluation uses a leave-one-out cross validation, and uses KNN classifiers with k from 1 to 10 (10 is the number of graphs in the ego graph class, which is the smallest class). The evaluation metrics are accuracy (fraction of classifications that are correct), and precision@ k ($P@K$, mean fraction of k nearest neighbors with the correct class). Fig. 3 shows the accuracy and $P@K$ for each value of k . Accuracy is high (0.90-0.93) and fairly robust to the value of k . $P@K$ is similarly high for small values of k . Fig. 3 shows the 13 graphs that were misclassified along with the number of classifiers that misclassified them; the remaining 79 graphs were always correctly classified. Interestingly, we see that the artificial graphs are never misclassified. This is surprising because they are designed to generate graphs that mimic human social interaction graphs, and there are several in this dataset (collaboration graphs, character interaction graphs, and ego social graphs). This indicates that these models do not capture some essential characteristics found in real and fictional human interaction graphs.

If we remove the four most highly correlated features (links, disconnected, bridge, and tree), we can test whether the extra features provide a useful increase in classification performance. By doing this and running the same experiment above

Fig. 4. A multi-dimensional scaling plot showing distances between 92 graphs using gragnostics. In general, there is good class separation, although the character and software classes overlap. The figure calls out Tokyo, David Copperfield, and sjbullet in **bold** and shows their gragnostic values and a bar chart of their 10 nearest neighbors and the distances to those neighbors. The two nearest neighbors are also called out for comparison, allowing us to see that the nearest neighbors do indeed have similar topological structure.

using only the 6 features, the accuracy drops to 0.78 to 0.80, depending on the value of k . Similarly, the precision drops to 0.66 to 0.82. Additionally, 25 graphs were misclassified for at least one value of k , with 13 graphs always misclassified regardless of the value of k . Therefore, the extra four features appear to be beneficial for improving classifier performance.

This evaluation compares Degree Distribution Quantification and Classification (DDQC) [16]—another set of graph-level features used for visualization [14]—to gragnostics. This evaluation does not include other feature sets because of known limitations: edge-based feature vectors require graphs to be the same size, many graph kernels only work on graphs with one component, and many other common graph statistics such as diameter or betweenness centrality run in $O(|V|^2)$ time or slower. For these reasons, this evaluation only compares gragnostics and DDQC. See Section II for further discussion.

The same experiment was run with DDQC [16] features instead of gragnostics, but accuracy and $P@K$ are considerably worse (see 3). In addition, 41 graphs were misclassified at least once, and 23 were misclassified by all 10 classifiers.

Fig. 4 shows a multi-dimensional scaling (MDS) plot [37]

of the 92 graphs projected onto 2 dimensions from the 10-dimensional gragnostic feature space. We see good overall class separation, indicating that each graph class has different gragnostics values. We do see a few exceptions, which correspond to graphs misclassified by the KNN classifiers. For example, the David Copperfield graph’s nearest neighbors are all Facebook ego networks. This classification makes sense if we consider that David Copperfield is often considered to be a semi-autobiographical novel about Charles Dickens, in essence making it an ego graph of the central character. The Les Miserables and Storm of Swords graphs are more typical of fictional character graphs. The software and the character classes overlap in the MDS plot. The sjbullet software graph’s gragnostics are similar to the Storm of Swords character graph, although it is larger and less dense. We can visually confirm the topological similarity by comparing the two force-directed node-link diagrams. Finally, let’s examine the Tokyo subway graph: its distance to the London subway graph is very short. Their gragnostics are nearly identical, and their force-directed node-link diagrams share the same visual structure. Meanwhile, the Tokyo graph’s second nearest neighbor is the

Fig. 5. Gragnostics can be used to identify patterns and features of subgraphs in Chiron, a visual analytic tool for exploring subgraphs. On the left is the distribution of gragnostics and other subgraph attributes for the CHI dataset. The *Suggested subgraphs* panel shows a list of subgraphs, in the middle is an overview of the entire graph, and on the right is a detailed view of one of the communities. This subgraph has a relatively high star feature, where a few authors tend to work with many other authors. In this dataset, this tends to indicate advisor-student relationships. Vertex color indicates degree, vertex and edge size indicate number of papers.

Shanghai subway graph, which is farther away than London. Shanghai has higher bridge, constriction, and line gragnostics. We can visually confirm this dissimilarity by looking at Shanghai’s force-directed node-link diagram and noting that it has more bridge edges, it has more constriction points, and it is more line-like because more vertices have only two edges.

C. Are Gragnostics Effective for Visual Analytics?

This evaluation examines whether gragnostics can be effective for visual analytics applications. This is performed by implementing gragnostics in a rank-by-feature framework in the Chiron graph visualization tool. Chiron is designed to explore subgraphs within a larger graph. Chiron has four main user interface components. (1) The *Detailed overview* panel in the center of the user interface presents a zoomable and pannable overview of the graph. Node-link visualizations of large graphs are often criticized as being uninterpretable. Although they might not convey detailed information, they can provide a useful overview [18]. For example, the overview in Fig. 5 shows that there is one very large component and many very small components. This can be useful information to help users decide how to filter and select subgraphs. (2) The *Suggested subgraphs* panel is a sortable and filterable list of all identified subgraphs of the graph. The panel shows the attributes and gragnostic feature values for each subgraph. (3) The *Settings panel* controls filtering and rendering of the subgraphs. Scented range sliders [38] show users the distribution of subgraphs for each attribute and gragnostic feature (see the left side of Fig. 5). Users can also filter out clusters of similar subgraphs. (Clusters are determined by clustering the subgraphs on the gragnostic features using k -means clustering. Chiron tests a range of values for k , and then chooses the clustering that achieves the highest average silhouette score.) (4) When a user selects a subgraph from the *Suggested subgraphs* panel, it is then highlighted in

the *Detailed overview* with a convex hull and displayed for detailed analysis in the *Selected subgraph* panel on the right. The *Selected subgraph* panel allows users to compute better layouts, and click on vertices and edges to see their details. Fig. 5 shows a screenshot of these panels in Chiron.

Chiron can import Graphviz dot files. In the dot file, users specify a list of subgraphs that each vertex belongs to using a custom attribute called `subgraphs`. Each entry in the list contains the ID of the subgraph, and any optional subgraph attributes, such as outlier score. Chiron uses these attributes to create sliders in the settings panel for users to sort and filter the subgraphs (see Fig. 5). Some examples of subgraphs are the graph’s biconnected components or vertices grouped by an attribute. If users have not specified any subgraphs, Chiron can run the Louvain community detection algorithm [39] to create subgraphs when the dot file is imported. During import, Chiron also computes the gragnostic features for each subgraph.

Below are two example analyses using gragnostics and Chiron’s rank-by-feature functionality to analyze two graphs.

csranks: The *csranks* dataset is a collaboration graph of faculty at top computer science schools based on publications in the most selective conferences [40]. The dataset also includes the raw count of papers an author has published at top conferences, an adjusted count weighted by the number of co-authors, and the area of each publication (e.g. AI, visualization, HCI, etc.). A custom Python script parsed the dataset to create a Graphviz dot file to import into Chiron, and vertex positions were generated with a force-directed group-in-a-box layout [41]. Vertices represent authors, and edges represent a co-authorship relationship between authors. The resulting graph contains 5,995 vertices and 15,094 edges. Initially, vertices are grouped into subgraphs by the author’s affiliation. In this grouping, there are 192 subgraphs.

After loading the dataset in Chiron, in the *Detailed overview* panel we see a large group of vertices that appears to be a

Fig. 6. Subgraphs from csrankings. In the U. of Pittsburgh collaboration subgraph (A) vertex color is the count of papers published, and vertex and edge size are the number of areas. In (B) and (C) vertex color indicates affiliation, vertex size is the number of areas, and edge size is the count of papers. In (B) and (C), intra-institution collaboration is more common than inter-institution, but interestingly David R. Cheriton connects researchers from the U. of Waterloo.

component, with several isolated vertices (degree 0) on the perimeter. Next, let’s adjust the filters to show only subgraphs with low star, density, and disconnection features. Following the typology from Himelboim et al. [11] this should show us CS departments at universities that are clustered, i.e. there are some groups of researchers who collaborate, but there is no collaboration between groups. We choose the disconnection feature instead of the isolation feature because we are looking for subgraphs with many components, and not necessarily many isolated researchers who do not collaborate with anyone. Sorting the subgraphs by descending disconnection feature reveals several CS departments that are clustered. For example, in the University of Pittsburgh subgraph (6A), we find five authors publishing work in AI, NLP, and machine learning, but only two of them have collaborated, producing one paper in this dataset. In contrast, there is a cluster of six authors whose work centers around architecture, design automation, software engineering, embedded systems, high performance computing, communication, and data management. There is also a third cluster who have published mostly in data management.

Next, let’s discard the subgraphs grouped by affiliation and run the Louvain community detection algorithm on the graph. This gives us 1092 subgraphs, with 932 that are only a single vertex, and 107 that only have two vertices. We discard these, and import the remaining 53 subgraphs. Filtering by the number of affiliations in each subgraph, we find 12 subgraphs that only have one affiliation. All of these have either three or four vertices. The subgraph with the highest number of affiliations has 132 affiliations.

Next, we reset the filters and filter out subgraphs with small numbers of vertices, and sort by the star feature. We see a few subgraphs like that in Fig. 6B, which has one vertex from one university that is a cut vertex connecting two researchers from a different university. This is David R. Cheriton, who

is a professor at Stanford University, but he is known for his strong ties to Waterloo. Sorting by star in descending order, we see a common pattern where authors from the same institution collaborate, and one or two of those authors will then collaborate with members of another institution.

CHI: The CHI dataset is the co-authorship network of the Conference on Human Factors in Computing Systems from 1990 to 2014. Vertices represent authors, and an edge denotes that two authors co-authored at least one paper together. Vertices and edges have a `weight` attribute that indicates the number of papers written by an author, or co-authored by two authors. There are 20,046 vertices and 54,111 edges.

Let’s begin by using the Louvain community detection algorithm to generate subgraphs, and discarding all subgraphs with fewer than four vertices. This leaves 707 subgraphs. The overview (see Fig. 5) shows one large central component of vertices, and many small components on the periphery. Chiron’s subgraph clustering functionality is especially useful here: Cluster 0 (see the settings panel on the left of Fig. 5) is composed only of small cliques. By unselecting this and other clusters of small, peripheral communities, we filter the suggested subgraphs down to 84 subgraphs. The subgraphs tend to have low density, and there are 54 subgraphs that have a tree feature greater than 0.9. In these communities, we see many vertices that connect local areas of density; indeed, many of the subgraphs have a relatively high star feature. This indicates that communities in the CHI research community are sparse, but there are a few key authors who connect the community (see the selected subgraph on the right of Fig. 5).

V. DISCUSSION

This paper introduced two graph-level features (tree and line) and combined them with eight existing graph-level features to form gragnostics, a set of $O(|V| + |E|)$ human-understandable features for comparing graphs. This paper

evaluated gragnostics for graph classification, and showed their utility in a rank-by-feature framework for exploring subgraphs.

This paper shows that gragnostics are a capable tool for classifying graphs, understanding the classifications, and finding interesting subgraphs within a larger graph. A byproduct of this research shows that artificial and geometric graphs are easily distinguishable from real-world graphs, suggesting that researchers should carefully choose datasets for research studies. Gragnostics-based classifiers outperformed DDQC-based classifiers. Like Scagnostics, some gragnostics are correlated; however, in practice the features are sufficiently uncorrelated such that removing them degrades classification performance.

We found Chiron's gragnostics-based rank-by-feature framework quite useful for exploring the subgraphs in the csrankings and CHI datasets, and it guided us to insights about specific subgraphs as well as patterns across many subgraphs. The ability to sort and cluster subgraphs proved useful, replicating the findings of other researchers [5]–[7].

ACKNOWLEDGMENT

This work was funded by the Defense Advanced Research Projects Agency (DARPA). The views expressed are those of the author(s) and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

REFERENCES

- [1] A. Sopan, P. Rey, and B. Shneiderman, "The dynamics of web-based community safety groups: Lessons learned from the nation of neighbors," *IEEE Signal Processing Magazine*, vol. 30, no. 6, pp. 157–162, 2013.
- [2] O.-H. Kwon, T. Crnovrsanin, and K.-L. Ma, "What Would a Graph Look Like in This Layout? A Machine Learning Approach to Large Graph Visualization," *IEEE TVCG*, vol. 24, no. 1, 2018.
- [3] A. Perer and B. Shneiderman, "Balancing systematic and flexible exploration of social networks," *IEEE TVCG*, vol. 12, no. 5, pp. 693–700, 2006.
- [4] B. Shneiderman and A. Aris, "Network Visualization by Meaningful Substrates," *IEEE TVCG*, vol. 12, no. 5, pp. 733–740, 2006.
- [5] M. Freire, C. Plaisant, B. Shneiderman, and J. Golbeck, "ManyNets: An interface for multiple network analysis and visualization," in *CHI*, 2010, pp. 213–222.
- [6] M. Harrigan, D. Archambault, P. Cunningham, and N. Hurley, "Egonav: Exploring networks through egocentric spatializations," in *Proc. AVI*, ACM, 2012, pp. 563–570.
- [7] T. von Landesberger, M. Gerner, and T. Schreck, "Visual analysis of graphs with multiple connected components," in *Proc. VAST*, IEEE, 2009, pp. 155–162.
- [8] Z. C. Lipton, "The mythos of model interpretability," *ACM Queue*, vol. 16, no. 3, p. 30, 2018.
- [9] J. Seo and B. Shneiderman, "A rank-by-feature framework for interactive exploration of multidimensional data," *Information Visualization*, vol. 4, no. 2, pp. 99–113, 2005.
- [10] L. Wilkinson, A. Anand, and R. Grossman, "Graph-theoretic scagnostics," in *Proc. IEEE Info Vis*, 2005, pp. 157–164.
- [11] I. Himelboim, M. A. Smith, L. Rainie, B. Shneiderman, and C. Espina, "Classifying Twitter Topic-Networks Using Social Network Analysis," *Social Media + Society*, vol. 3, no. 1, pp. 1–13, 2017.
- [12] M. E. J. Newman, *Networks: An Introduction*. Oxford University Press, 2010.
- [13] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic, "Time curves: Folding time to visualize patterns of temporal evolution in data," *IEEE TVCG*, vol. 22, no. 1, pp. 559–568, 2016.
- [14] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk, "Reducing snapshots to points: A visual analytics approach to dynamic network exploration," *IEEE TVCG*, vol. 22, no. 1, pp. 1–10, 2016.
- [15] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Art. Int. and Stat.*, 2009, pp. 488–495.
- [16] S. Aliakbari, J. Habibi, and A. Movaghar, "Feature extraction from degree distribution for comparison and analysis of complex networks," *Comput J*, vol. 58, no. 9, pp. 2079–2091, 2015.
- [17] J. Guerra-Gomez, A. Wilson, J. Liuy, D. Davies, P. Jarvis, and E. Bier, "Network explorer: Design, implementation, and real world deployment of a large network visualization tool," in *Proc. AVI*, 2016, pp. 108–111.
- [18] P. Eades, S.-H. Hong, K. Klein, and A. Nguyen, "Shape-based quality metrics for large graph visualization," in *Graph Drawing*. Springer, 2015, pp. 502–514.
- [19] L. C. Freeman, "Centrality in social networks conceptual clarification," *Social Networks*, vol. 1, no. 3, pp. 215–239, 1978.
- [20] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [21] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
- [22] D. E. Knuth, *The Stanford GraphBase: A Platform for Combinatorial Computing*, 1st ed. ACM Press, 1994.
- [23] A. J. Holanda, M. Matias, S. M. S. P. Ferreira, G. M. L. Benevides, and O. Kinouchi, "Character Networks and Book Genre Classification," *ArXiv e-prints*, 2017.
- [24] E. Gabasova. (2015) The Star Wars social network. [Online]. Available: <https://github.com/evelinag/StarWars-social-network>
- [25] S. Kobourov, S. Pupyrev, and P. Simonetto. (2017) GMap Datasets. [Online]. Available: <http://gmap.cs.arizona.edu/datasets>
- [26] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph Evolution: Densification and Shrinking Diameters," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, 2007.
- [27] M. E. J. Newman, "The structure of scientific collaboration networks," *Proc. Natl. Acad. Sci.*, vol. 98, no. 2, pp. 404–409, 2001.
- [28] —, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E*, vol. 74, no. 3, p. 036104, 2006.
- [29] J. McAuley and J. Leskovec, "Learning to Discover Social Circles in Ego Networks," in *Proc. NIPS*. Curran Associates Inc., 2012, pp. 539–547. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2999134.2999195>
- [30] L. Šubelj and M. Bajec, "Community structure of complex software systems: Analysis and applications," *Physica A Stat. Mech. Appl.*, vol. 390, no. 16, pp. 2968–2975, 2011.
- [31] —, "Clustering assortativity, communities and functional modules in real-world networks," *ArXiv e-prints*, 2012.
- [32] —, "Software systems through complex networks science: Review, analysis and applications," in *Proc. KDD Workshop on Software Mining*, 2012, pp. 9–16.
- [33] L. Šubelj, M. Bajec, and N. Blagus, "Group extraction for real-world networks: The case of communities, modules, and hubs and spokes," in *Proc. Net. Sci.*, 2013, pp. 152–153.
- [34] L. Šubelj, S. Žitnik, N. Blagus, and M. Bajec, "Node mixing and group structure of complex software networks," *Advances in Complex Systems*, vol. 17, no. 7, p. 1450022, 2014.
- [35] C. Roth, S. M. Kang, M. Batty, and M. Barthelemy, "A long-time limit for world subway networks," *J. R. Soc. Interface*, 2012.
- [36] R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, and S. Shen-Orr, "Superfamilies of evolved and designed networks," *Science*, vol. 303, no. 5663, pp. 1538–1542, 2004.
- [37] I. Borg and P. J. Groenen, *Modern Multidimensional Scaling: Theory and Applications*. Springer Science & Business Media, 2005.
- [38] W. Willett, J. Heer, and M. Agrawal, "Scented widgets: Improving navigation cues with embedded visualizations," *IEEE TVCG*, vol. 13, no. 6, pp. 1129–1136, 2007.
- [39] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J Stat Mech-Theory*, vol. 2008, no. 10, p. P10008, 2008.
- [40] E. Berger. (2017) Computer Science Rankings. [Online]. Available: <http://csrankings.org/>
- [41] S. Chaturvedi, C. Dunne, Z. Ashktorab, R. Zachariah, and B. Shneiderman, "Group-in-a-Box Meta-Layouts for Topological Clusters and Attribute-Based Groups: Space-Efficient Visualizations of Network Communities and Their Ties," *CGF*, vol. 33, no. 8, pp. 52–68, 2014.