TABLE OF CONTENTS

Contributed Articles

Association for Computing Machinery

Advancing Computing as a Science & Profession

Higher-Order Networks Representation and Learning: A Survey

Hao Tian and Reza Zafarani
Data Lab, EECS Department, Syracuse University
{haotian,reza}@data.syr.edu

ABSTRACT

Network data has become widespread, larger, and more complex over the years. Traditional network data is *dyadic*, capturing the relations among pairs of entities. With the need to model interactions among more than two entities, significant research has focused on *higher-order networks* and ways to represent, analyze, and learn from them. There are two main directions to studying higher-order networks. One direction has focused on capturing *higher-order patterns* in traditional (dyadic) graphs by changing the basic unit of study from nodes to small frequently observed subgraphs, called *motifs*. As most existing network data comes in the form of pairwise dyadic relationships, studying higher-order structures within such graphs may uncover new insights. The second direction aims to directly model higher-order interactions using new and more complex representations such as *simplicial complexes* or *hypergraphs*. Some of these models have long been proposed, but improvements in computational power and the advent of new computational techniques have increased their popularity. Our goal in this paper is to provide a succinct yet comprehensive summary of the advanced higher-order network analysis techniques. We provide a systematic review of the foundations and algorithms, along with use cases and applications of higher-order networks in various scientific domains.

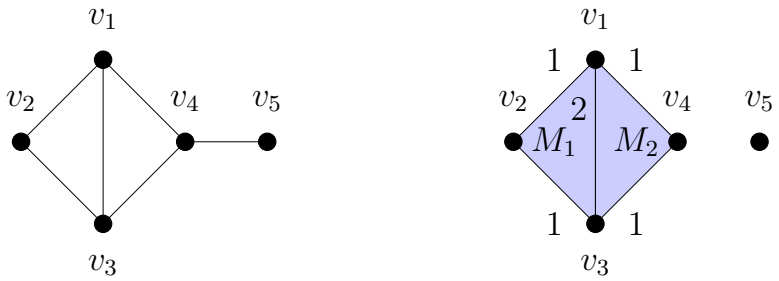
1. INTRODUCTION

Networks are natural representations of relationships between entities using nodes and edges [8]. Real-world networks are observed everywhere: the structure of chemical substances, ecological systems, communication networks, air and land transportation networks, power grids, among many other examples. Many problems can be naturally described and solved using networks. For instance, epidemic models on networks [45] can help predict the spread of pandemics by analyzing the interaction network among infected individuals; link-analysis methods such as PageRank [86] can help assess the importance of Websites, which in turn can be used to fine-tune searching engine results; Shortest paths algorithms [71] calculate the most efficient driving route between two locations on the transportation networks. With the rise in demand to model systems with more complex information, researchers have enriched network models by adding additional attributes to nodes and edges: weights, signs, labels, timestamps, and even metadata. Some models have even changed or extended the network basics. An example is a *heterogeneous network* [125], where nodes and edges can be of different types. Another example is a *dynamic network* [93], where each node or edge can exist only for a specific period of time. While models for networks have been enriched from various aspects, in most network models, edges still represent *dyadic* relationships, that is, relationships among two entities.

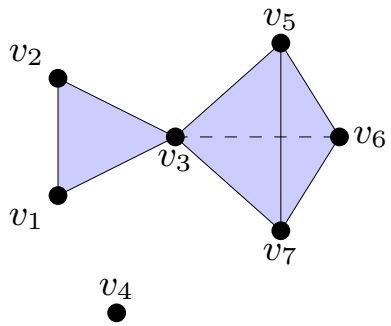
Dyadic relationships are insufficient in many real-world scenarios, specifically when there is an interaction involving more than two entities. For example, a social event may include more than two people. This is not equivalent to social interactions among all pairs of people in the event. However, such a *higher-order* pattern frequently occurs in social networks due to *triadic closure* [41], where a triangle's formation is often dependent on three edges. Another example is the frequent appearance of some specific small subgraphs (with more than two nodes) in real-world networks [10; 62; 115]. In 2002, Shen-Orr et al. [102] introduced the term *network motifs* to represent such frequent subgraphs as the building blocks of transcriptional regulation networks. The idea was further explored for various types of graphs by Milo et al. [76], where they showed that different types of networks can be distinguished using motif counts as features [75]. Such discoveries clearly indicate that *it is insufficient to only model dyadic networks*.

As a result, modeling higher-order networks has a long history. Some higher-order representations are proposed as early as the 1960s-1970s, e.g., *simplicial complexes* [107]. However, there were many obstacles to utilizing such higher-order representations at that time. First, the computational power was insufficient to compute using such representations, even for counting simple motifs. Second, without the development of the internet, data collection was inefficient and small-scale. Hence, there was an earlier decline in the demand to model and represent higher-order networks.

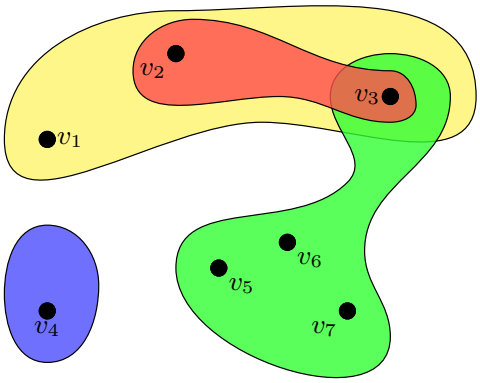
With the fast development of computational resources and algorithmic tools, higher-order network analysis is now widely used across various fields leading to various discoveries. For instance, in brain networks, some motifs with high functionality are generated more than other motifs to increase neural efficiency [108; 31]; In biological networks, motifs are commonly found and capture evolution [53]; In social networks, motifs help understand group interactions [63; 47]. As most existing network data is already collected as dyadic graphs, it is often impossible to recover the original higher-order interactions (if they exist). In recent years, more higher-order network data is collected, e.g., in collabo-



(a) Simple graph and its corresponding motif graph



(b) Simplicial complex



(c) Hypergraph

Figure 1: Comparison among Higher-Order Network Representations. (a). A simple graph (left) along with its *motif graph* (right) of *3-Cliques* (triangles). With the same set of nodes, motif graphs transform edges into "membership in given motifs." From the example, the given motif is a triangle, so there are two triangles detected and only one edge (v_1, v_3) shared by both triangles. The edge (v_4, v_4) that does not belong to any motif will be ignored. (b). A Simplicial Complex, including a 0-simplex (single node), a 2-simplex (triangle), and a 3-simplex (tetrahedron). Note that any *face* (sub-simplex) of an existing simplex is also included in the simplicial complex, for example, (v_5, v_6, v_7). (c). A Hypergraph. Unlike simplicial complexes, any subedge of hyperedges does not have to appear in the edge set, i.e., (v_1, v_2, v_3) and (v_2, v_3) are two different hyperedges.

ration networks or on hashtags [12]. As a result, more complex higher-order algorithms and representations can now be directly used. Our goal is to broadly survey such ways to represent, analyze, and learn from higher-order networks.

Higher-Order Network Representations: there are three main branches of network abstractions that are widely studied or utilized for higher-order networks — *motifs*, *simplicial complexes*, and *hypergraphs*. Before diving into detailed mathematical representations of them, we briefly describe their differences using concrete examples involving three individuals `A`, `B`, and `C` [13]:

1. **Network Motifs**: `A`, `B`, and `C` have contact information of **each other** in their contact list. They form a triangle (an example of a *motif*).

2. **Simplicial Complexes**: `A`, `B`, and `C` are in the same class in high school. **Any subset of {A, B, C}** indicates a classmate relationship. `A`, `B`, and `C` form a *simplex*, a basic unit of a *simplicial complex*.

3. **Hypergraphs**: `A`, `B`, and `C` publish a paper together. We create a new *hyperedge* representing the collection of **all authors** of this paper, i.e., hyperedge {`A`, `B`, `C`}. `A`, `B`, and `C` form a *hypergraph* with a single hyperedge. Note that {`A`, `B`},{`A`, `C`} and {`B`, `C`} are not included as hyperedges in the hypergraph.

Figure 1 shows a comparison of these network representations. We still often study motifs in dyadic graphs, but instead of looking at pairwise relationships, we extract higher-order relationships. For example, we can identify whether an edge is part of some prespecified motif and set edge weights to the number of times the edge belongs to such motif. As shown in Figure 1 (a), one has to prespecify a motif to study, for example, a triangle. Then the original graph can be transformed by only keeping its edges that belong to such a given motif. Figure 1 (b) and (c) show a simplicial complex and a hypergraph with similar structures. Both edges (simplexes) can be represented by sets. However, in simplicial complexes, any subset simplex, by definition, also exists; however, in hypergraphs, a hyperedge only acknowledges the existence of the exact set. For example (v_1,v_2,v_3) and (v_2,v_3) exist in Figure 1 (c), while their subedges (v_1,v_2) and (v_1,v_3) do not exist.

Topics not covered in this survey. Some studies are outside of the scope of this survey and presenting them might obfuscate some of the formalism used in this survey. These studies either (1) apply a higher-order abstraction but outside the field of networks; or (2) use similar terminologies as those of higher-order networks but with a different meaning across various domains. Here, we list some such areas:

- *Network of Networks (NoN)*, or *multilayer networks* [65; 56; 23], are basically heterogeneous networks. Such models combine networks from different sources and merge into a larger, more complex network. Here, nodes and edges can be different kinds of entities. The basic unit of such networks is one type of network, but not subgraphs.

- *Higher-Order Markov Models*: These studies investigate higher-order dependencies in random walks on networks [58; 73; 123]. In some cases, first-order random walks are not able to describe network flows, so it becomes necessary to utilize higher-order Markov chains to fit the real-world observations in networks.

- *Higher-order Graph Signal Processing*: Graph signal processing [84] has also been extended from dyadic networks to simplicial complexes [96; 103; 95]. In graph signal processing, vertices carry samples of signals, and edges capture linear transformations on such signals. Such a system is denoted by a *graph filter*, which aims to model complex signal transformations based on a graph structure. For a comprehensive review of higher-order graph signal processing, refer to [98].

- *Higher-order Dynamical Systems on Networks*: A network dynamical system models pairwise node interactions based on a dynamic network structure [19]. For example, robots can form real-time shapes together by observing their neighbors. Such pairwise interactions are extended to higher-order interactions [17], using either simplicial complexes [74] or hypergraphs [80].

Scope and Organization. Compared to other surveys, our goal is to provide a succinct yet broad and comprehensive survey that focuses on higher-order networks. Abstract network topologies are categorized on the basis of the data type and applications. The surveys aims to assist researchers in identifying the appropriate methods, resources, and higher-order tools for their research tasks.

The rest of this paper is structured as follows. Section 2 introduces necessary dyadic graph basics. Sections 3, 4, and 5 introduce foundations, algorithms, and applications of network motifs, simplicial complexes, and hypergraphs, respectively. Section 6 summarizes existing datasets and tools that have been used in higher-order network studies. We present challenges and future direction and conclude in Section 7.

2. PRELIMINARY OF GRAPHS

We briefly introduce some basic dyadic graph concepts, which are being applied and generalized in various higher-order representations.

- Undirected/Directed Graph: An *undirected graph* is an ordered pair $G = (V, E)$. Set $V = \{v_1, v_2, ..., v_n\}$ is the set of vertices and set $E \subseteq \{\{v_i, v_j\} | v_i, v_j \in V\}$ is the set of edges, where $\{v_i, v_j\}$ is an unordered pair of nodes. In contrast, a *directed graph* is an ordered pair $G = (V, E)$, where V is the set of vertices and $E \subseteq \{(v_i, v_j) | (v_i, v_j) \in V^2\}$ are ordered pairs of nodes.

- Simple Graph: A *self-loop* is an edge that starts and ends at the same vertex, for example, (v_i, v_i). Duplicate edges in an edge set are called *multiple edges*, *multi-edges*, or *parallel edges*. A graph without any self-loop or multiple edges is a *simple graph*. Most graph-based studies focus on simple graphs.

- Weighted Graph: A *weighted graph* $G = (V, E, w)$ is a graph with assigned weights $w : E \to \mathbb{R}$ to its edges.

- Graph Isomorphism: Graphs G and H are *isomorphic*, denoted as $G \simeq H$, if there is a bijection between the vertex sets of G and H, denoted by $V(G)$ and $V(H)$,

$$f : V(G) \to V(H),$$

such that any two vertices u and v of G are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in H.

- Walk, Trail, Path and Cycle: A *walk* is a sequence of vertices and edges of a graph. For example, one can traverse from one vertex to another once there is an edge between them. A walk is said to be *open* when the starting and ending nodes are different, and *closed*, otherwise. A *trail* is an open walk where no edge is repeated. A *path* is a trail in which neither a vertex nor an edge is repeated. A *cycle* is a closed walk that neither a vertex nor an edge is repeated.

- Cut, Volume, and Conductance: A *cut* is a partition of the vertices of a graph into two disjoint subsets, marked as $(S, \bar{S})$. The *volume* of a node set $S \subseteq V$ is defined as the total number (or weight) of the edges incident with S, denoted as $\text{vol}(S)$. The *conductance* for set S, denoted as $\varphi(S)$, measures the 'goodness' of a cut separating a graph,

$$\varphi(S) = \frac{\sum_{i \in S, j \in \bar{S}} A_{ij}}{\min(\text{vol}(S), \text{vol}(\bar{S}))},$$

where A_{ij} are the entries of the adjacency matrix for G. Lower conductance ensures a balanced cut with fewer cross-edges (between S and $\bar{S}$).

- Adjacency Matrix: a square matrix used to represent a finite graph. Assume a graph has n nodes. Its corresponding adjacency matrix A is a matrix of size $n \times n$, where $A_{i,j} = 1$ when nodes i and j are connected and $A_{i,j} = 0$, otherwise. Adjacency matrices of undirected graphs are symmetric. Adjacency matrices of simple graphs are binary, with all zeros on the main diagonal.

- Laplacian Matrix: Given a simple graph G, the *Laplacian Matrix* of G is defined as $L = D - A$, where D is the diagonal degree matrix and A is the adjacency matrix of G. If a graph G is undirected, its *Laplacian Matrix* is also a symmetric matrix. It can be normalized to matrix of unit vectors, usually denoted as $\mathcal{L} = D^{-1/2} L D^{1/2} = I - D^{-1/2} A D^{1/2}$.

3. NETWORK MOTIFS

Compared to more complex higher-order network representations, motifs have been studied for a relatively longer period. There are two main reasons: (1) motifs are studied directly on dyadic graphs, which are widely used in various research fields; (2) some specific subgraphs already have some special meanings in the real world, so it is natural to study them in networks.

3.1 Foundation and Algorithms

A network motif is generally defined as **a highly significant subpattern or subgraph in the network** [76]. The term "significant" indicates that the number of times the motif appears in the graph is higher than what is expected or normal, where such expected numbers are often understood within *random graphs* (e.g., the *Erdős–Rényi model* [32]). A motif can be some fixed-size subgraph such as a triangle; or can have a variable size representing some general conceptual pattern such as a star or a loop [102]. For brevity, we focus on fixed-size subgraphs. Formally, a network motif is

DEFINITION 3.1 (NETWORK MOTIF). *Motif M of graph G is a subgraph of G that has multiple isomorphic graphs that are also subgraphs of G. That is, $G_1 \subseteq G, G_2 \subseteq G, \ldots, G_n \subseteq G$ and $G_1, ..., G_n$ are isomorphic to M.*

3.1.1 Motif Frequency

For motif M, its number of appearances in graph G can be denoted as $F_G(M)$. By comparing this frequency with the mean (expected) count in random graphs with the same size as G, we obtain the Z-score of motif frequency

$$Z(M) = \frac{F_G(M) - \mu_R(M)}{\sigma_R(M)}, \tag{1}$$

where $\mu_R(M), \sigma_R(M)$ are the expected mean and standard deviation of frequencies of M in a random graph. This Z-score is an important statistic for measuring the significance of motifs.

3.1.2 Motif Matrix and Motif Cuts

Similar to the adjacency matrix, Benson et al. [15] define *motif adjacency matrix* based on the memberships of edges in the given motifs. Formally, given a specific motif M, the motif adjacency matrix W_M is defined as

$$(W_M)_{ij} = \Big|\{M | i \in M, j \in M\}\Big|, \tag{2}$$

where $(W_M)_{ij}$ is the number of instances of M that contain nodes i and j. When the given motif is an edge (two connected nodes), the motif adjacency matrix is simply the adjacency matrix. Note that when the given motif is not a complete graph, the nodes i and j can belong to the same motif even if they are not connected.

The motif adjacency matrix is also of size $|V| \times |V|$, so most algorithms designed for the adjacency matrix are also suitable for the motif adjacency matrix.

With the motif adjacency matrix in place, a *motif cut* can be defined consequently for motif M and motif adjacency matrix W_M:

$$cut_M(S, \bar{S}) = \sum_{i \in S, j \in \bar{S}} W_{Mij}. \tag{3}$$

Similarly, *motif conductance* is defined as

$$\varphi_M(S) = \frac{\text{cut}_M(S, \bar{S})}{\min(\text{vol}_M(S), \text{vol}_M(\bar{S}))}, \tag{4}$$

where $\text{vol}_M(S)$ denotes the volume (total sum of edge weights) of set S in the motif matrix. If a clustering algorithm minimizes motif conductance, the result leads to preserving the structure of the given motif in the graph while generating a balanced split by the clustering algorithm.

3.1.3 Motif Clustering Coefficient

The *clustering coefficient* is a measure of the degree to which nodes in a graph tend to cluster together [121]. In dyadic graphs, the clustering coefficient can be calculated by the fraction of length-2 paths (wedges) that are involved in triangles. From this perspective, the *global clustering coefficient* can be defined as

$$C = \frac{6|K_3|}{|W|}, \tag{5}$$

where $|K_3|$ is the number of triangles (3-cliques), and $|W|$ is the number of wedges. Each triangle is counted six times since it contains six different wedges, considering the order. The *local clustering coefficient* of node u is defined as

$$C(u) = \frac{2|K_3(u)|}{|W(u)|}, \tag{6}$$

where $C(u)$ is the fraction of triangles that node u belongs to over the number of wedges in which u is the center node.

Based on the above interpretation of the clustering coefficients, Yin et al. [126] introduce a generalized higher-order clustering coefficient for motifs of higher-order cliques. For order $l \geq 2$, clustering coefficient of order l can be defined as

$$C_l = \frac{(l^2 + l)|K_{l+1}|}{|W_l|}, \tag{7}$$

where K and W are higher-order cliques and wedges, and $(l^2 + l)$ is basically $(l + 1)l$, which is the number of wedges that an $(l+1)$-clique closes. Consequently, the local higher-order clustering coefficient is generalized as

$$C_l(u) = \frac{l|K_{l+1}(u)|}{|W_l(u)|}. \tag{8}$$

3.2 Use Cases and Applications

Motifs are utilized across many scientific fields. Here, we survey use cases and applications of motifs.

3.2.1 Capturing Functionalities in Networks

Highly frequent motifs are often related to specific functionalities that networks capture, especially in biology. We provide some examples in biological and brain networks.

Biological networks. In 2002, Shen-Orr et al. [102] distinguished three families of motifs in Escherichia coli (E. coli) directed transcriptional network. These families are closely related to specific functionalities. They are: feedforward loop (a directed acyclic graph), single input module (one to many transactions) and dense overlapping regulons (many to many transactions). Each motif relates to a specific function in the determination of gene expression. Such frequent motifs can be detected using a brute-force approach on some sub-matrix of the adjacency matrix.

The functionality of motifs in biological networks is further validated through a simulation of *spontaneous evolution process* [53]. First, an electronic combinatorial logic circuit is initiated by random wiring. The goal of network evolution is to increase the fraction of correct output under given logical functions. The baseline, a *fixed goal* given by $G_1 = (X \oplus Y)AND(Z \oplus W)$, is very slow to converge with a low rate of reaching the perfect solution. Networks that evolved under fixed goal have less significant motifs and lower modularity (the separability of the design into units that perform independently). Addressing this issue, the authors introduce *modularly varying goals*, where the goals switch between G_1 and $G_2 = (X \oplus Y)OR(Z \oplus W)$ every 20 epochs. Surprisingly, such evolved networks could always find perfect solutions of the current goal (either G_1 and G_2) quickly within a few epochs. Similar findings are discovered in neural networks, which explain adaptiveness and robustness of motifs in real-world biological networks.

Brain Networks. Due to the complexity of brain networks, relationships between subareas of the brain and their functionality are of significant research interest. In brain networks, motifs are often divided into two groups: *functional* and *structural*. Structural motifs are those currently presented in this survey and capture the anatomical building blocks of the brain network, whereas functional motifs capture patterns of elementary processing within such structural motifs. In other words, functional motifs can be considered as all possible subgraphs of the structural motif with the same number of nodes but different edges. For example, a triangle structural motif consists of three different functional motifs (all paths of length two). One hypothesis suggests that the number and variety of *functional motifs* are maximized in the brain to increase effectiveness [108].

Functional motifs vary significantly over time. Duclos et al. [31] investigate motif appearances in the brain network over time. They count all connected subgraphs of size three in directed brain networks, and as a result, show that anesthetic-induced unconsciousness is associated with a topological re-organization of the brain network. Specifically, the frequency of chain-like and loop-like motifs change significantly when people transition from a responsive state to an unresponsive state. Such observations demonstrate links between motifs and functionalities.

In terms of the shape of motifs in brain networks, research has been more interested in specific motifs that are easier to count, such as cliques and *cavities* (enclosed spaces). For instance, all maximal cliques are counted, and their frequency is compared to that of what is expected in some null model. The results indicate the spatial distributions of maximal cliques are more than expected in different brain regions. Similarly, cavities can be studied (ranging from minimum cycles to incomplete cliques). Research shows that, unlike cliques, cavities are less than expected in different areas of the brain [105].

3.2.2 Network Classification

In the seminal work of Milo et al. [76], network motifs are defined as specific fixed subgraphs whose frequencies are higher than what one would expect in random graphs. It turns out motifs found and their frequencies from various types of networks (including food webs, biological networks, electronic circuits, World Wide Web, and the like) can be utilized to classify types of networks. In particular, graphs from the same category exhibit significant overlap in terms of motifs observed and their frequencies, which can be used as features to distinguish graphs. For example, on food webs, a three-node chain is frequently observed. However, this motif is not frequently observed in any other category of networks. The feed-forward loop is popular on most information-processing networks, including brain networks, electronic circuits, and the World Wide Web.

Milo et al. further investigate the number of motifs across various categories [75]. Z-scores (see Equation 1) are calculated for subgraph frequencies and are compared with those expected in random graphs. The results show that the graphs from the same category have highly similar subgraph frequencies. However, some graphs from different categories also show similarities in their frequencies, which captures the intrinsic similarity between graphs that are from similar categories. As a result of this discovery, networks from different categories can be classified into superfamilies using motif frequencies.

3.2.3 Network Models

As motif frequencies are different in real-world graphs compared to what one would expect in random graphs, there is a need for *network models* that can generate realistic random graphs with motif frequencies similar to those of real-world graphs.

Pržulj et al. [91] propose a new series of network models called geometric random graphs, which uniformly generate nodes (i.e., points) in 2D/3D/4D Euclidean space and form links between nodes based on a threshold on their distances. By counting all possible motifs under size five, they found that the random graphs generated by geometric models are more similar in motif counts to the original protein-protein networks than those generated by other network models.

There is also a significant need to develop non-random network models that can generate motifs with similar frequencies to those observed in real-world data. One example is the network model designed by Leskovec et al. [63]. The work examines all triangles in *signed* networks, where edges can have a sign: $+$ or $-$. For instance, a $+$ edge may indicate that two nodes are "friends," and a $-$ edge may indicate that two nodes are "enemies." They discovered that network models that simulate the *balance theory* (colloquially stated as "an enemy of an enemy is my friend") might be insufficient to explain the frequency of triangles appearing in real-world networks. Therefore, they propose another network model for directed links, inspired by *status theory*, where a positive link from node a to b indicates that a has a higher "status" than b. Given a three-node directed cycle with two positive signs, these two theories will predict opposite signs for the third link. Balance theory explains this as three pairwise friends (friend of a friend is a friend), while status theory considers $a \rightarrow b \rightarrow c$ as a pattern of increasing status, so c should link to a with a negative sign (high to low status). In directed graphs, research shows that graphs generated based on status theory could more realistically replicate the frequencies of signed motifs compared to graphs generated based on balance theory.

3.2.4 Clustering

Motifs are closely related to clusters in higher-order networks. Benson et al. [15] develop a framework of network clustering based on higher-order properties. More specifi-

cally, cuts on edges are generalized to cuts on motifs, and the adjacency matrix is generalized to the motif membership matrix. Their results show that such clustering accurately preserves higher-order structures. Two real-world examples are presented. For clustering based on a *bi-fan* motif, the clustering clearly distinguishes between the role of source and sink by assigning them to different clusters in neuronal networks. In the airline network, transportation hubs are clustered together by using a bi-directional 2-path motif.

Building upon the ideas of motif matrix and motif cuts, Yin et al. [127] generalize clustering methods to the motif level. They first propose a motif-based approximate personalized PageRank (MAPPR), which performs an approximation of the Personalized PageRank using the motif matrix. The method can quickly find a cluster that contains a given seed node that has the minimum *motif conductance*. To enhance the performance in case the clustering is performed on the whole graph, they introduce an efficient method to identify good seed nodes to be used as input to MAPPR. The proposed method is validated by performing cluster recovery tasks on synthetic and real-world graphs. Experimental results show that the proposed techniques could preserve higher-order clustering coefficient (as detailed in Section 3.1.3).

Furthermore, as we also showed in Section 3.1.3, some traditional measurements of clusters (or clusterability) are generalized to the motif-level. One important generalized graph measurement was the *clustering coefficient*, which reflects the degree of cohesiveness of communities. Yin et al. [126] introduced higher-order variants of the local and global clustering coefficients. For order-3, the global clustering coefficient is defined as the ratio of cliques to wedges (length-2 paths); the local clustering coefficient is defined as the ratio of cliques that a node involved over wedges. Interested readers can refer to Section 3.1.3 for extensions of clustering coefficients to orders greater than three.

Another important measure for a clustering is its *modularity*. Modularity is a quantitative measure to evaluate the significance of clusters, which is also generalized to the motif level [56], specifically two special motifs – cycles and paths. In the general case, motif modularity is defined as the fraction of motifs laying fully inside the community subtracted by that of expected in the random graphs. Higher-order modularity can distinguish differences in higher-order structures, such as cycles and cliques/hubs and leaves. For example, in a multipartite network roles can be easily distinguished simply by applying higher-order modularity.

3.2.5 Representation Learning

Representation Learning aims at encoding specific network properties into fixed-length vectors. In order to capture higher-order structures, Rossi et al. [94] propose a network embedding method based on motifs. First, they build several weighted motif adjacency matrices based on the nodes' occurrences in specific motifs. Then they define a series of functions over these weighted motif matrices, such as k-step paths, the transition matrix, and various Laplacians. By minimizing the distances between the motif-based matrix formulation and the embedding matrix, each motif matrix learns a local embedding. Finally, they concatenate the local embedding to calculate a global embedding for the network. Experimental results show that the proposed higher-order network embeddings outperform other embedding methods in link prediction tasks.

Another higher-order representation learning method using motifs is *LEMON* [101]. First, *LEMON* converts the graph by adding *supervertices* for motifs (e.g., triangles), and then links the nodes that are involved in such motifs to the corresponding supervertices. The result is a *two-mode* network that captures the memberships of nodes in motifs, where the edges between motif supervertices to regular nodes are defined as structural edges. The embedding vector is learned through a random walk process that captures the similarities of nodes that share similar motif structures. A parameter q controls the traversal probability from a regular node to supervertices. With larger q, any node will become closer to nodes similar in motif properties rather than its structural neighbors. *LEMOM* has been successfully applied in anomaly detection, link prediction, and node classification.

3.2.6 Link Prediction

Motif counts can be used a powerful feature to predict missing links. Abuoda et al. [1] convert the link prediction problem into a classification problem by counting the motifs involved in the link candidates. All possible connecting motifs within size five are enumerated as features. The performance of several classical classifiers shows that larger motifs can lead to higher performance and that a combination of motifs can further improve the results. The work shows that motif-combined feature classification outperforms most state-of-the-art link prediction methods.

4. SIMPLICIAL COMPLEXES

A simplicial complex can be interpreted as another generalization of a graph. In graphs, there are two different types of entities – nodes and edges. But in simplicial complexes, the concepts of nodes and edges are merged into a generalized basic unit, the *simplex*, where 0-simplex represents the single vertex and 1-simplex represents the edge. Furthermore, a simplicial complex can contain any *order* of interactions, such as k-simplices ($k \geq 0$).

The main difference between simplicial complexes and hypergraphs is the requirement of being *inclusive*; that is, a simplicial complex also contains all *faces* (sub-simplices) of its current simplices. For example, if three people A, B, and C belong to the same university, then all pairs: AB, AC, and BC have the same relationship (being part of the university). When modeling higher-order data with simplicial complexes, ensuring the relationship being modeled is inclusive is the first requirement.

4.1 Foundation and Algorithms

In mathematics, a simplicial complex is a set composed of points, line segments, triangles, and their n-dimensional counterparts.

4.1.1 Simplex

A *simplex* is the basic unit of a simplicial complex and is the generalization of the notion of a triangle or a tetrahedron to higher dimensions. More specifically, a *k-simplex* is a k-dimensional polytope that is the convex hull of its $k+1$ vertices.

The convex hull of any nonempty subset of the $k+1$ points that define a k-simplex is called a *face* of the simplex.

Faces are also simplices. Any $k-1$-face of a k-simplex is called a *facet*.

4.1.2 Simplicial Complex

Definition 4.1 (Simplicial Complex). *A simplicial complex $\mathcal{X}$ is a set of simplices that satisfies the following:*

1. *Every face of a simplex from $\mathcal{X}$ is also in $\mathcal{X}$; and*
2. *The non-empty intersection of any two simplices $\sigma_1, \sigma_2 \in \mathcal{X}$ is a face of both σ_1 and σ_2.*

Roughly speaking, simplicial complexes are simplices that are (1) closed under taking faces and (2) have no inner intersections other than faces. A *simplicial k-complex* $\mathcal{X}$ is a simplicial complex where the largest dimension of any simplex in $\mathcal{X}$ equals k.

4.1.3 Homology

First, we define the *orientation* of a simplex. The orientation of a k-simplex is given by an ordering of the vertices $(v_0, v_1, \ldots, v_k)$. There are exactly two orientations – even and odd permutations, and switching any two vertices in the ordering leads to a change of the orientation. For example, in a two-dimensional space, we have clockwise and counterclockwise ordering for a triangle. The orders (v_1, v_2, v_3),(v_2, v_3, v_1),(v_3, v_1, v_2) indicate one orientation, and the orders (v_1, v_3, v_2),(v_3, v_2, v_1),(v_2, v_1, v_3) indicate the opposite one.

Let $\mathcal{X}$ be a simplicial complex. A *simplicial k-chain* is a finite formal sum

$$\sum_{i=1}^{N} c_i \sigma_i, \tag{9}$$

where c_i is an integer and σ_i is an oriented simplex. For each simplex, the sum includes a sign based on the orientation. One way of assigning orientations is to order all vertices of the simplicial complex and give each simplex the orientation corresponding to it. The group of k-chains on $\mathcal{X}$ is written $C_k(\mathcal{X})$, and for simplicity we write C_k. Note that C_k is a vector space with the number of k-simplices as its dimension.

Based on the group of k-chains C_k, we define *boundaries* and *cycles*. First, we define the boundary operator.

Definition 4.2 (Boundary Operator). *Let $\sigma = (v_0, ..., v_k)$ be an oriented k-simplex, viewed as a basis element of C_k. The boundary operator $\partial_k : C_k \to C_{k-1}$ is the homomorphism defined by:*

$$\partial_k(\sigma) = \sum_{i=0}^{k} (-1)^i (v_0, \ldots, \widehat{v_i}, \ldots, v_k),$$

where $(v_0, \ldots, \widehat{v_i}, \ldots, v_k)$ is the i^{th} face of σ, which deletes v_i from σ.

The boundary of each k-simplex is the collection of all its $(k-1)$-faces. In C_k, elements of the subgroup $Z_K := \ker \partial_k$ are referred to as *cycles*, which is the collection of k-simplexes whose boundary is zero. While subgroup $B_K := \operatorname{Im} \partial_{k+1}$ denotes the boundaries, i.e., boundaries of $(k+1)$-simplices. Note that using definition 4.2, it is easy to prove that the boundary of boundaries is empty.

Cycles are essential entities for detecting holes. However, some simplices in Z_K are just boundaries of $(k+1)$-simplices, which are not holes themselves. Hence, we remove the boundaries of the $(k+1)$-simplices from the cycles. This can be defined as the *quotient abelian group*

$$H_k = Z_k / B_k = \ker \partial_k / \operatorname{Im} \partial_{k+1}, \tag{10}$$

where the remaining simplices in H_k represent k-dimensional holes in the complex. H_k is called the *homology group*.

4.1.4 Cohomology and Hodge Laplacian

Remember the group of k-chain C_k is a vector space over $\mathbb{R}$. Hence, it is possible to give an inner product structure to each C_k to make the basis (oriented simplices) orthogonal. We denote this dual space of C_k as C^k [79], called the group of *k-cochains*.

We denote the dual operator of the boundary map ∂_k as δ_k. Operator $\delta_k : C^k \to C^{k+1}$ is called the *co-boundary* operator, which is the adjoint of boundary map ∂_k. Consequently, the *cohomology* group is defined over *cochains*

$$H^k = \ker \delta_k / \operatorname{Im} \delta_{k-1}, \tag{11}$$

which is exactly a dual group of the homology group H_k. Note that the cohomology groups are defined more algebraically with less geometric meaning. The main purpose of introducing the cohomology group is to derive the *Hodge Laplacian* (see Definition 4.3). For a detailed explanation, interested readers can refer to [66].

Given a simplicial complex $\mathcal{X}$, its boundary map ∂_k can be represented as a matrix B_k. B_k has the dimension $n_{k-1} \times n_k$, where n_{k-1} and n_k are the number of $(k-1)$-simplices and k-simplices, respectively. For example, $B_0 = 0$ and B_1 is a matrix of dimension $|V| \times |E|$.

Similarly, the co-boundary map δ_k can also be represented as the adjoint matrix B_k^*. In a finite real space, it is equal to the transpose of B_k, so we can also write it as $B_k^\intercal$.

Definition 4.3 (Hodge Laplacian). *The kth Hodge Laplacian of a simplicial complex $\mathcal{X}$ is defined as*

$$\mathcal{L}_k = B_k^\intercal B_k + B_{k+1} B_{k+1}^\intercal. \tag{12}$$

When $k = 0$, $\mathcal{L}_0 = B_1 B_1^\intercal$ is exactly the Laplacian matrix in dyadic graphs, with dimension $|V| \times |V|$. Matrix $\mathcal{L}_1$ has dimension $|E| \times |E|$, capturing relationships among basic units of edges [97].

4.1.5 Degrees and Random Simplicial Complex

Degree is generalized in simplicial complex as follows [27]:

Definition 4.4 (Degree of a Simplex). *For any simplex $\sigma \in \mathcal{X}$, the degree $k_{d,\lambda}(\sigma)$ is the number of d-dimensional simplices adjacent with σ in λ-faces.*

When we are only interested in the degree of vertices, we let $\lambda = 0$. Then $k_d(v)$ becomes the number of d-simplex incident to v, i.e., those that v belongs to.

As an analog to the *Erdős–Rényi* model [32] in dyadic graphs, the generative model of 2-complexes can be defined as follows:

Definition 4.5 (Random 2-Complex). *The $\mathcal{X}(n, p)$ model of a simplicial complex is defined to have vertex set $[n]$, edge set $\binom{[n]}{2}$, and each of the $\binom{n}{3}$ possible triangle faces is included independently with probability p.*

Note that for a 2-complex both nodes and edges (e.g. a complete graph) have to be specified, and the random process only occurs on random triangles [50]. One can define random simplicial complexes of higher order in similar ways.

4.2 Use Cases and Applications

Applications of the simplicial complex have mainly focused on two directions. One direction is focused on topological properties, where a simplicial complex is used to represent a space. In many fields, the real-world information can be abstracted to pure topology entities through a process called *filtration*, which transforms real distances into topological edges. Such techniques are widely used in sensor coverage problems, biological networks, mobility analysis, robotics, and the like. The second direction is to model real-world interactions of more than three individuals as simplices.

4.2.1 Sensor Coverage

Sensor coverage problem aims at measuring a "coverage" area and detecting locations that are uncovered: also known as *holes*. Ghrist and Muhammad [36] modeled the sensor coverage problem using simplicial complexes. A coordinate-free sensor network is formed by relative distances between any pair of nodes, without any specific coordinates. This is simpler to obtain through the strength of signals sent by the sensors, especially in dynamic systems. Based on simplicial homology theory, coverage holes are what remain after removing boundaries from cycles. The theoretical results are also verified by practical simulations in computational homology software.

The sensor cover can be further linked to the homology of the diagram of complexes [28]. In particular, the sensor cover can be defined as a collection of discs of radius r_c, and the radius of strong and weak signals of pairwise distances can be represented as r_s and r_w. *Rips complex* is defined as a simplicial complex whose simplices are tuples of nodes whose pairwise Euclidean distances are within a certain threshold. Each node can detect the existence of the boundary of the domain within another radius r_f. By forming the simplicial complexes of all these graphs, one can derive the sensor coverage.

Under similar settings with previous studies, Tahbaz-Salehi and Jadbabaie [109] present a distributed algorithm for coverage verification without any metric information. The goal of coverage verification is basically three aspects – detecting coverage holes, calculating their locations, and detecting redundancies in the network. The main novel contribution of this work is to solve the homology problem through a linear programming relaxation.

4.2.2 Disease/Abnormality Detection

Point cloud is one of the classic data formats often used in biology, where points are substances such as proteins. Similar to the sensor coverage problem, a simplicial complex can be constructed over a point cloud through the *filtration* process [81]. Points agglomerate together and become simplicies when their distances fall under a specific threshold, specified by some distance function. As a result, a simplicial complex can be used for preprocessing to enhance the clustering performance [82]. One notable usage is to identify subtypes of breast cancer. Simplicial complexes can also help distinguish between recurrent and non-recurrent subtypes [29; 6].

In brain networks, a new topology called *homological scaffold* can be defined to represent low-connection areas in the network [90]. First, a brain network can be seen as a weighted network, where larger weights indicate longer distances. Then a filtration process is applied to generate sparse structures (larger weights), followed by the detection of a homology group. The remaining structure in the homology group contains cycles with larger distances, which captures areas in the network that exhibit extremely low connections.

In neuroscience, for amplifying the differences in network topologies, a novel matrix signature is proposed to facilitate forming the homology groups [38; 37]. Instead of the absolute distances, the orders of distance are used. For example, if the distance of v_0 and v_1 is the minimum of all pairwise distances, then the entry of A_{01} will be encoded as 0. Such a non-linear transformation obscures the distances but focuses more on the intrinsic structure of the network. The order matrix captures a more robust relationship with the topological structure, for example, the number of non-contractible cycles. Experiments on pyramidal neurons in the rat hippocampus show that the proposed signature is capable of detecting geometric organization.

4.2.3 Mobility Analysis

To study mobility, topological signatures have been proposed that represent trajectories as points in k-dimensional space, where k is the number of obstacles [35]. The goal of this mapping is to characterize the differences in traces when passing by obstacles. First, obstacles are represented as simplicial complexes, and any motion toward faces can be recorded by sensors. Based on homology, these faces (edges) are encoded as real numbers. These values are added to the entries of the related obstacles as trajectories records. Then, trajectory traces can be distinguished by this signature. For example, one coming across an obstacle from the left is encoded as 1, while as -1 if coming from the right; if one loops clockwise around an obstacle, we can encode that as 2, and -2 for counterclockwise loops.

4.2.4 Network Modeling

The configuration model is a method for generating random networks from a given degree sequence. For a simplicial complex, the configuration model is also generalized along with the *canonical ensemble* [27]. In short, the canonical ensemble aims to derive the probability of the simplicial complex that maximizes the entropy defined by it. The configuration model is the uniform distribution of all possible simplicial complexes with the same degree sequence.

Based on such generalizations, Young et al. [128] further develop efficient sampling algorithms for the simplicial complex configuration model. First, they elaborate the numerical constraint of the configuration model by switching the simplicial complex to its equivalent *graphical* representation. That is, to introduce extra nodes to represent adjacent relationships between nodes and simplicies. In this way, the simplicial complex is transformed into a dyadic bipartite graph, which can yield a solution [33].

The *social contagion* can be modeled as a propagation network, where people get infected through social interactions (edges). In terms of a simplicial complex, a contagion model could also consider an infection being caused by a

group, called *Simplicial Contagion Model* [48]. Similar to the dyadic contagion model, any pairwise interaction can lead to an infection with a uniform probability (β_1). In addition, higher-order interactions have unique contagion probabilities if there are multiple infecteds involved. For example, in a simplicial complex, if the other two nodes are infected, the candidate will have a probability of (β_Δ) being infected. The behavior of the infection pattern is discussed by simulating the process over both real-world and synthetic graphs. The *Simplicial Contagion Model* is a more flexible fit for more complex diseases with varying infection probabilities of different orders.

In quantum physics, Bianconi and Rahmede [16] propose a model of emergent geometry that is based on a growing simplicial complex. The model is simple as it just keeps including simplicies with fixed dimension d and gluing them to the existing simplicial complex on one of its $d-1$ faces. Many advantages of such a model are validated and discussed under certain settings, including scale-free degree distribution, small-world properties, and modular structure.

4.2.5 Network Analysis Tools

Instead of studying motifs in dyadic networks, Benson et al. [12] directly collect higher-order relationships in the real world, such as co-authorships, event participation, drug instances, among other similar interactions. Such coappearances are modeled as simplicial closures (timestamped vertex sets). For example, a closed triangle indicates relationships among three nodes, while an open triangle just represents pairwise relationships between any two nodes. This representation enriches the network information and can be used in dynamic graph-evolving models or link predictions. In the link prediction task, the goal is to predict whether the open triangles will become close in the future. Results show that even simple local features such as the mean of weights on three edges perform pretty well and are comparable with state-of-the-art methods.

Based on the 1st normalized Hodge Laplacian, Schaub et al. [97] discuss random walks on basic units of edges in a simplicial complex. This work enriches the traditional field of network analysis, which is mostly node-based. Two applications are performed to verify the usage. One is representation learning of edge-flows and trajectory data, as a higher-order generalization of diffusion maps and Laplacian eigenmaps. Another is the edge-based generalization of PageRank [39], which focuses on the importance of edges rather than nodes.

Regarding clustering, Osting et al. [85] applied a sparsification process on a simplicial complex, which downgrades the maximum dimension under a given threshold. It is proved that such a sparsification preserves the *up Laplacian*. The authors also generalize *Cheeger inequality* to a simplicial complex. The preservation of the spectrum is verified through experiments, and spectral clustering is performed as one application.

Advanced network representation learning methods have also been extended to the simplicial complex. Hajij et al. [43] use the autoencoder to perform simplex-level embedding. The encode function ($X \to \mathbb{R}^d$) maps each simplex to a fixed vector. The decode function ($\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$) maps each pair of simplices to a similarity score that reflects the relationship between the two simplices. An example of a user-defined similarity is the simplex-level adjacency matrix. Finally, the representation of the whole simplicial complex is obtained by a weighted sum of the simplex-level representations.

5. HYPERGRAPHS

Another natural generalization of a graph is a *hypergraph*, which extends the edge space from $|V|^2$ to $|V|^{|V|}$. In other words, an edge of a hypergraph can be any subset of the vertices. The main difference with the simplicial complex is that any subset of an edge can exist independently from others.

Though easy to understand, the extremely sparse and high-dimensional edge space causes difficulty in computations. So, relatively more studies on hypergraphs have focused on the theoretical aspects rather than applications. For a more comprehensive review of concepts and measurements in hypergraphs, we refer interested readers to the survey by Lee et al. [59].

5.1 Foundation and Algorithms

Many definitions and properties of hypergraphs are directly inherited from graphs, such as *node degrees*, *hyperedge weights*, *simple/multi* hypergraphs, hypergraph *isomorphism*, and so on. Here, for brevity, we mainly focus on the definitions unique to hypergraphs.

5.1.1 Definition and Basics

We use $\mathcal{H} = (V, \mathcal{E})$ to distinguish a hypergraph from a graph, where only vertex set V remains the same. The edge set $\mathcal{E} = \{e | e \subseteq V\}$ is the set of subsets of V. We define *size* of edge $|e|$ as the number of nodes that belong to edge e.

We can have special kinds of hypergraphs:

- d-regular: each vertex has degree d;
- k-uniform: each edge has size k;
- k-partite: vertices belong to one of k different classes, and each edge has exactly one node from each class.

A hypergraph can be always represented by a bipartite graph of vertices and edges. The *biadjacency matrix* of this bipartite graph is a $|V| \times |E|$ matrix, which is also called the *incidence matrix* of the hypergraph.

5.1.2 Tensor Representation

The natural generalization of the adjacency matrix for the hypergraph is a *tensor*, often denoted by $\mathbf{T}$. For example, a 3-uniform hypergraph (or the subset of order-3 edges) can be represented as an order-3 tensor $\mathbf{T} \in \mathbb{R}^{|V| \times |V| \times |V|}$, i.e., the entry $(i, j, k) = 1$ when $(v_i, v_j, v_k) \in \mathcal{E}$. If the hypergraph is undirected, the corresponding tensor is *symmetric*, where its value at any permutation of (i, j, k) remains the same. A *simple tensor* can be written as the outer product of the vectors. The *rank* of a tensor is the minimum number of simple tensors whose linear combination equals that tensor.

(Tensor Decomposition) Due to tensor dimensionality, it is expensive and inconvenient to perform calculations directly on it. Decomposition techniques are widely used to decrease the dimension and preserve the graph characteristics. Here, we introduce two popular decomposition methods – CP decomposition and Tucker decomposition. For a detailed reference on the decomposition techniques of the tensor, interested readers can refer to [104].

CP (CANDECOMP/PARAFAC) Decomposition [22; 46] is also called *tensor rank decomposition* or *Canonical Polyadic Decomposition (CPD)*. The CP decomposition factorizes a tensor into a sum of component vectors. For example, a tensor $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$ can be decomposed as

$$\mathbf{T} \approx \sum_{i=1}^{R} \mathbf{a_i} \otimes \mathbf{b_i} \otimes \mathbf{c_i}, \tag{13}$$

where R is an integer and $\mathbf{a_i} \in \mathbb{R}^I$, $\mathbf{b_i} \in \mathbb{R}^J$, $\mathbf{c_i} \in \mathbb{R}^K$, and $\otimes$ denotes the outer product sign. This decomposition is often solved by some minimization algorithm.

Tucker Decomposition [114] is the generalization of *Singular Value Decomposition (SVD)* for the tensors. Tucker decomposition of a tensor $\mathbf{T} \in \mathbb{R}^{I \times J \times K}$ is represented as

$$\mathbf{T} \approx \mathcal{G} \times \mathbf{A} \times \mathbf{B} \times \mathbf{C} = [\![\mathcal{G}; A, B, C]\!], \tag{14}$$

where $\mathbf{A} \in \mathbb{R}^{I \times P}$, $\mathbf{B} \in \mathbb{R}^{J \times Q}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$. Here, tensor $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$ is called the *core tensor*.

5.1.3 Hypergraph Cuts

In dyadic graphs, a cut is defined as partitioning the vertices into two disjoint subsets, where the cut edge connects two nodes, one from each subset. However, for a hyperedge, there is no such fair split where we cannot assign some nodes to one side and the rest to the other side. Among various cut functions, there is one that might be more reasonable to minimize when solving cut-based hypergraph problems, called *all-or-nothing* [118].

Specifically, a hyperedge is in the middle of the cut when it is assigned to both sides of the cut. One can present the set of *cut hyperedges* by

$$\partial S = \{e \in \mathcal{E} : e \cap S \neq \emptyset \text{ and } e \cap \bar{S} \neq \emptyset\}. \tag{15}$$

The cut function is

$$\textbf{all-or-nothing}(s) = \sum_{e \in \partial S} w_e, \tag{16}$$

where w_e is the edge weight in the case of weighted hypergraphs.

5.1.4 Random Walks and Laplacian

In dyadic graphs, a sequence of vertices is sufficient to define a walk as there is only one way to traverse from one node to another in one step. However, due to the flexibility of hypergraphs, one has to specify the order of both edges and walks.

DEFINITION 5.1 (s-WALK). *Let $\mathcal{H}$ be an r-uniform hypergraph, for $1 \leq s \leq r-1$, an s-walk of length k is defined as a sequence of vertices*

$$v_1, v_2, ..., v_j, ..., v_{(r-s)(k-1)+r}$$

together with a sequence of edges $F_1, F_2, ..., F_k$ such that

$$F_i = \{v_{(r-s)(i-1)+1}, v_{(r-s)(i-1)+2}, ..., v_{(r-s)(i-1)+r}\}.$$

Basically, any two adjacent edges of an s-walk have exactly s vertices in their intersection. For the vertex set V, let $V^{\underline{s}}$ be the set of all ordered s-tuples consisting of s distinct elements in V. For example when $s = 2$,

$$V^{\underline{2}} = \{(v_1, v_2), (v_1, v_3), ..., (v_2, v_1), (v_2, v_3), ...\}.$$

To compute the Laplacian, we consider the following two cases [70]:

(1) In the case of $1 \leq s \leq r/2$, for any F_i, there will not be any intersection with F_{i+2} or F_{i-2}. So, the s-walk can be interpreted as the walk on a weighted dyadic graph. We define a weighted undirected graph $G^{(s)}$ over $V^{\underline{s}}$ as follows. Let the weight $w(x, y) = |\{F \in \mathcal{E} : [x] \cup [y] \subseteq F\}|$. Here, $[x] \cup [y]$ is the disjoint union of $[x]$ and $[y]$, and x, y are vertices of s-tuple of the original vertices. Then, we define the *s-th Laplacian* $\mathcal{L}^{(s)}$ of hypergraph $\mathcal{H}$ to be the Laplacian of graph $G^{(s)}$.

(2) Another case is $r/2 < s \leq r-1$, where F_i also intersects with F_{i+2} or F_{i-2} (if it exists). We define a directed graph $D^{(s)}$ over the vertex set $V^{\underline{s}}$ as follows. With x, y are still the s-tuples of the original vertices, let (x, y) be a directed edge if $x_{r-s+j} = y_j$ for $1 \leq j \leq 2s - r$ and $[x] \cup [y]$ is an edge of $\mathcal{H}$. Then, we define the *s-th Laplacian* $\mathcal{L}^{(s)}$ of hypergraph $\mathcal{H}$ to be the Laplacian of Eulerian directed graph $D^{(s)}$.

5.1.5 Downgrading a Hypergraph to a Dyadic Graph

To downgrade hypergraphs to dyadic graphs, one of the most straightforward ways is to perform *(clique) expansion*. For each hyperedge e, we enumerate all its size-two subedges to form a dyadic graph. Depending on the application, one can either inherit weights from the original hypergraph for these new edges or only keep the structural information [113]. While it is often much more convenient to perform dyadic graph algorithms, the expanded new graph indeed loses higher-order information.

A more general expansion is the *Multi-Level decomposition* [30]. In Multi-Level decomposition in addition to enumerating size-two subedges to form a dyadic graph, we construct hypernodes based on the coexistences within the original hyperedges. For example, assume there is a hyperedge of size $|e|$. At each layer k, we generate all possible $\binom{|e|}{k}$ hypernodes of size k to form a clique. By selecting k ranging from 2 to the maximum order of the hypergraph, we construct a corresponding dyadic graph for each k. The most appealing feature of such a decomposition is that one can easily reconstruct the hypergraph from its expansions across all layers.

5.2 Use Cases and Applications

Here, we summarize studies utilizing hypergraphs. One branch focuses on developing tools for network analysis, mostly extending graph theories to hypergraph-level. Another branch applies hypergraph to model higher-order interactions for network analysis.

5.2.1 Network Measurements

Many network properties and measurements are generalized from dyadic graphs to hypergraphs. As an analogy to walks on dyadic graphs, s-walk is proposed and applied to hypergraphs [3]. As mentioned in Definition 5.1, an s-walk is a series of hyperedges where the intersection nodes between any adjacent hyperedges have size greater than s. More specifically, a larger s indicates a component filled by denser overlapped hyperedges. Consequently, s-connected components and s-distance are also defined based on s-walks, which form a series of hypergraph analysis tools. Such measurements can help distinguish real-world networks from random hypergraphs generated by network models.

Centrality measures assess how important a node is in terms of its position and how it connects to other nodes in the graph. Three eigenvector centralities for uniform hypergraphs are defined by Benson [11]. Similar to dyadic networks, the centrality of a node in hypergraph can be influenced by centralities of all its neighbors. For each specific hyperedge to which it belongs, there could be a weighting function based on the centrality scores of its neighbors on that edge. First version is called *Clique motif Eigenvector Centrality*, where it simply refers to the total sum of centrality of all neighbors; Second is *Z-Eigenvector Centrality*, which multiplies the neighbors' centralities on each hyperedge and then sums them up; Third is *H-Eigenvector Centrality*, which is the square root of the *Z-Eigenvector Centrality*. Experimental results show that none of these three centralities is consistently superior to others. So, one has to consider a specific objective to make an informed selection.

As motifs are significant subgraph patterns in dyadic graphs, *higher-order motifs* are defined in a similar way [69]. Given a set of nodes of size k, a higher-order motif is formed by a collection of hyperedges consisting of only these k nodes. As k increases, the motif variations can exponentially increase, which makes it impossible to enumerate and detect all motifs in the hypergraph. Addressing this problem, Lee et al. [61] propose *hypergraph motifs*, a special kind of higher-order motifs. Hypergraph motifs have a fixed structure that consists of three connected hyperedges. Based on overlapping nodes in hyperedges, all nodes are classified into one of the seven possible areas in a Venn diagram. Such a structure significantly simplifies motif detection and isomorphism checks, which are required to count motifs.

5.2.2 Generative Models

Graph generation algorithms aim to generate realistic graphs similar to those observed in the real world. Chodrow generalizes two variants (*vertex-labeled* and *stub-labeled*) of the *configuration model*, a well-known graph generation algorithm, to hypergraphs [24]. Configuration model in dyadic graphs requires the knowledge of the degree sequence. In hypergraphs, in addition to degree sequences, dimension sequence (sizes of edges) is also needed to generate a random graph. The vertex-labeled hypergraph configuration model is just a uniform distribution over the space of hypergraphs defined by degree and dimension. The stub-labeled hypergraph configuration model simply copies nodes as many times as their degrees and places them into a multiset. The algorithm then uniformly samples hyperedges based on the dimension sequence, where each node can only appear once in a specific hyperedge.

Lee et al. [60] extend the *Chung-Lu* model [26] to hypergraphs (*HyperCL*) by ensuring to preserve the distribution from the given degree sequence and the edge-size sequence as input. However, real-world hypergraphs exhibit stronger communities than random graphs. Addressing this issue, the authors further propose *HyperLap*, a multilevel HyperCL that introduces a group parameter L at each level, which aims to help reconstruct community patterns of real-world hypergraphs. New hyperedges generated within each group are expected to have high number of overlapping nodes, especially when the group is small.

Furthermore, a later study extends the *degree-corrected stochastic blockmodel* [51], which is a generative model of graphs with both community structure and degree sequences, to hypergraphs [25]. For hyperedge candidates, the authors introduce an affinity function to compute the wiring possibility based on the group memberships of their nodes. Basically, more nodes in the same group have a higher probability of forming hyperedges. Three estimates —the affinity function, node labels, and node degrees, are alternatively learned by optimizing a likelihood function. To solve this objective, the authors propose an 'All-or-Nothing' (AON) *Louvain*-type algorithm [18] under the assumption that hyperedges are expected to lie fully within the cluster. Experimental results on both synthetic and empirical data validate the efficiency and accuracy of the framework.

5.2.3 Hypergraph Partitioning and Clustering

Hypergraph partitioning methods are generalized from classical graph cut problems. Veldt et al. [118] propose a comprehensive set of steps for solving hypergraph $s-t$ cuts problem. The first step is to select a splitting function, which maps the hyperedge that is going to be cut to a real number penalty (this has to be defined specifically). The authors specify a property for the splitting functions called *cardinality-based*, where the penalty only correlates with the sizes of split clusters. The hypergraph $s-t$ cuts problem is defined as minimizing the total splitting penalty for all crossing hyperedges. Various splitting functions are analyzed and tested over real-world datasets. Based on the results, a new clustering framework for hypergraphs is proposed [117], which minimizes the localized *ratio cut* objective. The algorithm requires a set of input nodes and returns a well-connected cluster that highly overlaps with the inputs. The running time of this algorithm only depends on the size of input set, and guarantees cuts or conductance under a specific bound.

The hypergraph cut problem can also be solved with tensor representations. By extending matrix-based methods, a tensor spectral clustering method is developed for partitioning higher-order networks [14]. For example, an order-3 undirected network can be represented as an order-3 symmetric tensor. As an analog to random walk on dyadic networks, a second-order Markov process is applied to express state changes on order-3 networks. Clustering of higher-order networks can be achieved by recursively partitioning the graph by minimizing *sweep cuts*. Such a clustering method preserves higher-order structures rather than just edges, as shown through experiments on both synthetic and real-world networks.

The clustering method can be further extended when introducing additional information. As for labeled networks, Amburg et al. [4] propose a novel hypergraph clustering framework based on given edge labels. The objective simultaneously minimizes (1) edges across clusters and (2) edges that do not belong to the assigned cluster. These two requirements can be combined by simply counting the number of nodes whose labels are inconsistent with their connected edges. In the case of two categories, this problem reduces to an s-t cut problem by forming a dyadic graph and adding a terminal node to it. In the case of more than two categories, multiple approximation algorithms for such an NP-hard problem are developed, such as an LP relaxation and multiway cuts. Experimental results on synthetic and real-world graphs show that the proposed method outperforms baselines including *Majority Vote*, *Chromatic Balls* and *Lazy Chromatic Balls*.

Table 1: Tools for Discovering Motifs

Package Name	Year	Description	Official Link (if exist)
MFinder [54]	2005	motif detection, enumeration/edge sampling	https://www.weizmann.ac.il/mcb/UriAlon/download/ParTI
MAVisto [100]	2005	motif detection, enumeration	https://kim25.wwwdns.kim.uni-konstanz.de/vanted/addons/mavisto/
FANMOD [122]	2005	motif detection, enumeration/node sampling	https://github.com/gabbage/fanmod-cmd (unofficial)
Grochow–Kellis [42]	2007	motif detection, mapping	https://github.com/jptboy/CSCI3104_GC2 (unofficial)
MODA [83]	2009	motif detection, mapping/sampling, undirected only	https://github.com/smbadiwe/ParaMODA (unofficial)
Kavosh [52]	2009	motif detection, enumeration	https://github.com/shmohammadi86/Kavosh
G-Tries [92]	2010	motif detection, enumeration/mapping, undirected only	https://www.dcc.fc.up.pt/gtries/
TemporalMotif [87]	2016	temporal motif count	http://snap.stanford.edu/temporal-motifs/
MODET [88]	2019	motif detection, mapping, undirected only	https://github.com/sabyasachipatra/modet

Table 2: Tools for Learning Simplicial Complexes

Package Name	Environment	Description	Official Link
Simplicial	Python	topology, homology, filtrations	https://simplicial.readthedocs.io/en/latest/
Javaplex [111]	Matlab/Java	persistent homology, filtrations	https://github.com/appliedtopology/javaplex
Ripser [9]	C++	persistent homology, Vietoris–Rips filtrations	https://github.com/Ripser/ripser
simplextree	R	topology	https://github.com/peekxc/simplextree
Simplicial.jl	Julia	simplicial complexes, directed complexes	https://github.com/nebneuron/Simplicial.jl
simplicial-complex	JavaScript	structural and topological operations	https://www.npmjs.com/package/simplicial-complex
Dionysus 2	C++	persistent homology	https://mrzv.org/software/dionysus2/
DIPHA	C++	distributed, persistent homology	https://github.com/DIPHA/dipha
Perseus [78]	C++	persistent homology	https://people.maths.ox.ac.uk/nanda/perseus/
Moise	Maple	homology groups	https://www.math.drexel.edu/~ahicks/Moise/
TopoEmbedX [44]	Python	representation learning	https://github.com/pyt-team/TopoEmbedX

As discussed in Section 5.1.5, downgrading hypergraph to dyadic is also an effective way to utilize existing algorithms. Liu et al. [67] propose a hypergraph clustering method, called *Local Hypergraph Quadratic Diffusions (LHQD)*. The first step of LHQD reduces the hypergraph to a directed graph that preserves the conductance property of the original graph. The equality of conductance is achieved by introducing auxiliary nodes for each node. The second step of LHQD creates a source and a sink node in the directed graph, whose weights to the auxiliary nodes are equal to their degrees. Such a conceptual transformation ensures that the objective function becomes the same as the original problem. The performance is validated by performing clustering on two real-world networks.

5.2.4 *Modeling Higher-Order Interactions*

Many real-world interactions can be modeled directly as hypergraphs. The entities often have different types, but in terms of hypergraphs, research rarely emphasizes on node heterogeneity. Tan et al. [110] model the music recommendation problem as a hypergraph ranking problem. At the beginning, different objects (users, groups, tags, tracks, albums, and artists) are represented as nodes and pairwise relationships among them are identified. Hyperedges are created by combining edges based on the intrinsic connections among them, e.g. tracks in the same album. Given a query (set of nodes), the recommendation functions based on rank scores of all other nodes on the hypergraph. This scoring process is trained by the ground truth of node labels under the smoothing constraint based on which close nodes should have similar scores.

Similar to music recommender systems, the image retrieval problem can be modeled as the hypergraph ranking problem, where images are nodes that are assigned to hyperedges based on similarities. Liu et al. [68] applied a *soft hypergraph* model in which the entries on the incidence matrix are calculated using some similarity functions instead of arbitrary values. Hyperedges are created by selecting any node as centroid and adding its k nearest neighbors. When an image query comes, the recommender system solves a linear system based on a cost function capturing hypergraph partitioning, which ensures vertices sharing many incidental hyperedges obtain similar labels.

Rather than built on the basis of similarities, hypergraphs are also directly used to model data in biology. Patro and Kingsford [89] model *network history inference* using a hypergraph structure. Generally speaking, *network history inference* is to find a small set of tuples that record the historical interactions between leaves on the protein network. The mapping of different states is represented as a hypergraph, where current state and correlated historical states are connected by order-3 hyperedges. The problem is solved by minimizing total cost over the network. Such a model can be applied to reconstruct the ancestral networks or to predict missing links.

5.2.5 *Hypergraph Neural Networks*

Graph neural networks have been generalized beyond pairwise interactions modeled as hypergraphs. Hypergraphs can be applied to either (1) model higher-order graph data as input matrices, such as the incidence matrix of hypergraphs; or to (2) build multilayer neural network structures but using higher-order forward- and back-propagation instead. Many state-of-the-art models, especially for graph neural networks, are extended to hypergraphs. Examples include hyper-models such as HGNN [34], HGAT [7; 120], MHCN [129], and HGCN [124; 119]. For a comprehensive survey on graph neural networks, readers are referred to the survey by Thomas et al. [112].

6. DATASETS AND TOOLS

We summarize some available datasets and tools for studying higher-order networks. As some of these official links might disappear in the future, we provide a comprehensive

Table 3: Tools for Learning Hypergraphs

Package Name	Environment	Description	Official Link
HyperG	R	Hypergraph Modeling	https://cran.r-project.org/web/packages/HyperG/
HyperNetX [49]	Python	Hypergraph Modeling, Visualization	https://github.com/pnnl/HyperNetX
GraphML [20]	XML	File Format	http://graphml.graphdrawing.org/index.html
hypergraph	R	Hypergraph Modeling	https://bioconductor.org/packages/3.15/bioc/html/hypergraph.html
SimpleHypergraphs.jl [106]	Julia	Hypergraph Modeling, Visualization	https://github.com/pszufe/SimpleHypergraphs.jl
halp	Python	Hypergraph Modeling, Algorithms	https://murali-group.github.io/halp/
kahypar [99]	Python	Hypergraph Partitioning	https://pypi.org/project/kahypar/
Tensorly [57]	Python	Tensor Learning	http://tensorly.org/stable/index.html
Tensors.jl [21]	Julia	Tensor Learning	https://juliahub.com/ui/Packages/Tensors/F7rKl/1.11.0
rTensor [64]	R	Tensor Learning	https://cran.r-project.org/web/packages/rTensor

Table 4: Applications of Higher-Order Networks

	Network Motifs	Simplicial Complexes	Hypergraphs
Statistical Significance	[102] [53] [108] [31] [105]		
Graph Classification	[76] [75]		
Network Modeling	[91] [63]	[27] [128] [48] [16]	[24] [60] [25]
Clustering	[15] [127] [126] [15] [12]	[85]	[118] [117] [14] [4] [67]
Representation Learning	[94] [101]	[43]	[116] [40]
Link Prediction	[1]	[12]	[89] [116]
Persistent Homology		[36] [28] [109] [81] [82] [29] [6] [90] [38] [37] [35]	
Analysis Tools and Measurements	[126] [15]	[97]	[3] [11] [69] [61]
Recommender System			[110] [68]
Neural Networks			[34] [7] [120] [129] [124] [119]

backup of all tools and datasets in our own repository.[1]

6.1 Higher-Order Network Tools

For most network motif based studies, the first step is to find motifs. In Table 1, we list some scalable algorithms for enumerating or counting motifs. Among these methods, *enumeration* indicates an exhaustive search through the whole graph. *Sampling* indicates that the method calculates an estimated frequency of a given motif by sampling the node/edge and exploring its neighborhood. The *mapping* strategy is a reverse process of enumeration, which maps the given motif onto the whole network.

Table 2 collects packages and software for studying simplicial complexes. Some are tagged as 'topology', which are comprehensive packages that build the data structure from the lower level information. Some are software that are easy-to-use for most popular applications such as persistent homology and filtrations.

Table 3 collects packages for modeling hypergraphs and tensors. Most packages provide a data structure and implement the most basic algorithms using it; some packages support network visualization.

6.2 Higher-Order Datasets

We summarize some dataset resources with higher-order interactions:

- ARB Data[2]: A dataset repository (19 datasets–4 with millions of nodes, 10 with thousands of nodes, and 5 with hundreds of nodes) collected by Austin R. Benson. Most are higher-order networks from various fields, including temporal and labeled hypergraphs.
- LINQS[3]: A collection of 11 relational datasets (1 with a million nodes, others are thousands of nodes or less). Many of them are collaboration networks, which naturally include higher-order interactions.
- Twitter Data[4]: Tweets can be modeled as higher-order networks by taking hashtags as nodes and co-appearances as edges (require preprocessing). Besides these, one can search twitter datasets online for any specific interest or collect data using APIs.
- Temporal Co-authorship[5]: Three large-scale (sizes: 27 million / 13 million / 41 thousand nodes) hypergraph datasets in both static and temporal forms [55].

6.3 Expected Time Complexities

While time complexity of exploring higher-order networks can vary across topologies and algorithms, some time complexities are typically expected for some basic higher-order algorithms. Here, we briefly list some expected time complexities for analyzing higher-order networks.

Motif counting: The time complexity of counting motifs depends highly on the structural complexity of given motifs as the essential algorithm for finding motifs involves checking for subgraph isomorphism, which is known to be NP-complete. Most fast motif-finding algorithms focus on size 3 or 4 motifs. For example, counting motifs of size 3 (triangles) can be solved in $O(|E|d_{max})$ [2] and counting motifs of size 4 can be solved in $O(|E|d_{max} + |E|^2)$ [72], where d_{max} is the maximum degree in the graph.

Homology groups: The time complexity of computing homology groups of the simiplical complex is $O(n^\omega)$, where n is the number of simplices and the exponent $\omega \leq 2.4$ [77]. Such an acceptable and stable complexity facilitates the wide usage of homology methods.

[1] https://github.com/haotian-syr/HON-tools
[2] https://www.cs.cornell.edu/~arb/data/
[3] https://linqs.soe.ucsc.edu/
[4] https://data.world/datasets/twitter
[5] https://github.com/kswoo97/pcl

7. CONCLUSIONS AND FUTURE DIRECTIONS

We survey essential algorithms and applications in the literature of higher-order network modeling and analysis. In Table 4, we summarize the applications collected in this survey. Due to the scope of this survey, we have highlighted representative studies from each field. To explore each area comprehensively, we have directed readers to other surveys focusing on each domain.

However, research on higher-order networks has significant future potential. Here, we list some open problems or under-explored research directions:

7.1 Data Source and Modeling

Unlike dyadic graphs, not many repositories of higher-order network data are available. Building tools to collect, store, and model higher-order data is of significant interest for various academic use cases. Below, we list some essential, yet under developed, tools for studying higher-order data.

Recovering Higher-order Data in Dyadic Graphs: Most existing network data is collected in dyadic form, which has already lost higher-order interactions. In motif-based studies, one cannot distinguish whether a specific motif is indeed a higher-order interaction or formed by a combination of dyadic interactions. It is therefore a challenge worth addressing to build tools that can distinguish higher-order interaction or that can rebuild higher-order networks from a dyadic graph.

Dynamic Higher-Order Graph: Most higher-order interactions are associated with time, such as protein interactions, hashtags, and group chats. Rather than a snapshot analysis of a network during some small interval, there is a demand for tools that can analyze the whole or partial network structure of a temporal higher-order network. Such tools enable real-time fast algorithms for various tasks including link prediction, community detection, anomaly detection, and the like.

Matrix/Tensor Representation: Topologies representing higher-order interactions are always associated with matrix or tensor representations, such as motif matrix [15], tensors [14] and incidence matrix. However, due to complexity and lack of mathematical support, algorithms on these matrix representations are not explored as extensively as matrix-based methods for dyadic graphs.

Interpretability and Causality: Most higher-order networks studies develop algorithms and applications that explore collected higher-order data. Current research rarely aims to interpret findings in higher-order networks or understand why some high-order interactions or patterns exist. As mentioned, some patterns might reflect important real-world information beyond network structures, such as the small functional unit in brain networks [108]. Interpretability is especially crucial as more black-box techniques (e.g., deep neural networks) or embedding methods are designed for higher-order networks.

7.2 Machine Learning Applications

Many real-world applications have focused on higher-order graphs. Below, we list three general application domains for higher-order networks that have a significant potential for future research.

Representation Learning: Representation learning is a powerful tool for transforming high-dimensional data into fixed-size vectors and has been extremely successful for downstream machine learning tasks, such as node classification, link prediction, among others. However, not many representation learning methods are introduced for higher-order networks. Hence, there is a significant demand for representation learning methods that can embed higher-order graphs both at the node-level [116; 40] and the graph-level.

Recommender Systems: One of the most direct uses of higher-order networks is in recommender systems. For example, nodes involved in same hyperedge can share some similarities. As discussed in Section 5.2.4, some applications such as music recommendation [110] and image retrieval [68] are developed. However, for individual recommendations, the advantage of higher-order graphs over simple or heterogeneous graphs needs to be further studied. Similarly, group recommendation [5] is another direction that has the potential to be studied.

Graph Neural Networks: Similar to modeling real-world interactions, the structure of neural networks can be designed to accept higher-order interactions as input when necessary. Due to insufficient studies on hypergraphs, there is only limited work that directly utilizes hypergraphs in neural networks. One main challenge is to extend the adjacency matrix, where some studies have considered the incidence matrix as a solution to this challenge [34].

8. REFERENCES

[1] G. AbuOda, G. D. F. Morales, and A. Aboulnaga. Link prediction via higher-order motif features. *CoRR*, abs/1902.06679, 2019.

[2] N. K. Ahmed, J. Neville, R. A. Rossi, and N. Duffield. Efficient graphlet counting for large networks. In *2015 IEEE International Conference on Data Mining*, pages 1–10, 2015.

[3] S. G. Aksoy, C. A. Joslyn, C. O. Marrero, B. Praggastis, and E. Purvine. Hypernetwork science via high-order hypergraph walks. *EPJ Data Sci.*, 9(1):16, 2020.

[4] I. Amburg, N. Veldt, and A. Benson. *Clustering in Graphs and Hypergraphs with Categorical Edge Labels*, page 706–717. Association for Computing Machinery, New York, NY, USA, 2020.

[5] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu. Group recommendation: semantics and efficiency. *Proc. VLDB Endow.*, 2(1):754–765, aug 2009.

[6] J. Arsuaga, N. A. Baas, D. DeWoskin, H. Mizuno, A. Pankov, and C. Park. Topological analysis of gene expression arrays identifies high risk molecular subtypes in breast cancer. *Appl. Algebra Eng. Commun. Comput.*, 23(1-2):3–15, 2012.

[7] S. Bai, F. Zhang, and P. H. S. Torr. Hypergraph convolution and hypergraph attention. *CoRR*, abs/1901.08150, 2019.

[8] A.-L. Barabási and M. Pósfai. *Network science*. Cambridge University Press, Cambridge, 2016.

[9] U. Bauer. Ripser: efficient computation of vietoris-rips persistence barcodes. *Journal of Applied and Computational Topology*, 2021.

[10] A. Bavelas. Communication Patterns in Task-Oriented Groups. *Acoustical Society of America Journal*, 22(6):725, Jan. 1950.

[11] A. R. Benson. Three hypergraph eigenvector centralities. *CoRR*, abs/1807.09644, 2018.

[12] A. R. Benson, R. Abebe, M. T. Schaub, A. Jadbabaie, and J. M. Kleinberg. Simplicial closure and higher-order link prediction. *CoRR*, abs/1802.06916, 2018.

[13] A. R. Benson, D. F. Gleich, and D. J. Higham. Higher-order network analysis takes off, fueled by classical ideas and new data. *CoRR*, abs/2103.05031, 2021.

[14] A. R. Benson, D. F. Gleich, and J. Leskovec. Tensor spectral clustering for partitioning higher-order network structures. *CoRR*, abs/1502.05058, 2015.

[15] A. R. Benson, D. F. Gleich, and J. Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.

[16] G. Bianconi and C. Rahmede. Emergent hyperbolic network geometry. *Scientific Reports*, 7(1), feb 2017.

[17] C. Bick, E. Gross, H. A. Harrington, and M. T. Schaub. What are higher-order networks? *CoRR*, abs/2104.11329, 2021.

[18] 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):P10008, 2008.

[19] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4):175–308, 2006.

[20] U. Brandes, M. Eiglsperger, J. Lerner, and C. Pich. Graph markup language (graphml). In *Handbook of Graph Drawing and Visualization*, 2013.

[21] K. Carlsson and F. Ekre. Tensors.jl — tensor computations in julia. *Journal of Open Research Software*, 7, 03 2019.

[22] J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35:283–319, 1970.

[23] C. Chen, J. He, N. Bliss, and H. Tong. Towards optimal connectivity on multi-layered networks. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2332–2346, 2017.

[24] P. S. Chodrow. Configuration models of random hypergraphs. *Journal of Complex Networks*, 8(3), 08 2020. cnaa018.

[25] P. S. Chodrow, N. Veldt, and A. R. Benson. Generative hypergraph clustering: From blockmodels to modularity. *Science Advances*, 7(28):eabh1303, 2021.

[26] F. Chung and L. Lu. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25):15879–15882, 2002.

[27] O. T. Courtney and G. Bianconi. Generalized network structures: The configuration model and the canonical ensemble of simplicial complexes. *Physical Review E*, 93(6), jun 2016.

[28] V. De, Silva, and R. Ghrist. Coverage in sensor networks via persistent homology.

[29] D. DeWoskin, J. Climent, I. Cruz-White, M. Vázquez, C. C. Park, and J. Arsuaga. Applications of computational homology to the analysis of treatment response in breast cancer patients. *Topology and its Applications*, 157:157–164, 2010.

[30] M. Do, S.-e. Yoon, B. Hooi, and K. Shin. Structural patterns and generative models of real-world hypergraphs. pages 176–186, 08 2020.

[31] C. Duclos, D. Nadin, Y. Mahdid, V. Tarnal, P. Picton, G. Vanini, G. Golmirzaie, E. Janke, M. S. Avidan, M. B. Kelz, G. A. Mashour, and S. Blain-Moraes. Brain network motifs are markers of loss and recovery of consciousness. *bioRxiv*, 2020.

[32] P. Erdös and A. Rényi. On random graphs i. *Publicationes Mathematicae Debrecen*, 6:290, 1959.

[33] P. L. Erdös, I. Miklós, and L. Soukup. Towards random uniform sampling of bipartite graphs with given degree sequence. 2010.

[34] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao. Hypergraph neural networks, 2018.

[35] A. Ghosh, B. Rozemberczki, S. Ramamoorthy, and R. Sarkar. Topological signatures for fast mobility analysis. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '18, page 159–168, New York, NY, USA, 2018. Association for Computing Machinery.

[36] R. Ghrist and A. Muhammad. Coverage and hole-detection in sensor networks via homology. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, Piscataway, NJ, USA, 2005. IEEE Press.

[37] C. Giusti, R. Ghrist, and D. S. Bassett. Two's company, three (or more) is a simplex: Algebraic-topological tools for understanding higher-order structure in neural data, 2016.

[38] C. Giusti, E. Pastalkova, C. Curto, and V. Itskov. Clique topology reveals intrinsic geometric structure in neural correlations. *Proceedings of the National Academy of Sciences*, 112(44):13455–13460, 2015.

[39] D. F. Gleich. Pagerank beyond the web. *CoRR*, abs/1407.5107, 2014.

[40] X. Gong, D. J. Higham, and K. Zygalakis. Generative hypergraph models and spectral embedding, 2023.

[41] M. Granovetter. The strength of weak ties. *The American Journal of Sociology*, 78(6):1360–1380, May 1973.

[42] J. A. Grochow and M. Kellis. Network motif discovery using subgraph enumeration and symmetry-breaking. In *Proceedings of the 11th Annual International Conference on Research in Computational Molecular Biology*, RECOMB'07, page 92–106, Berlin, Heidelberg, 2007. Springer-Verlag.

[43] M. Hajij, G. Zamzmi, and X. Cai. Simplicial complex representation learning. *CoRR*, abs/2103.04046, 2021.

[44] M. Hajij, G. Zamzmi, T. Papamarkou, N. Miolane, A. Guzmán-Sáenz, K. N. Ramamurthy, T. Birdal, T. K. Dey, S. Mukherjee, S. N. Samaga, N. Livesay, R. Walters, P. Rosen, and M. T. Schaub. Topological deep learning: Going beyond graph data, 2023.

[45] T. Harko, F. S. Lobo, and M. Mak. Exact analytical solutions of the susceptible-infected-recovered (sir) epidemic model and of the sir model with equal death and birth rates. *Applied Mathematics and Computation*, 236:184–194, 2014.

[46] R. A. Harshman. Foundations of the parafac procedure: Models and conditions for an "explanatory" multi-model factor analysis. 1970.

[47] X. Hong Lin, Y. Han-bing, G. Cui-fang, and Z. Ping. Social network analysis based on network motifs. *Journal of Applied Mathematics*, 2014:1–6, 02 2014.

[48] I. Iacopini, G. Petri, A. Barrat, and V. Latora. Simplicial models of social contagion. *Nature Communications*, 10(1), Jun 2019.

[49] C. A. Joslyn, S. Aksoy, T. J. Callahan, L. E. Hunter, B. A. Jefferson, B. Praggastis, E. A. H. Purvine, and I. J. Tripodi. Hypernetwork science: From multidimensional networks to computational topology. *CoRR*, abs/2003.11782, 2020.

[50] M. Kahle. Topology of random simplicial complexes: a survey, 2013.

[51] B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1), Jan 2011.

[52] Z. R. M. Kashani, H. Ahrabian, E. Elahi, A. Nowzari-Dalini, E. Ansari, S. Asadi, S. Mohammadi, F. Schreiber, and A. Masoudi-Nejad. Kavosh : a new algorithm for finding network motifs. *BMC Bioinformatics*, 10, 2009. Article Number: 318.

[53] N. Kashtan and U. Alon. Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences*, 102(39):13773–13778, 2005.

[54] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics (Oxford, England)*, 20:1746–58, 08 2004.

[55] S. Kim, D. Lee, Y. Kim, J. Park, T. Hwang, and K. Shin. Datasets, tasks, and training methods for large-scale hypergraph learning. *Data Mining and Knowledge Discovery*, 37:1–39, 07 2023.

[56] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3):203–271, 07 2014.

[57] J. Kossaifi, Y. Panagakis, A. Anandkumar, and M. Pantic. Tensorly: Tensor learning in python. *Journal of Machine Learning Research*, 20(26):1–6, 2019.

[58] R. Lambiotte, M. Rosvall, and I. Scholtes. Understanding complex systems: From networks to optimal higher-order models, 2018.

[59] G. Lee, F. Bu, T. Eliassi-Rad, and K. Shin. A survey on hypergraph mining: Patterns, tools, and generators, 2024.

[60] G. Lee, M. Choe, and K. Shin. How do hyperedges overlap in real-world hypergraphs? - patterns, measures, and generators. *CoRR*, abs/2101.07480, 2021.

[61] G. Lee, J. Ko, and K. Shin. Hypergraph motifs: Concepts, algorithms, and discoveries. *CoRR*, abs/2003.01853, 2020.

[62] S. Leinhardt and J. Berger. The structure of positive interpersonal relations in small groups. *Sociological Theories in Progress. Boston: Houghton Mifflin*, 1971.

[63] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Signed networks in social media, 2010.

[64] J. Li, J. Bien, and M. T. Wells. rtensor: An r package for multidimensional array (tensor) unfolding, multiplication, and decomposition. *Journal of Statistical Software*, 87(10):1–31, 2018.

[65] J. Li, C. Chen, H. Tong, and H. Liu. *Multi-Layered Network Embedding*, pages 684–692.

[66] L. Lim. Hodge laplacians on graphs. *CoRR*, abs/1507.05379, 2015.

[67] M. Liu, N. Veldt, H. Song, P. Li, and D. F. Gleich. Strongly local hypergraph diffusions for clustering and semi-supervised learning. *Proceedings of the Web Conference 2021*.

[68] Q. Liu, Y. Huang, and D. N. Metaxas. Hypergraph with sampling for image retrieval. *Pattern Recognition*, 44(10):2255–2262, 2011. Semi-Supervised Learning for Visual Content Analysis and Understanding.

[69] Q. F. Lotito, F. Musciotto, A. Montresor, and F. Battiston. Higher-order motif analysis in hypergraphs. *Communications Physics*, 5(1), Apr. 2022.

[70] L. Lu and X. Peng. High-ordered random walks and generalized laplacians on hypergraphs, 2011.

[71] A. Madkour, W. G. Aref, F. U. Rehman, M. A. Rahman, and S. M. Basalamah. A survey of shortest-path algorithms. *CoRR*, abs/1705.02044, 2017.

[72] D. Marcus and Y. Shavitt. Rage – a rapid graphlet enumerator for large networks. *Computer Networks*, 56(2):810–819, 2012.

[73] N. Masuda, M. A. Porter, and R. Lambiotte. Random walks and diffusion on networks. *Physics Reports*, 716-717:1–58, 2017. Random walks and diffusion on networks.

[74] A. P. Millá n, J. J. Torres, and G. Bianconi. Explosive higher-order kuramoto dynamics on simplicial complexes. *Physical Review Letters*, 124(21), may 2020.

[75] R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, 2004.

[76] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, October 2002.

[77] N. Milosavljević, D. Morozov, and P. Skraba. Zigzag persistent homology in matrix multiplication time. In *Proceedings of the Twenty-Seventh Annual Symposium on Computational Geometry*, SoCG '11, page 216–225, New York, NY, USA, 2011. Association for Computing Machinery.

[78] K. Mischaikow and V. Nanda. Morse theory for filtrations and efficient computation of persistent homology. *Discrete Comput. Geom.*, 50(2):330–353, sep 2013.

[79] A. Muhammad and M. Egerstedt. Control using higher order laplacians in network topologies. In *Proc. of 17th International Symposium on Mathematical Theory of Networks and Systems, Kyoto*, pages 1024–1038, 2006.

[80] R. Mulas, C. Kuehn, and J. Jost. Coupled dynamics on hypergraphs: Master stability of steady states and synchronization. *Phys. Rev. E*, 101:062313, Jun 2020.

[81] V. Nanda and R. Sazdanovic. Simplicial models and topological inference in biological systems. 2014.

[82] M. Nicolau, R. Tibshirani, A.-L. Børresen-Dale, and S. S. Jeffrey. Disease-specific genomic analysis: identifying the signature of pathologic biology. *Bioinformatics*, 23(8):957–965, 02 2007.

[83] S. Omidi and F. Schreiber. Moda: An efficient algorithm for network motif discovery in biological networks. *Genes & genetic systems*, 84:385–95, 10 2009.

[84] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828, 2018.

[85] B. Osting, S. Palande, and B. Wang. Towards spectral sparsification of simplicial complexes based on generalized effective resistance. *CoRR*, abs/1708.08436, 2017.

[86] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[87] A. Paranjape, A. R. Benson, and J. Leskovec. Motifs in temporal networks. *CoRR*, abs/1612.09259, 2016.

[88] S. Patra and A. Mohapatra. Application of dynamic expansion tree for finding large network motifs in biological networks. *PeerJ*, 7:e6917, 05 2019.

[89] R. Patro and C. Kingsford. Predicting protein interactions via parsimonious network history inference. *Bioinformatics*, 29(13):i237–i246, 06 2013.

[90] G. Petri, P. Expert, F. Turkheimer, R. Carhart-Harris, D. Nutt, P. J. Hellyer, and F. Vaccarino. Homological scaffolds of brain functional networks. *Journal of The Royal Society Interface*, 11(101):20140873, 2014.

[91] N. Pržulj, D. G. Corneil, and I. Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 07 2004.

[92] P. Ribeiro and F. Silva. G-tries: An efficient data structure for discovering network motifs. pages 1559–1566, 01 2010.

[93] G. Rossetti and R. Cazabet. Community discovery in dynamic networks: A survey. *ACM Comput. Surv.*, 51(2), feb 2018.

[94] R. A. Rossi, N. K. Ahmed, E. Koh, S. Kim, A. Rao, and Y. A. Yadkori. Hone: Higher-order network embeddings, 2018.

[95] A. Sandryhaila and J. Moura. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *Signal Processing Magazine, IEEE*, 31:80–90, 09 2014.

[96] A. Sandryhaila and J. M. F. Moura. Discrete signal processing on graphs. *CoRR*, abs/1210.4752, 2012.

[97] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie. Random walks on simplicial complexes and the normalized hodge laplacian. *CoRR*, abs/1807.05044, 2018.

[98] M. T. Schaub, Y. Zhu, J.-B. Seby, T. M. Roddenberry, and S. Segarra. Signal processing on higher-order networks: Livin' on the edge... and beyond. *Signal Processing*, 187:108149, 2021.

[99] S. Schlag. *High-Quality Hypergraph Partitioning*. PhD thesis, Karlsruhe Institute of Technology, Germany, 2020.

[100] F. Schreiber and H. Schwöbbermeyer. Frequency concepts and pattern detection for the analysis of motifs in networks. In C. Priami, E. Merelli, P. Gonzalez, and A. Omicini, editors, *Transactions on Computational Systems Biology III*, pages 89–104, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[101] P. Shao, Y. Yang, S. Xu, and C. Wang. Network embedding via motifs. *ACM Trans. Knowl. Discov. Data*, 16(3), oct 2021.

[102] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature Genetics*, 31:64–68, 2002.

[103] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. Signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular data domains. *CoRR*, abs/1211.0053, 2012.

[104] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65(13):3551–3582, 2017.

[105] A. Sizemore, C. Giusti, A. Kahn, J. Vettel, R. Betzel, and D. Bassett. Cliques and cavities in the human connectome. *Journal of Computational Neuroscience*, 44:1–31, 02 2018.

[106] C. Spagnuolo, G. Cordasco, P. Szufel, P. Pralat, V. Scarano, B. Kaminski, and A. Antelmi. Analyzing, exploring, and visualizing complex networks via hypergraphs using SimpleHypergraphs.jl. *Internet Mathematics*, apr 2020.

[107] E. H. Spanier. *Algebraic topology*. McGraw-Hill Book, New York, 1966. Includes index.

[108] O. Sporns and R. Kötter. Motifs in brain networks. *PLoS biology*, 2:e369, 12 2004.

[109] A. Tahbaz-Salehi and A. Jadbabaie. Distributed coverage verification in sensor networks without location information. *IEEE Transactions on Automatic Control*, 55(8):1837–1849, 2010.

[110] S. Tan, J. Bu, C. Chen, B. Xu, C. Wang, and X. He. Using rich social media information for music recommendation via hypergraph model. *ACM Trans. Multimedia Comput. Commun. Appl.*, 7S(1), nov 2011.

[111] A. Tausz, M. Vejdemo-Johansson, and H. Adams. JavaPlex: A research software package for persistent (co)homology. In H. Hong and C. Yap, editors, *Proceedings of ICMS 2014*, Lecture Notes in Computer Science 8592, pages 129–136, 2014. Software available at http://appliedtopology.github.io/javaplex/.

[112] J. Thomas, A. Moallemy-Oureh, S. Beddar-Wiesing, and C. Holzhüter. Graph neural networks designed for different graph types: A survey, 04 2022.

[113] H. Tian, S. Jin, and R. Zafarani. Exploiting cross-order patterns and link prediction in higher-order networks. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1–9, 2022.

[114] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966c.

[115] J. Ugander, L. Backstrom, and J. M. Kleinberg. Subgraph frequencies: Mapping the empirical and extremal geography of large graph collections. *CoRR*, abs/1304.1548, 2013.

[116] M. Vaida and K. Purcell. Hypergraph link prediction: Learning drug interaction networks embeddings. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1860–1865, 2019.

[117] N. Veldt, A. R. Benson, and J. Kleinberg. *Minimizing Localized Ratio Cut Objectives in Hypergraphs*, page 1708–1718. Association for Computing Machinery, New York, NY, USA, 2020.

[118] N. Veldt, A. R. Benson, and J. M. Kleinberg. Hypergraph cuts with general splitting functions. *CoRR*, abs/2001.02817, 2020.

[119] J. Wang, K. Ding, L. Hong, H. Liu, and J. Caverlee. *Next-Item Recommendation with Sequential Hypergraphs*, page 1101–1110. Association for Computing Machinery, New York, NY, USA, 2020.

[120] J. Wang, K. Ding, Z. Zhu, and J. Caverlee. Session-based recommendation with hypergraph attention networks. *CoRR*, abs/2112.14266, 2021.

[121] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[122] S. Wernicke. Efficient detection of network motifs. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 3(4):347–359, oct 2006.

[123] J. Xu, T. L. Wickramarathne, and N. V. Chawla. Representing higher-order dependencies in networks. *Science Advances*, 2(5):e1600028, 2016.

[124] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar. Hypergcn: A new method of training graph convolutional networks on hypergraphs, 2018.

[125] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han. Heterogeneous network representation learning: Survey, benchmark, evaluation, and beyond. *CoRR*, abs/2004.00216, 2020.

[126] H. Yin, A. R. Benson, and J. Leskovec. Higher-order clustering in networks. *Physical Review E*, 97(5), May 2018.

[127] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich. Local higher-order graph clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 555–564, New York, NY, USA, 2017. Association for Computing Machinery.

[128] J.-G. Young, G. Petri, F. Vaccarino, and A. Patania. Construction of and efficient sampling from the simplicial configuration model. *Physical Review E*, 96(3), sep 2017.

[129] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang. Self-supervised multi-channel hypergraph convolutional network for social recommendation. *CoRR*, abs/2101.06448, 2021.

Synthetic data for learning-based knowledge discovery

William Shiao
University of California, Riverside
Riverside, CA, USA
wshia002@ucr.edu

Evangelos E. Papalexakis
University of California, Riverside
Riverside, CA, USA
epapalex@cs.ucr.edu

ABSTRACT

Recent advances in deep learning have demonstrated the ability of learning-based methods to tackle very hard downstream tasks. Historically, this has been demonstrated in predictive tasks, while tasks more akin to the traditional KDD (Knowledge Discovery in Databases) pipeline have enjoyed proportionally fewer advances. Can learning-based approaches help with inherently hard problems within the KDD pipeline, such as *"how many patterns are in the data"*, *"what are different structures in the data"*, and *"how can we robustly extract those structures?"* In this vision paper, we argue for the need for synthetic data generators to empower cheaply-supervised learning-based solutions for knowledge discovery. We describe the general idea, early proof-of-concept results which speak to the viability of the paradigm, and we outline a number of exciting challenges that await, and a set of milestones for measuring success.

1. INTRODUCTION

Supervised and self-supervised learning has made and continues to be making tremendous strides. Numerous examples include (but are not limited to) language models [7; 14], vision models [12; 5], graph neural networks [26; 11], and even some "general purpose" models that can work for multiple data types and tasks [3]. The superiority of these modern deep learning models is primarily shown in downstream tasks that are predictive in nature, e.g., image classification, speech recognition, General Language Understanding Evaluation (GLUE) [21] tasks, graph node classification or link prediction.

In stark contrast to traditional downstream tasks, tasks that relate to what we collectively call Knowledge Discovery in Databases (KDD) or "the KDD process" [9] have enjoyed considerably less attention and, as a result, significantly fewer advances. This disparity, at first glance, is rather understandable since tasks that pertain to the KDD process are much more open-ended than prediction or classification-based downstream tasks and are inherently unsupervised in nature.

However, when we look at the state of the art of the KDD process overall, there has been steady and significant progress made in introducing new mining algorithms, new pre or post-processing techniques, and new evaluation techniques, but for the most part, the "glue" of any practical such pipeline is by-and-large human-based. Many design choices and algorithmic hyperparameters in that pipeline are typically chosen by an experienced data scientist and are a result of the application of a number of heuristics and copious amounts of trial-and-error experimentation.

A natural question that arises is whether recent advances in deep (self-)supervised learning can transform the way that practitioners perform the KDD process in similar ways that they have transformed the way in which we approach classification and prediction problems in real life. For instance, can we use cutting-edge deep learning methods to solve inherently hard problems which lie at the heart of the KDD process, such as "Are there any interesting patterns in my data? If so, how many, and what kinds of structure(s) do they follow?" Furthermore, can we do so while having no real supervision—without real data with annotations that directly answer those questions? In this vision paper, we propose a "Blue Sky" idea, borrowing the terminology from the initiative set forth by the Computing Research Association (CRA) [6], to tackle the above question, towards transforming the process of knowledge discovery.

2. PROPOSED VISION

The Blue Sky idea: The key to transforming data discovery is the design of *high-quality realistic synthetic data* used in conjunction with cutting-edge deep (self-)supervised machine learning models. An overview of the proposed idea is shown in Figure 1.

Unlike "traditional" supervised approaches, this paradigm introduces "cheap" supervision where human involvement is ideally zero (or close to zero), thus remaining essentially unsupervised. Furthermore, this eliminates the current need for running the analytical pipeline (or parts thereof) multiple times, in trial-and-error mode, in order to manually or heuristically determine the best result out of the myriad executions. Because of the quick response/inference time of modern deep models, this idea has the potential to decrease the KDD process execution time by orders of magnitude.

In addition to practicality and scalability, this idea, extending existing efforts for uncertainty quantification in "traditional" supervised scenarios, can allow for robust hypothesis testing and provide uncertainty bounds on the presence of certain types of structure in real data.

Finally, this idea has the potential to allow us to solve problems for which we currently have no widely accepted solution by generalizing from examples and problems that are "easier" and for which we have acceptable solutions, by leveraging the problem structure (see Section 3 for an example).

Why is it a Blue Sky idea? The proposed idea has the potential to transform the traditional KDD process, which

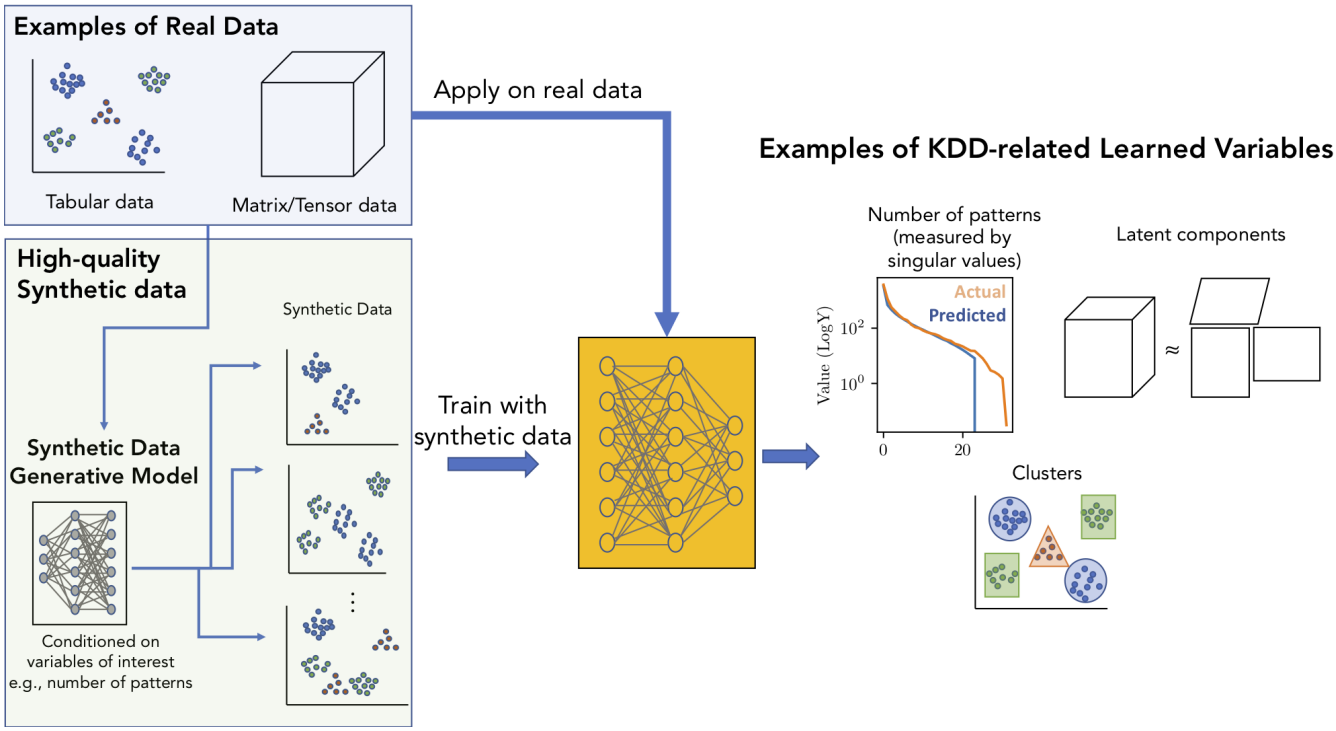


Figure 1: Overview of the proposed vision.

is especially useful and relevant in emerging domains where insights and structure in the data are the desired outputs. Furthermore, problems that this proposed idea is promising to tackle are extremely hard and usually left to be solved manually by the end user of a given algorithm/pipeline employing heuristics.

At the heart of it, the proposed idea aims towards a unified and generalizable framework for a very heterogeneous and multi-faceted process and can ultimately push the frontier of data mining in the design of automated and personalized KDD pipelines.

Why should the community ponder over it? The data mining community has the collective expertise and domain knowledge necessary for this kind of endeavor and can inject it into the data generation process.

Moreover, the proposed idea is a treasure trove of interesting and hard research problems: It poses a number of fascinating and unique challenges that can not only advance the state of the art in the KDD process, but are also poised to advance generative models and (self-)supervised learning since both the kinds of data to be generated and learn generalizable representations from are novel in that context.

Why now? Currently, the data mining and machine learning communities have a mature understanding of a number of crucial components that are necessary for executing this research agenda, ranging from recent advances in generative models (from adversarial generation [10] to diffusion models [15]) and deep (self-)supervised models. The twist, of course, is that this understanding pertains to the current use of the above methods, which is not necessarily aligned with the proposed use, which in itself poses interesting challenges.

It is important to acknowledge here the broad and profound impact that synthetic data have already had in data mining and machine learning, starting from classical and powerful oversampling techniques, such as SMOTE [4], to the advances of Generative Adversarial Networks (GANs) [10] and the remarkable results produced by Diffusion [15] and Transformer-based Large Language Models [1, 20].

The key novelty here is that our proposed synthetic data generation is not focused on data augmentation or mere generation of realistic data (with the term "mere" meant here strictly as a qualifier of single-purpose and by no means implies that such generation is trivial) but should rather focus on hidden patterns in the data and the inclusion of "knobs" such as the "number of patterns", which render the synthetic data more suitable for exploring different aspects of the KDD process.

3. PROOF OF CONCEPT

We would like to offer two particular data points of reference which provide preliminary results for the viability of our general idea. The particular hard problem at hand that we have been focusing on is the identification of the low-rank in matrix or tensor data, from which one can draw parallels to problems such as identifying the number of clusters in data [8].

In recent work [23], we demonstrated that we can successfully learn the singular value profile of a given matrix, which is essentially what is needed in order to identify the full and the low rank of that matrix. Given that this has been successful in matrix data, can we generalize it to tensor data, where this problem is extremely hard and wide open, by leveraging the algebraic structure of the two different prob-

lems? In concurrent work [18], we show that by using simple but carefully-designed synthetic tensor data, where the low rank is known, we can accurately learn the low rank.
We, by no means, claim that we have solved this problem, however, those two instances provide strong evidence for the viability of our proposed paradigm.

4. RESEARCH CHALLENGES

A number of exciting research challenges need to be addressed for this paradigm shift to take effect.

4.1 Designing data generators

The design of synthetic data generators is of paramount importance. Generators ought to obey the following properties:
P1: Generate *realistic* data which closely mimic the distribution of real data.
P2: Offer *control over parameters* of importance to the KDD process (e.g., number of clusters in the data).
P3: Offer substantial *diversity* in the generated data points such that they can be used to successfully train a model that learns generalizable features.
For example, we recently introduced generation of graph adjacency matrices [17] and tensors [16], where the rank is a controllable parameter of the generator.

4.2 Evaluating realism

When we are generating synthetic data, even though our goal is not the generation of novel-looking data (e.g., images), we still have to make sure that the generated data are realistic, in that the closely follow the distribution of the real data.
When measuring realism, we may need to take modality-specific approaches (e.g., treat images differently from graphs), and when generating synthetic data for novel and emerging applications, we have to carefully decide upon "realism" criteria that which we can use to hold our data to this important test. We can derive such an example from our recent work [16] where the goal is to generate multiplex graphs. Even though there exist established realism criteria for single graphs, applying them on each individual view of the multiplex graph is not enough, since a major consideration for the output data is that each generated graph view is not independent from the rest. Thus, in order to capture this relation across graph views, and how close it is to real data, we would have to define novel tests. In the particular case at hand, we opted for viewing the generated multiplex graph as a tensor, and measure how "compressible" it is for different decomposition ranks, and subsequently compared this behavior against the one observed for real-world multiplex graphs when treated as such.
Finally, beyond realism in the raw feature dimensions of the data (such as realism in produced images), in this case we should be able to measure realism in the hidden pattern dimensions of the data as well. For example, in the application of community detection in graphs, earlier work has demonstrated that in many real-world graphs communities have hyperbolic shapes [2]. In this case, a "community" is essentially a hidden pattern in the generated graph data, and ensuring that its generation adheres to this real-world observation, when supported by the data and application of interest, can enhance the realism of the latent patterns in our synthetic data.

4.3 Limited real data & knowledge-guided generation

Modern generative models assume that we have some seed real data available from which we learn their distribution and successfully generate new data points. What if we have no real data available, or the amount of data is rather insufficient for generating a diverse-enough synthetic dataset?
In such data-scarce scenarios, we may resort to model and knowledge-guided design of synthetic data, a process which would essentially bring knowledge-guided machine learning approaches [13] to our paradigm, and where we would infuse model-based knowledge to the data generation to compensate for the lack of real data.

4.4 Representation learning

How can we learn effective representations from structured or unstructured data which work for KDD-process downstream tasks?
It may be tempting to immediately endeavor to learn those representations fully automatically using deep learning modeling. As in most scenarios, doing so without having a firm grip over the different kinds of bias that are introduced in the generated process may yield suboptimal representations. Thus, in conjunction with fully-automated representation learning, domain-expertise-guided feature generation may be a reasonable first step which would allow us to understand what features work and what features fail (such as in our proof of concept work, where we define a set of descriptive features for tensor data, based on years of expertise [18]), and progressively "graduate" to fully-automated representations.

4.5 Generalization and transfer across tasks

This challenge is highly related to the previous one of representation learning. However, it underscores an important requirement for our approach to be generalizable and transferable when there exist structural similarities across tasks and when solutions exist for simpler tasks, and we wish to generalize to harder instances.

4.6 Designing end-to-end KDD pipelines

When we integrate all the different advances together into an analytical pipeline, this may look vastly different from existing pipelines. For example, we may be able to tailor entire pipelines to a specific problem and accordingly build multiple personalized KDD pipelines. Alternatively, we may opt for a generalist solution where a single powerful pipeline can handle most cases.
In addition to building the pipeline, under this approach, we may be able to offer more robust uncertainty quantification while reducing the execution time of a single pipeline by orders of magnitude, which may invite us to rethink the overall design, especially as it may integrate with domain experts in the loop.

4.7 Robust evaluation

Given that the nature of most problems that our proposed idea is poised to tackle is extremely hard, evaluation poses

a unique challenge in itself. As mentioned above, this new paradigm may allow us to revisit the design of the analytical pipeline, where interaction and potential evaluation by a domain expert may be much more scalable than ever before. We anticipate that evaluation should heavily rely on the help of domain experts, either directly or indirectly. For instance, when evaluating the accuracy of tensor rank learning in [18], we rely on chemometrics expertise which links rank to the number of chemicals in a mix.

5. MEASURING SUCCESS

In order to measure success of the proposed approach, the following milestones have to be progressively met, ideally for a number of different KDD-related problems. **M1**: Solve problems that we can already solve exactly (e.g., matrix rank and singular value profile): Success is measured by how far we are from the exact solution

M2: Solve problems for which we have widely acceptable and easy-to-use heuristics (e.g., matrix *low* rank or finding the number of clusters in K-means using the "elbow method"): Success is measured by how far we are from solutions produced by data scientist experts using heuristics afforded to them and their best judgement.

M3: Apply to problems for which there is no widely accepted heuristic solution (e.g., tensor low rank): Success will be measured by focusing on real-world application domains in collaboration with domain experts.

(a) Direct measures for M3: Do the results agree with what domain experts know to be true (after translating KDD terms such as "cluster" to the domain language, such as "phenotype")? Do domain experts evaluate results favorably to existing methods? **(b) Indirect measures for M3**: Did the application of the learning-based KDD process in a particular domain lead to a significant discovery in that domain?

6. DISCUSSION AND CONCLUSION

Our idea has parallels to another emerging direction which involves the use of Reinforcement Learning (RL) in solving hard data mining problems, such as fine-tuning the popular DBSCAN clustering algorithm [24]. We believe that the two approaches are synergistic and we are interested in exploring their interplay.

As this paper is meant to start a discussion around this topic and explore the opportunities and limitations of the proposed direction, we envision that there is a set of problems that where the proposed direction can have more immediate impact:

- Learning-based solutions developed as part of this vision can serve as:
  - **Auxiliary parts of a KDD pipeline**, such as replacing or augmenting existing heuristics that guide the discovery (such as Cluster Validation Indices [19])
  - **More optimistic: Main parts of a KDD pipeline**, where the learning-based solution will be able to learn either elements of the desired solution (e.g., cluster membership between two different points) or the entire desired solution (e.g., cluster assignments, alignment between data points [25, 22], etc)
- **Generalizing from simpler to harder problems**, where we can develop models in cases where there exist exact analytical descriptions for the sought-after patterns or latent variables, and work to extend them to cases where such analytical solutions no longer exist.

In closing, in this vision paper, we propose the transformation of the KDD process through the use of synthetic data which can train powerful deep learning models tailored to tackling the hardest problems in knowledge discovery from data.

7. ACKNOWLEDGEMENT

This work was supported by the National Science Foundation under CAREER grant no. IIS 2046086 and CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems (MECIS) grant no. 2112650, and by the Combat Capabilities Development Command Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Combat Capabilities Development Command Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes not withstanding any copyright notation here on.

8. REFERENCES

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] M. Araujo, S. Günnemann, G. Mateos, and C. Faloutsos. Beyond blocks: Hyperbolic community detection. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I 14*, pages 50–65. Springer, 2014.

[3] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*, 2022.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[5] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[6] C. R. A. (CRA). Blue sky ideas. `https://cra.org/ccc/visioning/blue-sky/`.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[8] C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 606–610. SIAM, 2005.

[9] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, et al. Knowledge discovery and data mining: Towards a unifying framework. In *KDD*, volume 96, pages 82–88, 1996.

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. 6 2014.

[11] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[12] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[13] A. Karpatne, R. Kannan, and V. Kumar. *Knowledge Guided Machine Learning: Accelerating Discovery using Scientific Knowledge and Data*. CRC Press, 2022.

[14] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[16] W. Shiao, B. A. Miller, K. Chan, P. Yu, T. Eliassi-Rad, and E. E. Papalexakis. Tengan: adversarially generating multiplex tensor graphs. *Data Mining and Knowledge Discovery*, 38(1):1–21, 2024.

[17] W. Shiao and E. E. Papalexakis. Adversarially generating rank-constrained graphs. In *2021 IEEE DSAA*, pages 1–8. IEEE, 2021.

[18] W. Shiao and E. E. Papalexakis. Frappe: Fast rank approximation with explainable features for tensors. *arXiv preprint arXiv:2206.09316*, 2022.

[19] W. Shiao, U. S. Saini, Y. Liu, T. Zhao, N. Shah, and E. E. Papalexakis. Carl-g: Clustering-accelerated representation learning on graphs. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2036–2048, 2023.

[20] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[21] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[22] Y. Wu, U. S. Saini, J. Chen, and E. E. Papalexakis. Tenalign: Joint tensor alignment and coupled factorization. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 568–577. IEEE, 2022.

[23] D. Xu, W. Shiao, J. Chen, and E. E. Papalexakis. Sv-learn: Learning matrix singular values with neural networks. In *2022 IEEE ICDM Workshops*. IEEE, 2022.

[24] R. Zhang, H. Peng, Y. Dou, J. Wu, Q. Sun, Y. Li, J. Zhang, and P. S. Yu. Automating dbscan via deep reinforcement learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2620–2630, 2022.

[25] S. Zhang, H. Tong, Y. Xia, L. Xiong, and J. Xu. Nettrans: Neural cross-network transformation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 986–996, 2020.

[26] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

The Case for Hybrid Multi-Objective Optimisation in High-Stakes Machine Learning Applications

Alex A. Freitas
University of Kent, UK
School of Computing, University of Kent
Canterbury, CT2 7FS, United Kingdom

ABSTRACT

Most classification (supervised learning) algorithms optimise a single objective, typically the predictive performance of the learned classification model. However, in high-stake classification applications, involving e.g. decisions about whether or not an individual should undergo a medical surgery, be granted a loan or be hired for a job, often there is a need to optimise multiple objectives, such as the predictive performance, interpretability or fairness of the learned model. In this context, this position paper discusses the pros and cons of two different multi-objective optimisation approaches (the Pareto and the lexicographic approaches), and proposes a conceptual framework for hybrid multi-objective optimisation, combining those two approaches.

Keywords

Classification, multi-objective optimisation, Pareto dominance, lexicographic optimisation.

1. INTRODUCTION

Classification algorithms, a major type of supervised machine learning algorithms [39], [64] are currently ubiquitously applied in a wide range of application domains; including domains that involve high-stakes decisions about people, e.g. predicting who should be granted a loan, hired for a job, undergo a surgery, etc. In such applications, it is often desirable that a classification algorithm should optimise not only predictive accuracy but also several other quality criteria of the learned model, such as its interpretability, fairness, etc. Optimising these criteria separately, one at a time, is in general not a good option, since there are usually strong trade-offs between different types of criteria – for instance, the trade-offs between accuracy and interpretability [12], [42], [62], [52] between accuracy and fairness [63], [1], [49], [59], between accuracy and inference time [65], [31], and between accuracy and privacy [8], [50]. Hence, there is a clear need for multi-objective optimisation methods that optimise multiple criteria (objectives) at the same time.

Furthermore, for each of these broad types of criteria (e.g. accuracy, interpretability, fairness), there are usually multiple specific measures of the quality of a predictive model measuring different aspects of that criterion – discussed e.g. in [25], [30], [37] for predictive accuracy measures; [38], [10], [60] for fairness measures; and [6], [41] for interpretability measures. Each of such specific measures of a model's quality can also be considered as a separate objective to optimised, leading again to the need for multi-objective optimisation methods to obtain more robust results. For example, there is no predictive accuracy measure which is universally superior to all other measures, with different accuracy measures having different pros and cons [21], [23], [44]; and so, in practice it makes sense to try to optimise more than one accuracy measures, to perform a more robust evaluation of predictive accuracy. There are also trade-offs between different measures of interpretability [48], [40] and different measures of fairness [2], [29], [9].

The need for multi-objective optimisation also arises naturally in several types of machine learning (sub)-areas. For example, multi-task learning problems in general can be naturally cast as multi-objective optimisation problems [53], where predictive accuracy in each task is an objective to be optimised. In addition, in the area of multi-label classification, which is a specific type of multi-task learning, it is standard procedure to report the results of multiple measures of predictive accuracy, since no measure captures all the nuances of multi-label classification performance [58], [45], [4]. Optimising multiple multi-label predictive accuracy measures can be naturally cast as a multi-objective optimisation problem. As another example, in federated learning [33], since the data and model computation have to be distributed across many local processors, including low-speed, low-memory local devices, objectives to be optimised include predictive accuracy, model complexity, communication costs and memory requirements on local devices [66].

Yet another machine learning area with a strong and natural need for multi-objective optimisation is Automated Machine Learning (Auto-ML), which essentially consists of using an optimisation method to search for the best learning algorithm (or pipeline) and its best hyper-parameter settings for an input dataset [3], [26], [67]; where, in the literature, "best" usually means "most accurate" based on a given objective function. However, given the very large and heterogenous search space of Auto-ML systems, there is a clear motivation to optimise not only predictive accuracy but also the computational resources (e.g. time) to learn each classifier, leading to 'resource-aware multi-objective optimisation' [65]. This is particularly relevant in the area of neural architecture search, a sub-area of Auto-ML where the search space includes (deep) neural network architectures – whose training usually requires a large amount of time and memory [24], [66]. In this scenario, multi-objective optimisation has been used to simultaneously optimise predictive accuracy and other objectives such as a network's inference time [28], [15], [16], a network's number of parameters [16], [15] or number of floating point operations / multiply-add operations [15], [36], [16], or memory usage on mobile phones [15].

Despite this clear need for multi-objective evaluation of predictive models in a wide range of classification problems, the vast majority of the literature still focus on the traditional framework of single-objective optimisation, focusing mainly on predictive accuracy – and often a single measure of predictive accuracy.

When multiple objectives are optimised in supervised learning, this is usually implemented by converting the original multi-objective problem into a single-objective one by using a linear combination (weighted sum) of the original objectives of the form: $w_1 \times Obj_1 + \ldots + w_m \times Obj_m$, where w_i, $i = 1,\ldots,m$, denotes the weight assigned to objective Obj_i, and m is the

number of objectives to be optimised. This approach has the advantage of conceptual simplicity, but it also has clear disadvantages: it requires the specification of *ad-hoc* weights to each objective, and each run of the optimisation algorithm considers only one possible trade-off among the objectives. In practice, to consider multiple trade-offs, users could run the algorithm many times by varying the objectives' weights across the runs, but this is inefficient (very time-consuming) and ineffective [13], [11], [19], since each run of the algorithm ignores valuable information about the qualities of candidate solutions evaluated in previous runs of the algorithm.

This article focuses instead on two genuinely multi-objective optimisation approaches, namely the Pareto and the lexicographic approaches [13], [18]. Both approaches have the advantage of exploring multiple trade-offs between the different objectives in a single run of the optimisation algorithm, avoiding the need for mixing different objectives into a linear combination of weighted objectives. In essence, the Pareto approach finds a set of 'non-dominated solutions' (the Pareto front) where, for each solution *s* in the Pareto Front, there is no other solution that performs better than *s* for at least one objective and performs at least as well as *s* for all other objectives; whilst the lexicographic approach optimises the multiple objectives in decreasing order of their priorities. These approaches will be reviewed in Section 2.

In the literature on multi-objective optimisation for machine learning, the Pareto approach is in general much more popular than the lexicographic approach. Actually, the Pareto approach is often presented as the only good approach to avoid the disadvantages of the weighted sum approach, and the Pareto approach's limitations are often ignored or downplayed; whilst the lexicographic approach is often ignored. As evidence for this, several surveys of multi-objective optimisation do not even mention the lexicographic multi-objective optimisation approach [56], [57], [34], [35], [43], [54].

In this context, this position article has two contributions. The first one is to show that the Pareto and the lexicographic approaches have to a large extent complementary pros and cons, i.e., none of them is inherently better than the other; and in real-world applications, their use should be determined mainly by the needs and interests of users and the requirements of the target application domain. The second contribution is to propose a new conceptual, hybrid multi-objective optimisation framework designed for synergistically combining the best aspects of the Pareto and lexicographic approaches, in order to offer users an effective and flexible approach for multiple objective optimisation – particularly in the context of high-stakes real-world machine learning applications, where there is a strong need for optimising multiple objectives, as discussed earlier.

The remainder of this article is organised as follows. Section 2 briefly reviews background on the Pareto and lexicographic approaches, to make this article more self-contained. Section 3 discusses the pros and cons of these two approaches. Section 4 described the proposed conceptual, hybrid framework for multi-objective optimisation. Section 5 reports the conclusions.

2. BACKGROUND

The Pareto approach is based on the concept of Pareto dominance between candidate solutions (classifiers, in this article). When comparing two classifiers, a classifier C_1 dominates another classifier C_2 if and only if: C_1 is better than C_2 with respect to at least one objective, *and* C_1 is not worse than C_2 with respect to all the objectives [13], [17], [18]. More formally, let $f_i(C_j)$ denote the value of the i-th objective for classifier C_j. Assuming, without loss of generality, that all m objectives are to be maximised, a classifier C_1 dominates another classifier C_2 if and only if: $\exists i$ such that $f_i(C_1) > f_i(C_2)$ and $\forall i$, i=1,…,m, $f_i(C_1) \geq f_i(C_2)$; where m is the number of objectives being optimised. A classifier is said to be non-dominated if it is not dominated by any other classifier.

The concept of Pareto dominance is illustrated in Figure 1, using as an example a hypothetical case where there are two objectives to be maximised, namely the predictive accuracy of a classifier and the fairness of its predictions. In Figure 1, classifier B is clearly dominated by classifier A, which has better accuracy and better fairness. Likewise, classifier D is dominated by classifier C. Classifier E is also dominated by classifier C, because, although classifiers C and E have the same accuracy, C has better fairness, which satisfies the aforementioned definition of Pareto dominance. In addition, classifier G is dominated by classifiers C, E, F. Finally, classifiers A, C, F are non-dominated, and they form the Pareto front in the context of the 7 classifiers in this simple example.

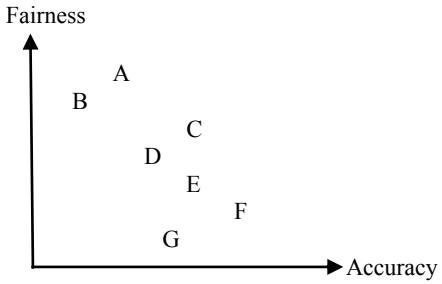


Figure 1: Examples of Pareto dominance

In the Pareto approach, in general the optimiser aims at finding the set of all non-dominated classifiers. However, the only way to guarantee that all non-dominated solutions are found would be to perform an exhaustive search evaluating all candidate solutions in the search space, which is not feasible in general. Hence, in practice Pareto-based optimisers return the best estimate of the set of non-dominated solutions that they were able to find within their computational budget. In most works in this area, it is simply assumed that all the non-dominated solutions found by the optimiser can be returned to the user and that the user would then presumably choose one of those solutions to be deployed in the real-world, based on the user's preferred trade-off among the multiple objectives [13], [27] (the pros and cons of leaving such choice to the user are discussed later). In some works, however, the optimiser returns only a subset of the found non-dominated solutions to simplify the user's analysis of those solutions, as discussed later.

The lexicographic approach requires the user to define a priority ordering for the objectives, and then the objectives are optimised in decreasing order of priority [18], [68], [20], [5]. That is, in order to select the best out of two classifiers, they are first compared with respect to the first (highest-priority) objective. If one classifier is better than the other regarding that objective, the former is declared the winner. Otherwise (i.e. there is a "tie" in the objective values of the two classifiers), the two classifiers are

compared with respect to the second objective. Again, if one classifier is better than another regarding that objective, the former is declared the winner, and so on, until a winner is chosen. When comparing classifiers, the choice of a winner depends on how "a tie" is defined for two values of an objective. In the simpler case of objectives with discrete values, a tie can be defined as two classifiers having exactly the same discrete value for an objective. However, in machine learning it is more common to have real-valued objectives, and in this case a tie is usually defined as a difference of objective values that is smaller than or equal to a small ε (a "tolerance threshold"), so that a classifier is "better than" another regarding an objective only if the difference in their objective values is greater than ε. Finally, if two classifiers are tied regarding all objectives based on the tolerance threshold, the best classifier can be chosen as the one with the best value of the first objective, ignoring the tolerance threshold.

To clarify the use of the lexicographic approach, let us consider a hypothetical case where again there are two objectives to be maximised, namely the predictive accuracy of a classifier (Acc) and the fairness of its predictions (Fairn), both objectives taking a value in the range [0…1] for each classifier. Assume that the user specified that maximizing Acc has priority over maximizing Fairn, and the tolerance threshold is ε = 0.01.

Consider now two classifiers: C_1, with Acc = 0.7 and Fairn = 0.8; and C_2, with Acc = 0.9 and Fairn = 0.6. When comparing classifiers C_1 and C_2 based on the lexicographic optimization approach, C_2 is declared the better classifier because it has substantially better Acc, i.e., the difference between C_2's Acc and C_1's Acc, 0.2 (0.9 – 0.7), is greater than ε (0.01). In this case, the fact that C_1 has substantially better fairness does not affect the result of the lexicographic comparison, because C_2 won over C_1 in the higher-priority objective of accuracy, so there is no need to consider the lower-priority objective of fairness.

Extending the previous example, consider now a classifier C_3, with Acc = 0.69 and Fairn = 0.85, and the classifier C_1 of the previous example (with Acc = 0.7 and Fairn = 0.8). Now, when comparing classifiers C_1 and C_3 based on the lexicographic optimization approach, they are "tied" in the higher-priority Acc objective, i.e. there is no substantial difference in their Acc values, since their Acc difference of 0.01 is not greater than the tolerance threshold ε (0.01). Hence, C_1 and C_3 need to be compared in terms of the lower-priority objective of fairness. In this case, C_3 has a substantially better Fairn value than C_1, with a difference of 0.05 (0.85 – 0.8), which is greater than the tolerance threshold ε (0.01). Therefore, C_3 is declared the winner of the lexicographic comparison; meaning that, in this case, it is acceptable to incur a small, non-substantial (1%) loss of accuracy in order to achieve a substantial gain of fairness, based on the user-defined priority order of the objectives and tolerance threshold.

Several examples of the use of the Pareto and lexicographic approaches in the classification task will be given in Section 4, where a hybrid Pareto/lexicographic multi-objective optimization framework is proposed.

3. PROS AND CONS OF THE PARETO AND LEXICOGRAPHIC APPROACHES

This section discusses the pros and cons of these two approaches in the context of two main issues: (a) how each approach copes with users' preferences about different objectives (Section 3.1); and (b) how users cope with the solution(s) returned by the multi-objective optimizer (Section 3.2).

3.1 Coping with Users' Preferences About Different Objectives

First, since the Pareto approach is agnostic regarding the relative importance of the objectives, it is a natural choice in scenarios where the user does not have any preference about the objectives. This partly explains the popularity of the Pareto approach in the academic literature. In many papers on multi-objective machine learning, the authors are data analysts with expertise on machine learning, rather than users with expertise on the data and its application domain, and the learned models are not used to make decisions in the real-world. In this context of academic research, it is intuitively appealing to data analysts to use the Pareto approach, which avoids the need to decide how to prioritise some objective(s) over others in the real-world.

In many real-world applications, however, users may naturally want to prioritise the optimisation of some objective(s) over others. For example, intuitively most users would prioritise the optimisation of a model's predictive accuracy over other objectives, like a model's interpretability or fairness; whilst some users might prioritise, e.g., fairness or privacy even over accuracy, if there is a legal requirement for fairness or privacy. In scenarios where users can easily specify a clear priority ordering for multiple objectives, the lexicographic approach is intuitively more natural, since it allows the optimisation algorithm to take the very important user preferences into account, whilst those preferences would be ignored by the Pareto approach [5]. It should also be noted that, in practice, it is usually much easier for users to specify a (qualitative) priority order for objectives than specifying the precise numerical (quantitative) weights for all objectives as required in the weighted-sum approach. For example, it is natural for a user to say that predictive accuracy has priority over model size; but it would be much harder for a user to justify why the weights for accuracy and model size should be e.g. 0.8 and 0.2, or 0.67 and 0.33, or whatever other weights.

In addition, a point that is usually ignored in the Pareto optimisation literature is that often the user will be interested in just a region of the Pareto front [19], [61], [47], and in such cases searching for the entire Pareto front would involve a waste of computational resources. For example, in the common scenario where maximising predictive accuracy has priority over minimising model size, a model with the smallest possible size and very low accuracy might be selected and remain in the Pareto front (to be compared against other models for updating the Pareto front) for many iterations of the optimiser, despite being clearly an unacceptable solution to users. In general, such a model would not be selected by the lexicographic approach, due to its very low accuracy (as the higher-priority objective).

On the other hand, an argument commonly used against the lexicographic approach is that, unlike the Pareto approach, the lexicographic approach has the disadvantage of requiring the specification of ad-hoc tolerance thresholds. At first glance, one could argue that, *in theory*, such tolerance thresholds are about as much ad-hoc as the numerical weights for each objective in the baseline weighted-sum approach. Actually, in the lexicographic approach, broadly speaking, other things being equal, an objective's importance is inversely proportional to its tolerance threshold value – since the smaller the tolerance threshold for an

objective, the fewer the "ties" between two values of that objective (for two solutions), meaning that the objective will be used more often to choose the winner solution when comparing two candidate solutions.

However, as the old saying goes: "in theory there is no difference between theory and practice, but in practice there is". In practice, the tolerance thresholds of the lexicographic approach are less problematic than the numerical weights of the weighted-sum approach, as follows.

First, there is in principle no need for any tolerance threshold when an objective to be optimised takes discrete values (like e.g. the objective of minimising the depth or size of a decision tree), since in this case there is a natural "tie" between two solutions when they have exactly the same discrete value of that objective. However, as mentioned earlier real-valued objectives are more common in machine learning; and some tolerance threshold is required when comparing two classifiers regarding a real-valued objective, for two reasons: in practice a difference very close to zero tends to be irrelevant, and strict equality is not a good operator to use when comparing two real-valued numbers in a computer (with finite arithmetic).

Note, however, that although the tolerance thresholds of the lexicographic approach have the effect of performing some fine-tuning of the relative importance of the different objectives, in practice the relative importance of an objective in this approach is still by far primarily determined by its position in the ordered priority list. In theory we could use a tolerance threshold to radically change an objective's importance, e.g. if we set the tolerance threshold of the highest priority objective to infinite, then there would always be a tie in that first objective, which would eliminate that objective's importance. In practice, however, no one sets tolerance threshold to infinite or even large values, tolerance thresholds are in general simply set to small values, say from about 1% to 5% of the range of values for an objective. With such "reasonably small values", tolerance thresholds have much less influence on the relative importance of different objectives than the user-specified priority order of objectives (which is an effective form of incorporating the user's preferences into the optimiser).

Another point is that, as long as different objectives have been normalised to the same range of values, in many cases it seems reasonable to specify a single value of a tolerance threshold for all (real-valued) objectives, rather than different values for different objectives. This substantially reduces the number of "ad-hoc" parameters.

In summary, arguably the need for specifying tolerance thresholds still counts as a disadvantage of the lexicographic approach, by comparison with the Pareto approach (which does not use such thresholds), but this disadvantage is in general substantially smaller than the disadvantage of having to specify ad-hoc numerical weights for the objectives in the weighted-sum approach. In addition, the use of tolerance thresholds is usually a price worth paying for the benefit of directly specifying the user's relative preferences for the multiple objectives to be optimised, in cases where the user has clear preferences (which would be ignored in the standard Pareto approach).

Note also that, although the Pareto approach does not *explicitly* require any parameter to cope with the users' preferences about different objectives, in practice, at the algorithm level, in order to search for the best Pareto front, a Pareto-based optimiser usually has some *implicit* parameters associated specifically with the Pareto optimisation process. For example, the NSGA-II algorithm [14], probably the most popular Pareto-based optimiser, uses a "crowding" procedure that encourages diversity in the non-dominated solutions in the Pareto front maintained by the algorithm along its search. It is claimed in [14] that this procedure does not require any user-specified parameter, but this procedure involves at least the choices of a distance function and a normalisation procedure for distance computation, which in practice can be considered implicitly user-specified parameters.

3.2 Coping with the Solution(s) Returned by the Multi-Objective Optimiser

In the Pareto approach the optimiser returns a set of non-dominated solutions, since the Pareto dominance concept is completely agnostic with respect to the relative importance of the different objectives, and so there is no clearly "best" solution among all the non-dominated solutions. As mentioned earlier, the standard approach for coping with a large number of non-dominated solutions returned to the user is to simply assume that it is up to the user to choose a single best among all non-dominated solutions using their own subjective preference [13], [27]. This approach is usually acceptable in academic research where the solutions returned by the optimiser will not be deployed in the real world.

However, in real-world applications, this approach can be regarded as a double-edged sword, considering that in many applications ultimately a single solution needs to be chosen for practical reasons. On one hand, returning many non-dominated solutions provides more flexibility to users, giving them the chance of using their subjective evaluation of the pros and cons of different solutions (i.e. the extent to which different measures were optimised) to choose the best solution. Importantly, since the user makes this choice by considering a set of "high-quality" non-dominated solutions *a posteriori* (after the optimiser returned its results), this is a much more well-informed choice than the much less well-informed choice of ad-hoc weights for each objective *a priori* (before running the optimiser) in the weighted-sum approach [13], [18].

On the other hand, users may find it difficult to subjectively choose among a large (often very large) set of non-dominated solutions. There are automated methods for choosing a subset of "the best" non-dominated solutions [46], [55], [65], [32], so that the user could focus their attention on a relatively small set of most promising solutions. However, there is no guarantee that such methods will choose the solution that would be really the best solution for the user in practice, since such methods tend to ignore users' preferences.

By contrast, in the lexicographic approach the optimiser returns a single optimised solution, representing the best trade-off among the objectives found by the lexicographic optimiser, which took into account the user's priority ordering of objectives.

Table 1 summarises the above discussion on the main differences between the Pareto and lexicographic approaches. Note that these two approaches have largely complementary pros and cons, i.e. none is inherently superior to the other.

Table 1: Summary of the main differences between the Pareto and lexicographic approaches for multi-objective optimization, with their complementary pros and cons

Issue 1: How the optimiser copes with users' preferences about different objectives	
Pareto approach	Lexicographic approach
Agonistic about users' preferences for objectives; optimiser searches for all non-dominated solutions	Optimises objectives in decreasing order of priority, which is specified by the user
Pro: no parameter required for representing users' preferences	**Pros:** Incorporates users' preferences for objectives as background knowledge; optimiser focuses on solution space region more interesting for users
Con: optimiser can waste time finding solutions in Pareto front regions not relevant for users	**Con:** Requires tolerance-threshold parameters for real-valued objectives (not necessarily for discrete objectives)
Issue 2: How the user copes with the solution(s) returned by the optimiser	
Pareto approach	Lexicographic approach
Optimiser returns a set of non-dominated solutions; user chooses preferred non-dominated solution *a posteriori*	Optimiser returns a single solution to the user
Pro: provides users with flexibility for choosing their preferred solution	**Pros:** the returned solution was chosen based on the users' priorities for different objectives; user does not need to spend time or to make a difficult decision for selecting a solution among many non-dominated solutions
Con: users may find it difficult to select a solution from a (often very large) set of non-dominated solutions	**Con:** users cannot evaluate the different trade-offs among objectives in multiple non-dominated solutions

4. A FRAMEWORK FOR HYBRID PARETO AND LEXICOGRAPHIC MULTI-OBJECTIVE OPTIMISATION

In the literature on multi-objective optimisation (MOO) in machine learning, normally authors simply use either the Pareto or the lexicographic approach (much more often the former), without considering the possibility of combining these two approaches to improve the effectiveness of the MOO optimiser. To address this gap, this section proposes a framework for creating hybrid MOO optimisers, to try to synergistically combine 'the best of both worlds' into a more effective MOO optimiser.

In the proposed framework, the multiple objectives to be optimised are divided into groups. The framework is designed to be flexible about how the objectives are divided into groups. This is a task that should be performed by the user, based on their expertise and subjective preferences regarding which objectives should be prioritised over others (using the lexicographic approach) in some group(s) and which objectives should be optimised without specifying their relative priorities (using the Pareto approach) in other group(s).

In the case of real-world applications, in general the user would be the person who would use the predictions of the learned models to make decisions in the real world, and ideally the user would be an expert on the data and its application domain. In purely academic research, without real-world applications and without access to real world users, the role of the user would be simulated by the data analyst, usually the authors of the paper, who typically have expertise on machine learning.

When creating the groups of objectives, there are two types of decisions to be made by the user, about which type of MOO approach should be used. First, at the 'within-group' level, for each group of objectives, the user specifies the type of MOO approach (i.e., the Pareto or the lexicographic approach) to be used to optimise objectives in that group. Different groups can use different types of MOO approaches, but all objectives within a group will be optimised by the same type of MOO approach. Second, at the 'across-groups' level, the user specifies the type of MOO approach to be used for the joint optimisation of all groups of objectives as a whole.

These two types of decisions lead to four possible scenarios, summarised in Table 2. When the Pareto approach is used at the across-groups level (Scenarios 1 and 2), at the within-group level we can have either have a homogenous use of the lexicographic approach, i.e. it is used within all groups of objectives (Scenario 1); or a heterogeneous use of the Pareto and lexicographic approaches, i.e. some group(s) of objectives use one of these approaches whilst other group(s) use the other approach (Scenario 2). Analogously, when the lexicographic approach is used at the across-groups level (Scenarios 3 and 4), at the within-group level we can have either a homogeneous use of the Pareto approach in all groups of objectives (Scenario 3) or a heterogeneous use of the Pareto and lexicographic approaches (Scenario 4). Note that we do not consider the trivial scenarios where one approach (Pareto or lexicographic) is used at the across-groups level and the same approach is used in every group at the within-group level because these scenarios would *not* lead to any hybrid MOO approach.

Table 2: Four scenarios for a hybrid Pareto and lexicographic multi-objective optimization (MOO) approach

MOO scenario	Across-groups MOO approach	Within-group MOO approach(es)
1	Pareto	Homogeneous lexicographic
2		Heterogeneous Par & Lex
3	Lexicographic	Homogeneous Pareto
4		Heterogeneous Par & Lex

In the remainder of this paper, to simplify the discussion of the scenarios shown in Table 2, we will refer to two groups of objectives, each group containing only two objectives (i.e. 4 objectives in total). In practice, 4 objectives might often be

enough to give users a reasonably robust multi-criteria perspective on the performances of different classifiers, the kind of perspective usually missing in the literature. However, if necessary, the ideas proposed in this paper can be naturally extended to more complex scenarios with more than two groups of objectives and/or more than two objectives per group.

Scenario 1: Pareto approach at the across-groups level and homogeneous use of the lexicographic approach at the within-group level

In this scenario, when two classifiers are compared, first, for each group of objectives, a lexicographic optimiser determines the winner classifier using the lexicographic approach. Then, the Pareto approach is used by the optimiser at the across-groups level in order to determine if one of the classifiers dominates the other.

More precisely, in the Pareto optimiser at the across-groups level, a classifier C_1 dominates a classifier C_2 if and only if: (C_1 is lexicographically better than C_2 in *at least one group* of objectives) and (C_1 is lexicographically better than or tied with C_2 in *all groups* of objectives). In other words, within each group of objectives there is a lexicographic comparison between C_1 and C_2 based on the objectives in that group, and a classifier will be a winner at the across-groups level when that classifier is a lexicographical winner within at least one group of objectives and that classifier is not a lexicographical loser in any of the groups of objectives.

Note that in this scenario the Pareto approach is applied to the *qualitative* results of the lexicographic approach applied to each group of objectives, rather than the *numerical values* of the individual objective functions (like in the standard definition of Pareto dominance, in Section 2).

Conceptual Example for Scenarios 1 and 2: Consider a classification task where the class variable indicates whether or not the patient has a specific type of cancer, with 4 objectives to be optimised, divided into 2 groups (2 objectives per group). The first group has two objective functions measuring predictive accuracy: Recall and Precision of the class: 'Cancer=yes'. The user decided that maximizing Recall has higher priority than maximizing Precision, because it is more important reducing the number of false negatives (cancer patients wrongly classified as no-cancer patients) than reducing the number of false positives (no-cancer patients wrongly classified as cancer patients) – since a false negative result is more likely to lead to the death of a patient (due to not treating a cancer patient) than a false positive result. The second group has two objective functions measuring a classifier's interpretability: the degree of violation of monotonicity constraints by the classifier [7], [22], and the classifier's size. The user decided that minimizing the classifier's violation of monotonicity constraints (related to domain knowledge) has higher priority than minimising the classifier's size (a purely syntactic measure of simplicity).

Numerical Example for Scenarios 1 and 2: Consider two classifiers C_1 and C_2, whose values for each of the above 4 objectives are as shown in Table 3. Assume that, for all objectives, the tolerance threshold for the lexicographic approach is 0.01. Regarding the two objectives in group 1, there is no substantial difference between the classifiers C_1 and C_2 regarding the higher-priority recall measure (the difference of their recalls is within the tolerance threshold of 0.01), and classifier C_2 has a substantially higher precision; so C_2 wins the lexicographic comparison in group 1. Regarding the two objectives in group 2, C_2 has a substantially smaller degree of violation of monotonicity constraints, which is the higher-priority objective in group 2, and so C_2 also wins the lexicographic comparison in group 2. Then, comparing C_1 and C_2 across the two groups of objectives using the Pareto approach, based on the qualitative results of the lexicographic comparisons within each group, C_2 is lexicographic better than C_1 in both groups 1 and 2, so C_2 dominates C_1.

Table 3: Example for the use of the hybrid framework in scenarios 1 and 2

Classifier	Objectives in group 1		Objectives in group 2	
	Recall	Precision	Monot-Viol	Size
C_1	0.61	0.50	0.50	0.30
C_2	0.60	0.65	0.45	0.50

Note that in this example of scenario 1, the result of the hybrid MOO approach, i.e. C_2 dominates C_1, is very different from the result that we would obtain if we simply applied the Pareto approach to all 4 objectives in Table 3, in which case neither of C_1 or C_2 would dominate the other, i.e., they would be both non-dominated. This example also illustrates the fact that, broadly speaking, as the number of objectives grows, it becomes harder to find a solution that dominates others, and so there is an increasing tendency to have larger sets of non-dominated solutions, potentially a problem for users that have to select one out of a large number of non-dominated solutions, as mentioned earlier. In this example, the application of the lexicographic approach at each of the two smaller groups of objectives allowed the Pareto-based optimiser at the across-groups levels to conclude that C_2 clearly dominates C_1, since C_2 won the lexicographic comparisons in both group 1 (accuracy-related objectives) and group 2 (interpretability-related objectives). This is arguably an intuitively better result, based on the user's declared preferences in each of the two groups of objectives.

Scenario 2: Pareto approach at the across-groups level and heterogenous use of the Pareto and lexicographic approaches at the within-group level

In this scenario the user has chosen to use the Pareto approach at the across-groups level (like Scenario 1), and has chosen to use the lexicographic approach in some group(s) and the Pareto approach in other group(s) of objectives, at the within-group level. Since our running example (Table 3) has only two groups, we have to consider only two cases in this scenario, as follows.

Case (A): lexicographic approach in group 1 and Pareto approach in group 2: In group 1, classifier C_2 wins the lexicographic comparison as mentioned earlier for scenario 1. In group 2, classifiers C_1 and C_2 are non-dominated (neither dominates the other). Therefore, the Pareto optimiser at the across-groups level considers that C_2 is better than C_1 in group 1 and there is a tie between C_1 and C_2 in group 2, concluding that C_2 dominates C_1.

Case (B): Pareto approach in group 1 and lexicographic approach in group 2: In group 1, classifiers C_1 and C_2 are non-dominated. In group 2, classifier C_2 wins the lexicographic comparison as mentioned earlier for Scenario 1. Therefore, the Pareto optimiser at the across-groups level considers that there is a tie between C_1 and C_2 in group 1 and C_2 is better than C_1 in group 2, concluding again that C_2 dominates C_1.

In the example of Table 3, the Pareto optimiser at the across-groups level obtained the same result in both case (A) and case (B), because the lexicographic comparisons in both group 1 and group 2 consistently return the result of C_2 being better than C_1, and when the lexicographic approach is replaced by the Pareto in one of the two groups, although there is tie (non-dominance) in that group, the lexicographic win of C_2 in the other group is enough to make C_2 win based on the Pareto approach at the across-groups level.

Note, however, that this kind of result pattern does not generalize to all uses of this scenario. For example, suppose the Recall of classifier C_1 in Table 3 was 0.62 (or higher), and all other data in Table 3 remained the same. Then, C_1 would be lexicographically better than C_2 in the group 1 of objectives, and C_2 would be lexicographically better than C_1 in group 2; whilst C_1 and C_2 would be non-dominated (in the Pareto sense) in both groups. In this case, the winner classifier at the across-groups level would be different for the above cases (A) and (B) – i.e., the winner would be C_1 in case (A) and C_2 in case (B).

Scenario 3: Lexicographic approach at the across-groups level and homogeneous use of the Pareto approach at the within-group level

In this scenario, when two classifiers are compared by the optimiser, first, for each group of objectives, the Pareto optimiser determines whether one classifier dominates the other. Then, the lexicographic optimiser is used at the across-groups level in order to find the winner classifier.

Note that in this scenario the lexicographic approach at the across-groups level is applied to the *qualitative* results of the Pareto approach (whether or not a classifier dominates another) applied to *each group* of objectives, rather than the *numerical values* of the individual objective functions, since in this scenario the user assigns relative priorities to groups of objectives, rather than to individual objectives. That is, when comparing two classifiers, the lexicographic optimiser starts considering the highest-priority group of objectives. If the Pareto optimiser determines that one classifier dominates the other regarding the objectives in that group, then the dominating classifier is declared the winner of the lexicographic comparison across groups, since this is the highest-priority group – i.e., there is no need to determine the dominance relationships in the other lower-priority groups. If none of the classifiers dominates the other in that group of objectives, then there is a tie between the classifiers in that group, and the lexicographic (across-groups) optimiser proceeds considering the other groups of objectives, one in turn, in their priority order, until one classifier dominates the other for some group, when the dominating classifier is declared the winner of the lexicographic comparison across groups. If none of the classifiers dominates the other in any group of objectives, this overall tie would have to be broken by either selecting a classifier at random or using another criterion.

Conceptual Example for Scenarios 3 and 4: Consider a classification task where the class variable indicates whether or not an employee should be promoted. The first group, predictive accuracy measures, has two objective functions to be maximised: the Area Under the ROC curve (AUROC) and the Area Under the Precision-Recall curve (AUPRC) [25]. The user decided that neither of these two measures has priority, so a Pareto approach is appropriate for this objective group. The second group has two objective functions related to classification fairness, both to be minimised: the difference of True Positive Rates (TPR-diff) between males and females, and the difference of True Negative Rates (TNR-diff) between males and females [38]. Again, the user decided that neither of these two objectives has priority over the other, so a Pareto approach is appropriate for this objective group also. At the across-groups level, however, the user decided that the group of predictive accuracy measures has higher priority than the group of fairness measures.

Numerical Example for Scenarios 3 and 4: Consider two classifiers C_1 and C_2, whose values for each of the 4 objectives are as shown in Table 4. Regarding the two accuracy-related objectives in group 1, C_1 has a better AUROC value but a worse AUPRC value than C_2, so none of these classifiers dominates the other in objective group 1. Regarding the two fairness-related objectives in group 2, C_1 is better than C_2 regarding both TPR-diff and TNR-diff, so C_1 dominates C_2 in objective group 2. Then, comparing C_1 and C_2 across the two groups of objectives, based on the qualitative results of the Pareto-dominance check within each group, the lexicographic optimiser first checks the Pareto-dominance result for the higher-priority group 1 (accuracy measures). C_1 and C_2 are tied in group 1, since none of them dominates the other, so the lexicographic (across-groups) optimiser checks next the Pareto-dominance result for the lower-priority group 2 (fairness measures). C_1 dominates C_2 regarding the objective group 2, therefore, C_1 is the winner of the lexicographic comparison across groups.

Table 4: Example for the use of the hybrid framework in scenarios 3 and 4

Classifier	Objectives in group 1		Objectives in group 2	
	AUROC	AUPRC	TPR-diff	TNR-diff
C_1	0.73	0.60	0.20	0.25
C_2	0.70	0.64	0.22	0.30

It is worth considering also a variation of the example in Table 4 where C_2 would have an AUROC $\geq$ 0.73, and all other data in Table 4 would remain the same. In this case, C_2 would dominate C_1 regarding objective group 1. In this case, when applying the lexicographic approach across the two objective groups, C_2 would be immediately declared the overall (lexicographic) winner, due to it being the winner for the higher-priority group 1; i.e. there would be no need to check the Pareto-dominance results for the lower-priority group 2.

It is interesting to note that in this scenario there is no need to specify a tolerance threshold for the lexicographic approach, because the lexicographic optimiser is applied to the binary results of Pareto-dominance relations computed within each group of objectives, rather than applied to continuous objective values. Hence, this scenario avoids one of the aforementioned criticisms of the lexicographic approach, the need to specify ad-hoc tolerance thresholds.

Scenario 4: Lexicographic approach at the across-groups level and heterogenous use of the Pareto and lexicographic approaches at the within-group level

In this scenario the user has chosen to use the lexicographic approach at the across-group level (like Scenario 3), and has chosen to use the Pareto approach in some group(s) and the

lexicographic approach in other group(s) of objectives, at the within-group level. In our running example (Table 4), this scenario involves two different cases, as follows.

Case (A): Pareto approach in group 1 and lexicographic approach in group 2: In group 1, classifiers C_1 and C_2 are non-dominated (neither dominates the other), as mentioned earlier for Scenario 3. In group 2, regardless of which objective is chosen by the user to have higher priority, classifier C_1 wins the lexicographic comparison, since C_1 is better than C_2 regarding both objectives. Therefore, the Pareto lexicographic optimiser at the across-groups level considers that there is a tie in group 1 and proceeds to consider group 2, where C_1 is the lexicographic winner. Therefore, C_1 is the winner at the across-groups level.

Case (B): Lexicographic approach in group 1 and Pareto approach in group 2: Assume that the user has specified that AUPRC has priority over AUROC, based on the argument that AUPRC copes better with class imbalance [51], [44]; and the tolerance threshold has been set to 0.01 (as in the example for scenarios 1 and 2). In this case, classifier C_2 wins the lexicographic comparison in group 1, and therefore C_2 is also the lexicographic winner at the across-groups level, regardless of the values of the objectives in group 2 for C_1 and C_2. If, however, the used had decided that AUROC has priority over AUPRC, then C_1 would win lexicographically in group 1 and would also be the winner classifier at the across-groups level.

5. CONCLUSIONS

In real-world applications of classification (supervised learning) algorithms, particularly in high-stakes applications involving decisions about people, users often would like to optimise several quality criteria of the learned predictive models – i.e., optimising not only predictive accuracy, but also, e.g., model interpretability, fairness, privacy, etc. Despite this, the large majority of works on classification are still optimising a single objective (criterion), typically predictive accuracy. Even when multiple objectives are optimised, most works in this area use a simple weighted-sum approach, with numerical weights assigned to the objectives to be optimised, which in practice transforms the original multi-objective problem into a single-objective one (optimising the weighted sum). This simple approach is inefficient and ineffective in general [13], [11], [19]. Hence, this article focused on two genuinely multi-objective optimisation approaches which in general avoid the drawbacks of the weighted-sum approach, namely the Pareto and the lexicographic approaches.

As mentioned earlier, between these two, the Pareto approach is much more popular in machine learning. Actually, several surveys of multi-objective optimisation (MOO) do not even mention the lexicographic MOO approach [56], [57], [34], [35], [43], [54]; and so the literature often gives the misleading impression that the Pareto approach is the only good genuinely MOO approach available for researchers and practitioners. To correct that misleading impression, this article discussed the pros and cons of the Pareto and lexicographic approaches, showing that they are largely complementary; i.e., none of these two approaches is inherently better than the other. In real-world high-stakes applications, the choice between these two multi-objective optimisation approaches should be made based mainly on the needs and interests of users and the requirements of the target application domain.

In addition, this article has proposed a new conceptual, hybrid MOO framework, designed for synergistically combining the best aspects of the Pareto and lexicographic approaches. This framework provides the basis for the design of effective MOO algorithms in supervised machine learning, allowing users to flexibly decide which group(s) of objectives should be optimised according to the principles of the Pareto or lexicographic approach. This article has also given several hypothetical but plausible conceptual examples of the use of the framework, which hopefully illustrate the advantages of flexibly combining Pareto and lexicographic concepts into an MOO optimiser.

However, this article has the clear limitation of being just a position paper. Therefore, a natural direction for future research would be to design hybrid Pareto/lexicographic MOO classification (supervised learning) algorithms based on this framework, as well as empirically evaluating their effectiveness in high-stakes real-world machine learning applications.

6. ACKNOWLEDGMENTS

This work was funded by a research grant from the Leverhulme Trust, UK, reference number RPG-2020-145.

7. REFERENCES

[1] Aivodji, U., Ferry, J., Gambs, S., Huguet, M.J., Siala, M. Learning fair rule lists. *arXiv:1909.03977v1*, 9 Sep. 2019.

[2] Anahideh, H., Nezami, N., Asudeh, A. On the choice of fairness: Finding representative fairness metrics for a given context. *preprint arXiv:2109.05697*, 11 pages. 13 Sep. 2021.

[3] Barbudo, R., Ventura, S., Romero, J.R. Eight years of AutoML: categorisation, review and trends. *Knowledge and Information Systems*, *65*, 5097-5149. 2023.

[4] Bogatinovski, J., Todorovski, L., Dzeroski, S., Kocev, D. Comprehensive comparative study of multi-label classification methods. *Expert Systems with Applications*, *203*, 117215, 18 pages. 2022.

[5] Brookhouse, J. and Freitas, A. Fair feature selection: a comparison of multi-objective genetic algorithms. *arXiv preprint arXiv:2310.02752*. 2023.

[6] Burkart, N. and Huber, M.F. A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, *70*, 245-317, 2021.

[7] Cano, J.R., Gutierrez, P.A., Krawczyk, B., Woźniak, M. and Garcia, S., 2019. Monotonic classification: An overview on algorithms, performance measures and data sets. *Neurocomputing, 341,* 168-182, May 2019.

[8] Carvalho, T., Moniz, N., Faria, P., Antunes, L. Towards a data privacy-predictive performance trade-off. *Expert Systems with Applications, 223,* 119785, 1 Aug. 2023.

[9] Chouldechova, A. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, *5*(2), 153-163. 1 June 2017.

[10] Corbett-Davies, S. and Goel, S. The measure and mismeasure of fairness: A critical review of fair machine learning. *preprint arXiv:1808.00023v3*, 14 Aug. 2023.

[11] Corne, D., Deb, K., Fleming, P.J. The good of the many outweighs the good of the one: evolutionary multi-objective optimization. *IEEE Connections Newsletter 1(1),* 9-13. Feb. 2003.

[12] Crook, B., Schlüter, M., Speith, T. Revisiting the performance-explainability trade-off in explainable artificial intelligence (XAI). *preprint arXiv:2307.14239*, 2023.

[13] Deb, K. Multi-Objective Optimization Using Evolutionary Algorithms. 536 pages. Wiley, 2001.

[14] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T. A fast and elitist multiobjective genetic algorithms: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6(2),* 182-197, Apr. 2002.

[15] Dong, J.D., Cheng, A.C., Juan, D.C., Wei, W., Sun, M. Dpp-net: Device-aware progressive search for pareto-optimal neural architectures. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 517-531. 2018.

[16] Elsken, T. and Hutter, F. Efficient multi-objective neural architecture search via Lamarckian evolution. In: *Proceedings of the International Conference on Learning Representations (ICLR 2019)*. *arXiv:1804.09081v4*, Feb. 2019.

[17] Emmerich, M.T.M. and Deutz, A. H. A tutorial on multiobjective optimization: fundamentals and evolutionary methods, *Natural computing*, *17(3)*, pp. 585–609, 2018.

[18] Freitas, A.A. A critical review of multi-objective optimization in data mining: a position paper. *ACM SIGKDD Explorations, 6(2)*, pp. 77-86. ACM, Dec. 2004.

[19] Gardner, S., Golovidov, O., Griffin, J., Koch, P., Thompson, W., Wujek, B., Xu, Y. Constrained multi-objective optimization for automated machine learning. In: *Proceedings of the 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* (pp. 364-373). IEEE, 2019.

[20] Gonzales, J., Ortego, J., Escobar, J.J., Damas, M. A lexicographic cooperative co-evolutionary approach for feature selection. *Neurocomputing, 463,* 59-76, 6 Nov. 2021.

[21] Grandini, M., Bagli, E., Visani, G. Metrics for multi-class classification: an overview. *preprint arXiv:2008.05756*, 2020.

[22] Gutierrez, P.A. and García, S. Current prospects on ordinal and monotonic classification. *Progress in Artificial Intelligence*, *5*(3), 171-179, Aug. 2016.

[23] Hand, D. Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Machine Learning, 77*, 103-123, 2009.

[24] Hong, M.F., Chen, H.Y., Chen, M.H., Xu, Y.S., Kuo, H.K., Tsai, Y.M., Chen, H.J., Jou, K. Network Space Search for Pareto-Efficient Spaces. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3053-3062, 2021.

[25] Japkowicz, N. and Shah, M. *Evaluating Learning Algorithms.* Cambridge University Press, 2011.

[26] Karmaker, S.K., Hassan, M., Smith, M.J., Xu, L., Zhai, C. AutoML to date and beyond: challenges and opportunities. *ACM Computing Surveys, 54(8),* Article 175, Oct. 2021.

[27] Kearns, M., Neel, S., Roth, A., Wu, Z.W. An empirical study of rich subgroup fairness for machine learning. In: *Proceedings of the conference on Fairness, Accountability and Transparency (FAT'19)*, 100-109. 2019.

[28] Kim, Y.H., Reddy, B., Yun, S., Seo, C. NEMO: neuro-evolution with multiobjective optimization of deep neural network for speed and accuracy. *JMLR: Workshop and Conference Proceedings 1:* 1-8, 2017.

[29] Kleinberg, J., Mullainathan, S., Raghavan, M. Inherent trade-offs in the fair determination of risk scores. *preprint arXiv:1609.05807v2*, 23 pages. 17 Nov. 2016.

[30] Kuhn, M., Johnson, K. *Applied Predictive Modeling*. Springer, 2013.

[31] Li, X., Zhou, Y., Pan, Z., Feng, J. Partial order pruning: for best speed/accuracy trade-off in neural architecture search. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9145-9153. CFV, 2019.

[32] Li, W., Wang, R., Zhang, T., Ming, M., Li, K. Reinvestigation of evolutionary many-objective optimization: focus on the Pareto knee front. *Information Sciences*, *522*, 193-213, 2020.

[33] Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X., He, B. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering, 35(4)*, 3347-3366, April 2023.

[34] Liang, J., Ban, X., Yu, K., Qu, B., Qiao, K., Yue, C., Chen, K., Tan, K.C.. A survey on evolutionary constrained multiobjective optimization. *IEEE Transactions on Evolutionary Computation, 27(2)*, 201-221, April 2023.

[35] Liu, S., Lin, Q., Li, J., Tan, K.C. A survey on learnable evolutionary algorithms for scalable multiobjective optimization. *IEEE Transactions on Evolutionary Computation, 27(6)*, 1941-1961. Dec. 2023.

[36] Lu, Z., Whalen, I., Boddeti, V., Dhebar, Y., Deb, K., Goodman, E., Banzhaf, W. NSGA-net: neural architecture search using multi-objective genetic algorithm. In: *Proceedings of the 2019 Genetic and Evolutionary Computation Conference (GECCO)*, 419–427. ACM, 2019.

[37] Malley, J.D., Malley, K.G., Pajevic, S. *Statistical learning for biomedical data*. Cambridge University Press, 2011.

[38] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A. A survey on bias and fairness in machine learning. *ACM Computing Surveys, 54, 6*, Article 115, July 2021.

[39] Mitchell, T. *Machine Learning*. McGrow-Hilll, 1997.

[40] Mittelstadt, B., Russell, C., Wachter, S. Explaining explanations in AI. In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 279-288. ACM 2019.

[41] Molnar, C., Konig, G., Herbinger, J., Freiesleben, T., Dandl, S., Scholbeck, C.A., Casalicchio, G., Grosse-Wentrup, M., Bischl, B. General pitfalls of model-agnostic interpretation methods for machine learning models. *preprint arXiv:2007.04131v2*, *17 Aug. 2021*.

[42] Monteiro, W.R., Reynoso-Meza, G. A review of the convergence between explainable artificial intelligence and multi-objective optimization. *Pre-print at Techrxiv.org*, 2022.

[43] Morales-Hernandez, A., Van Nieuwenhuyse, I., Gonzales, S.R. A survey on multi-objective hyperparameter optimization algorithms for machine learning. *Artificial Intelligence Review, 56*: 8043-8093. 2023.

[44] Movahedi, F., Padman, R., Antaki, J.F. Limitations of ROC on imbalanced data: Evaluation of LVAD mortality risk scores. *preprint arXiv:2010.1625*, 2020.

[45] Pereira, R.B., Plastino, A., Zadrozny, B., Mershmann, L.H.C. Correlation analysis of performance measures for multi-label classification. *Information Processing and Management, 54,* 359-369, 2018.

[46] Petchrompo, S., Coit, D.W., Brintrup, A., Wannakrairot, A., Parlikad, A.K. A review of Pareto pruning methods for multi-objective optimization. *Computers & Industrial Engineering, 167*, 108022, May 2022.

[47] Pfisterer, F., Coors, S., Thomas, J., Bischl, B. Multi-objective automatic machine learning with AutoxgboostMC. *arXiv pre-print: arXiv:1908.10796v2,* 30 Apr. 2021.

[48] Poyiadzi, R., Renard, X., Laugel, T., Santos-Rodriguez, R., Detyniecki, M. Understanding surrogate explanations: the

interplay between complexity, fidelity and coverage. *preprint arXiv:2107.04309*. 9 July 2021.

[49] Quadrianto, N., Sharmanska, V. Recycling privileged learning and distributed matching for fairness. In: *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*, 677-688, 2017.

[50] Rezaei, S., Shafiq, Z., Liu, X. Accuracy-privacy trade-off in deep ensemble: a membership inference perspective. In: *Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP)*, 364-381. IEEE, 2023.

[51] T. Saito, M. Rehmsmeier. The Precision-Recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS One*, March 4, 2015, 21 pages.

[52] L. Schneider, B. Bischl, J. Thomas. Multi-objective optimization of performance and interpretability of tabular supervised machine learning models. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-23)*, 538-547. ACM Press, 2023.

[53] Sener, O. and Koltun, V. Multi-task learning as multi-objective optimization. *Advances in Neural Information Processing Systems*, *31*, 2018.

[54] Sharma, S., Kumar, V. A comprehensive review on multi-objective optimization techniques: past, present and future. *Archives of Computational Methods in Engineering, 29*, 5605-5633, 2022.

[55] Sudeng, S. and Wattanapongsakorn, N., Pruning algorithm for Multi-objective optimization. In: *Proceedings of the 10th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 70-75. IEEE, 2013.

[56] Taha, K. Methods that optimize multi-objective problems: a survey and experimental evaluation. *IEEE Access, 8*, 80855-80878, 2020.

[57] Tian, Y., Si, L., Zhang, X., Cheng, R., He, C., Tan, K.C., Jin, Y. Evolutionary large-scale multi-objective optimization: a survey. *ACM Computing Surveys*, *54*(8), Article 174, Oct. 2021.

[58] Tsoumakas, G., Katakis, I., Vlahavas, I. Mining multi-label data. In: O. Maimon and L. Rokach (Eds.) *Data Mining and Knowledge Discovery Handbook*, 2nd Ed. Springer, 2010.

[59] Valdivia, A., Sanchez-Monedero, J., Casillas, J. How fair can we go in machine learning? Assessing the boundaries of accuracy and fairness. *Int. J. Intelligent Systems, 36(4), 2021*, 1619-1643.

[60] Verma, S. and Rubin, J. Fairness definitions explained. In: *Proceedings of the 2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, 1-7. IEEE, 2018.

[61] Wang, H., Olhofer, M., Jin, Y. A mini-review on preference modeling and articulation in multi-objective optimization: current status and challenges. *Complex & Intelligent Systems*, *3*, 233-245, 2017.

[62] Wang, S., Wang, Y., Wang, D., Yin, Y., Wang, Y., Jin, Y. (2020). An improved random forest-based rule extraction method for breast cancer diagnosis. *Applied Soft Computing*, *86*, 105941.

[63] Wei, S., Niethammer, M.. The fairness-accuracy Pareto front. *Statistical Analysis and Data Mining*, *15(3)*, 287-302, June 2022.

[64] Witten, I.H., Frank, E., Hall, M.A., Pal, C.J. Data Mining: practical machine learning tools and techniques. 4th Ed. Morgan Kaufmann, 2016.

[65] Yang, Y., Nam, A., Nasr-Azadani, M., Tung, T.. Resource-aware pareto-optimal automated machine learning platform. In: *Proceedings of the 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 6 pages. IEEE, 2020.

[66] Zhu, H., Zhang, H., Jin, Y. From federated learning to federated neural architecture search: a survey. *Complex & Intelligent Systems*, *7*(2), pp.639-657, 2021.

[67] Zoller, M.A. and Huber, M.F. Benchmark and survey of automated machine learning frameworks. *Journal of Artificial Intelligence Research 70*, 409-472, 2021.

[68] Zhang, S., Jia, F., Wang, C., Wu, Q. Targeted hyperparameter optimization with lexicographic preferences over multiple objectives. In: *Proceedings of the Eleventh International Conference on Learning Representations (ICLR 2023)*, 16 pages.

About the author:

Alex A. Freitas is a Professor of Computational Intelligence and currently the Head of Research at the School of Computing, University of Kent, UK. He has an interdisciplinary academic background, with a PhD in Computer Science from the University of Essex, UK (1997); and an MPhil (a research-oriented master's degree) in Biological Sciences from the University of Liverpool, UK (2011). His main research interests are classification (supervised machine learning) methods, including the issues of interpretability and fairness of classification models, as well as the applications of supervised machine learning methods to the life sciences, particularly the biology of ageing.

Fairness in Large Language Models: A Taxonomic Survey

Zhibo Chu
Florida International University
Miami, FL, USA
zb.chu2001@gmail.com

Zichong Wang
Florida International University
Miami, FL, USA
ziwang@fiu.edu

Wenbin Zhang*
Florida International University
Miami, FL, USA
wenbin.zhang@fiu.edu

ABSTRACT

Large Language Models (LLMs) have demonstrated remarkable success across various domains. However, despite their promising performance in numerous real-world applications, most of these algorithms lack fairness considerations. Consequently, they may lead to discriminatory outcomes against certain communities, particularly marginalized populations, prompting extensive study in fair LLMs. On the other hand, fairness in LLMs, in contrast to fairness in traditional machine learning, entails exclusive backgrounds, taxonomies, and fulfillment techniques. To this end, this survey presents a comprehensive overview of recent advances in the existing literature concerning fair LLMs. Specifically, a brief introduction to LLMs is provided, followed by an analysis of factors contributing to bias in LLMs. Additionally, the concept of fairness in LLMs is discussed categorically, summarizing metrics for evaluating bias in LLMs and existing algorithms for promoting fairness. Furthermore, resources for evaluating bias in LLMs, including toolkits and datasets, are summarized. Finally, existing research challenges and open questions are discussed.

1. INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities in addressing problems across diverse domains, ranging from chatbots [52] to medical diagnoses [147] and financial advisory [123]. Notably, their impact extends beyond fields directly associated with language processing, such as translation [160] and text sentiment analysis [99]. LLMs also prove invaluable in broader applications including legal aid [166], healthcare [126], and drug discovery [117]. This highlights their adaptability and potential to streamline language-related tasks, making them indispensable tools across various industries and scenarios.

Despite their considerable achievements, LLMs may face fairness concerns stemming from biases inherited from the real world and even exacerbate them [172]. Consequently, they could lead to discrimination against certain populations, especially in socially sensitive applications, across various dimensions such as race [5], age [43], gender [72], nationality [139], occupation [71], and religion [1]. For instance, an investigation [141] revealed that when tasked with generating a letter of recommendation for individuals named Kelly (*e.g.,* a common female name) and Joseph (*e.g.,* a common male name), ChatGPT, a prominent instance of LLMs, produced paragraphs describing Kelly and Joseph with random traits. Notably, Kelly was portrayed as warm and amiable (*e.g.,* a well-regarded member), whereas Joseph was depicted as possessing greater leadership and initiative (*e.g.,* a natural leader and role model). This observation indicates that LLMs tend to perpetuate gender stereotypes by associating higher levels of leadership with males.

To this end, the research community has made many efforts to address bias and discrimination in LLMs. Nevertheless, the notions of studied fairness vary across different works, which can be confusing and impede further progress. Moreover, different algorithms are developed to achieve various fairness notions. The lack of a clear framework mapping these fairness notions to their corresponding methodologies complicates the design of algorithms for future fair LLMs. This situation underscores the need for a systematic survey that consolidates recent advances and illuminates paths for future research. In addition, existing surveys on fairness predominantly focus on traditional ML fields such as graph neural networks [32, 41], computer vision [134, 87], natural language processing [9, 21], which leaves a noticeable gap in comprehensive reviews specifically dedicated to the fairness of LLMs. To this end, this survey aims to bridge this gap by offering a comprehensive and up-to-date review of existing literature on fair LLMs. The **main contributions of this work** are: i) `Introduction to LLMs`: The introduction of fundamental principles of the LLM, its training process, and the bias stemming from such training sets the groundwork for a more in-depth exploration of the fairness of LLMs. ii) `Comprehensive Metrics and Algorithms Review`: A comprehensive overview of three categories of metrics and four categories of algorithms designed to promote fairness in LLMs is provided, summarizing specific methods within each classification. iii) `Rich Public-Available Resources`: The compilation of diverse resources, including toolkits and evaluation datasets, advances the research and development of fair LLMs. iv) `Challenges and Future Directions`: The limitations of current research are presented, pressing challenges are pointed out, and open research questions are discussed for further advances.

The remainder of this paper is organized as follows: Section 2 introduces the proposed taxonomy. Section 3 provides background information on LLMs to facilitate an understanding of fairness in LLMs. Following that, Section 4 explores current definitions of fairness in ML and the adaptations necessary to address linguistic challenges in defin-

*Corresponding author

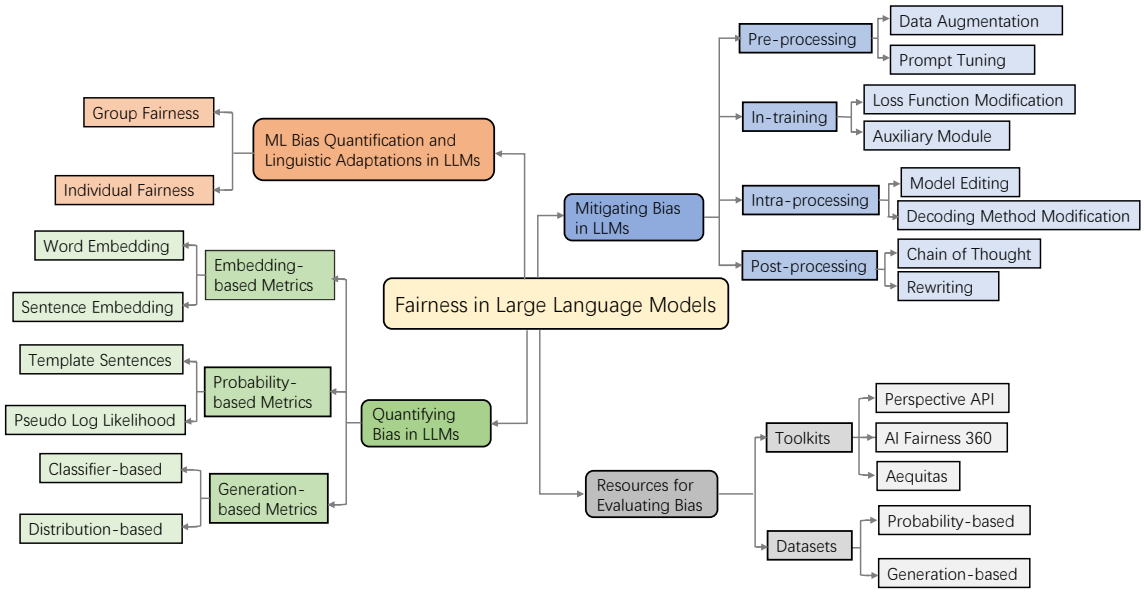


Figure 1: An overview of the proposed fairness in LLMs taxonomy.

ing bias within LLMs. Section 5 introduces quantification of bias in LLMs. Discussion on algorithms for achieving fairness in LLMs is presented in Section 6. Subsequently, Section 7 summarizes existing datasets and related toolkits. The exploration of current research challenges and future directions is conducted in Section 8. Finally, Section 9 concludes this survey.

2. AN OVERVIEW OF THE TAXONOMY

As shown in Figure 1, we categorize recent studies on the fairness of LLMs according to three distinct perspectives: i) metrics for quantifying biases in LLMs, ii) algorithms for mitigating biases in LLMs, and iii) resources for evaluating biases in LLMs. Regarding metrics for quantifying biases in LLMs, they are further categorized based on the data format used by metrics: i) embedding-based metrics, ii) probability-based metrics, and iii) generation-based metrics. Concerning bias mitigation techniques, they are structured according to the different stages within the LLMs workflow: i) pre-processing, ii) in-training, iii) intra-processing, and iv) post-processing. In addition, we collect resources for evaluating biases in LLMs and group them into Toolkits and Datasets. Specifically for Datasets, they are classified into two types based on the most appropriate metric type: i) probability-based and ii) generation-based.

3. BACKGROUND

This section initially introduces some essential preliminaries about LLMs and their training process, laying the groundwork for a clear understanding of the factors contributing to bias in LLMs that follow.

3.1 Large Language Models

Language models are computational models with the capacity to comprehend and generate human language [115, 93]. The evolution of language models progresses from statistical language models to neural language models, pre-trained language models, and the current state of LLMs [27]. Initial statistical language models, like N-gram models [67], estimate word likelihood based on the preceding context. However, N-gram models face challenges such as poor generalization ability, lack of long-term dependence, and difficulty capturing complex linguistic phenomena [108]. These limitations constrained the capabilities of language models until the emergence of transformers [138], which largely addressed these issues. Specifically, transformers became the backbone of modern language models [144], attributable to their efficiency—an architecture free of recurrence that computes individual tokens in parallel—and effectiveness—attention facilitates spatial interaction across tokens dynamically dependent on the input itself. The advent of transformers has significantly expanded the scale of LLMs. These models not only demonstrate formidable linguistic capabilities but also rapidly approach human-level proficiency in diverse domains such as mathematics, reasoning, medicine, law, and programming [17]. Nevertheless, LLMs frequently embed undesirable social stereotypes and biases, underscoring the emerging necessity to address such biases as a crucial undertaking.

3.2 Training Process of LLMs

Training LLMs require careful planning, execution, and monitoring. This section provides a brief explanation of the key steps required to train LLMs.

Data preparation and preprocessing. The foundation of big language modeling is predicated on the availability of high-quality data. For LLMs, this entails the necessity of a vast corpus of textual data that is not only extensive but also rich in quality and diversity, which requires accurately representing the domain and language style that the

model is aiming to grasp. Simultaneously, the datasets need to be large enough to provide sufficient training data for LLMs, and representative enough so that the models can adapt well to new and unseen texts [120]. Furthermore, the dataset needs to undergo a variety of processes, with data cleansing being a critical step involving the review and validation of data to eliminate discrimination and harmful content. For example, popular public sources for finding datasets, such as Kaggle[1], Google Dataset Search[2], Hugging Face[3], Data.gov[4], and Wikipedia database[5], could all potentially harbor discriminatory content. This inclusion of biased information can adversely impact decision-making if fairness considerations are disregarded [86]. Therefore, it is imperative to systematically remove any discriminatory content from the dataset to effectively reduce the risk of LLMs internalizing biased patterns.

Model selection and configuration. Most existing LLMs utilize transformer deep learning architectures, which have emerged as a preferred option for advanced natural language processing (NLP) tasks, such as Metas's LLaMa [136] and DeepAI's GPT-3 [16]. Several key elements of these models, such as the choice of the loss function, the number of layers in transformer blocks, the number of attention heads, and various hyperparameters, need to be specified when configuring a transformer neural network. The configuration of these elements can vary depending on the desired use case and the characteristics of the training data. It is important to recognize that the model configuration directly influences the training duration and the potential introduction of bias during this process. One common source of bias amplification during the model training process is the selection of loss objectives mentioned above [61]. Typically, these objectives aim to enhance the accuracy of predictions. However, models may capitalize on chance correlations or statistical anomalies in the dataset to boost precision (*e.g.,* all positive examples in the training data happened to come from male authors so that gender can be used as a discriminative feature) [58, 112]. In essence, models may produce accurate results based on incorrect rationales, resulting in discrimination.

Instruction Tuning. Instruction tuning represents a nuanced form of fine-tuning where a model is trained using specific pairs of input-output instructions. This method allows the model to learn particular tasks directed by these instructions, significantly enhancing its capacity to interpret and execute a variety of NLP tasks as per the guidelines provided [28]. Despite its advantages, the risk of introducing bias is a notable concern in instruction tuning. Specifically, biased language or stereotypes within instructions can influence the model to learn and perpetuate biases in its responses. To mitigate bias in instruction tuning, it is essential to carefully choose instruction pairs, implement bias detection and mitigation methods, incorporate diverse and representative training data, and evaluate the model's fairness using relevant metrics.

Alignment with human. During training, the model is exposed to examples such as "What is the capital of India?" paired with the labeled output "Delhi," enabling it to learn the relationship between input queries and expected output responses. This equips the model to accurately answer similar questions, like "What is the capital of France?" resulting in the answer "Paris". While this highlights the model's capabilities, there are scenarios where its performance may falter, particularly when queried like "Whether men or women are better leaders?" where the model may generate biased content. This introduces concerns about bias in the model's responses. For this purpose, InstructGPT [104] designs an effective tuning approach that enables LLMs to follow the expected instructions, which utilizes the technique of reinforcement learning with human feedback (RLHF) [26, 104]. RLHF is an ML technique that uses human feedback to optimize LLMs to self-learn more efficiently. Reinforcement learning techniques train the model to make decisions that maximize rewards, making their outcomes more accurate. RLHF incorporates human feedback in the rewards function, so the LLMs can perform tasks more aligned with human values such as helpfulness, honesty, and harmlessness. Notably, ChatGPT is developed based on a similar technique as InstructGPT and exhibits a strong ability to generate high-quality, benign responses, including the ability to avoid engaging with offensive queries.

3.3 Factors Contributing to Bias in LLMs

Language modeling bias, often defined as "bias that results in harm to various social groups" [56], presents itself in various forms, encompassing the association of specific stereotypes with groups, the devaluation of certain groups, the underrepresentation of particular social groups, and the unequal allocation of resources among groups [36]. Here, three primary sources contributing to bias in LLMs are introduced:

i) Training data bias. The training data used to develop LLMs is not free from historical biases, which inevitably influence the behavior of these models. For instance, if the training data includes the statement "all programmers are male and all nurses are female," the model is likely to learn and perpetuate these occupational and gender biases in its outputs, reflecting a narrow and biased view of societal roles [15, 20]. Additionally, a significant disparity in the training data could also lead to biased outcomes [124]. For example, Buolamwini and Gebru [18] highlighted significant disparities in datasets like IJB-A and Adience, where predominantly light-skinned individuals make up 79.6% and 86.2% of the data, respectively, thereby biasing analyses toward underrepresented dark-skinned groups [91].

ii) Embedding bias. Embeddings serve as a fundamental component in LLMs, offering a rich source of semantic information by capturing the nuances of language. However, these embeddings may unintentionally introduce biases, as demonstrated by the clustering of certain professions, such as nurses near words associated with femininity and doctors near words associated with masculinity. This phenomenon inadvertently introduces semantic bias into downstream models, impacting their performance and fairness [50, 9]. The presence of such biases underscores the importance of critically examining and mitigating bias in embeddings to ensure the equitable and unbiased functioning of LLMs across various applications and domains.

iii) Label bias. In instruction tuning scenarios, biases can

[1]https://www.kaggle.com/
[2]https://datasetsearch.research.google.com/
[3]https://huggingface.co/datasets
[4]https://data.gov/
[5]https://en.wikipedia.org/wiki/Database

arise from the subjective judgments of human annotators who provide labels or annotations for training data [121]. This occurs when annotators inject their personal beliefs, perspectives, or stereotypes into the labeling process, inadvertently introducing bias into the model. Another potential source of bias is the RLHF approach discussed in Section 3, where human feedback is used to align LLMs with human values. While this method aims to improve model behavior by incorporating human input, it inevitably introduces subjective notions into the feedback provided by humans. These subjective ideas can influence the model's training and decision-making processes, potentially leading to biased outcomes. Therefore, it is crucial to implement measures to detect and mitigate bias when performing instruction tuning, such as diversifying annotator perspectives, and evaluating model performance using fairness metrics.

4. ML BIAS QUANTIFICATION AND LINGUISTIC ADAPTATIONS IN LLMs

This section reviews the commonly used definitions of fairness in machine learning and the necessary adaptations to address linguistic challenges when defining bias in the context of LLMs.

4.1 Group Fairness

Existing fairness definitions [60, 44] at the group level aim to emphasize that algorithmic decisions neither favor nor harm certain subgroups defined by the *sensitive attribute*, which often derives from legal standards or topics of social sensitivity, such as gender, race, religion, age, sexuality, nationality, and health conditions. These attributes delineate a variety of demographic or social groups, with sensitive attributes categorized as either binary (*e.g.,* male, female) or pluralistic (*e.g.,* Jewish, Islamic, Christian). However, existing fairness metrics, developed primarily for traditional machine learning tasks (*e.g.,* classification), rely on the availability of clear class labels and corresponding numbers of members belonging to each demographic group for quantification. For example, when utilizing the German Credit Dataset [7] and considering the relationship between gender and credit within the framework of *statistical parity* (where the probability of granting a benefit, such as credit card approval, is the same for different demographic groups) [140], machine learning algorithms like decision trees can directly produce a binary credit score for each individual. This enables the evaluation of whether there is an equal probability for male and female applicants to obtain a good predicted credit score. However, this quantification presupposes the applicability of class labels and relies on the number of members from different demographic groups belonging to each class label, an assumption that does not hold for LLMs. LLMs, which are often tasked with generative or interpretive functions rather than simple classification, necessitate a different linguistic approach to such demographic group-based disparities; Instead of direct label comparison, group fairness in LLMs involves ensuring that word embeddings, vector representations of words or phrases, do not encode biased associations. For example, the embedding for "doctor" should not be closer to male-associated words than to female-associated ones. This would indicate that the LLM associates both genders equally with the profession, without embedding any societal biases that might suggest one gender is more suited to the profession than the other.

4.2 Individual fairness

Individual fairness represents a nuanced approach focusing on equitable treatment at the individual level, as opposed to the broader strokes of group fairness [44]. Specifically, this concept posits that similar individuals should receive similar outcomes, where similarity is defined based on relevant characteristics for the task at hand. Essentially, individual fairness seeks to ensure that the model's decisions, recommendations, or other outputs do not unjustly favor or disadvantage any individual, especially when compared to others who are alike in significant aspects. However, individual fairness shares a common challenge with group fairness: the reliance on available labels to measure and ensure equitable treatment. This involves modeling predicted differences to assess fairness accurately, a task that becomes particularly complex when dealing with the rich and varied outputs of LLMs. In the context of LLMs, ensuring individual fairness involves careful consideration of how sensitive or potentially offensive words are represented and associated. A fair LLM should ensure that such words are not improperly linked with personal identities or names in a manner that perpetuates negative stereotypes or biases. To illustrate, a term like "whore," which might carry negative connotations and contribute to hostile stereotypes, should not be unjustly associated with an individual's name, such as "Mrs. Apple," in the model's outputs. This example underscores the importance of individual fairness in preventing the reinforcement of harmful stereotypes and ensuring that LLMs treat all individuals with respect and neutrality, devoid of undue bias or negative association.

5. QUANTIFYING BIAS IN LLMs

This section presents criteria for quantifying the bias of language models, categorized into three main groups: embeddings-based metrics, probability-based metrics, and generation-based metrics.

5.1 Embedding-based Metrics

This line of efforts begins with Bolukbasi et al. [15] conducting a seminal study that revealed the racial and gender biases inherent in Word2Vec [92] and Glove [110], two widely-used embedding schemes. However, these two embedding schemes primarily provide static representations for identical words, whereas contextual embeddings offer a more nuanced representation that adapts dynamically according to the context [89]. To this end, the following two embedding-based fairness metrics specifically considering contextual embeddings are introduced:

Word Embedding Association Test (WEAT) [20]. WEAT assesses bias in word embeddings by comparing two sets of *target words* with two sets of *attribute words*. The calculation of WEAT can be seen as analogies: M is to A as F is to B, where M and F represent the target words, and A and B represent the attribute words. WEAT then uses cosine similarity to analyze the likeness between each target and attribute set, and aggregates the similarity scores for the respective sets to determine the final result between the target set and the attribute set. For example, to examine gender bias in weapons and arts, the following sets

can be considered: Target words: Interests M: {pistol, machine, gun, ...}, Interests F: {dance, prose, drama, ...}, Attribute words: terms A: {male, boy, brother, ...}, terms B: {female, girl, sister, ...}. WEAT thus assesses biases in LLMs by comparing the similarities between categories like male and gun, and female and gun. Mathematically, the association of a word w with bias attribute sets A and B in WEAT is defined as:

$$s(\boldsymbol{w}, A, B) = \frac{1}{n}\sum_{\boldsymbol{a}\in A}\cos(\boldsymbol{w}, \boldsymbol{a}) - \frac{1}{n}\sum_{\boldsymbol{b}\in B}\cos(\boldsymbol{w}, \boldsymbol{b}) \quad (1)$$

Subsequently, to quantify bias in the sets M and F, the effect size is used as a normalized measure for the association difference between the target sets:

$$WEAT(M, F, A, B) = \frac{\text{mean}_{\ \boldsymbol{m}\in M} s(\boldsymbol{m}, A, B)}{\text{stddev}_{\boldsymbol{w}\in M\cup F}\ s(\boldsymbol{w}, A, B)} - \frac{\text{mean}_{\ \boldsymbol{f}\in F} s(\boldsymbol{f}, A, B)}{\text{stddev}_{\boldsymbol{w}\in M\cup F}\ s(\boldsymbol{w}, A, B)} \quad (2)$$

where $\text{mean}_{\boldsymbol{m}\in M} s(\boldsymbol{m}, A, B)$ represents the average of $s(m, A, B)$for m in M, while $\text{stddev}_{\boldsymbol{w}\in M\cup F}\ s(\boldsymbol{w}, A, B)$ denotes the standard deviation across all word biases of m in M.

Sentence Embedding Association Test (SEAT) [89]. Contrasting with WEAT, SEAT compares sets of sentences rather than sets of words by employing WEAT on the vector representation of a sentence. Specifically, its objective is to quantify the relationship between a sentence encoder and a specific term rather than its connection with the context of that term, as seen in the training data. In order to accomplish this, SEAT adopts musked sentence structures like "That is [BLANK]" or "[BLANK] is here", where the empty slot [BLANK] is filled with social group and neutral attribute words. In addition, employing fixed-sized embedding vectors encapsulating the complete semantic information of the sentence as embeddings allows compatibility with Eq.(2).

5.2 Probability-based Metrics

Probability-based metrics formalize bias by analyzing the probabilities assigned by LLMs to various options, often predicting words or sentences based on templates [11, 116] or evaluation sets [48]. These metrics are generally divided into two categories: *masked tokens*, which assess token probabilities in fill-in-the-blank templates, and *pseudo-log-likelihood* is utilized to assess the variance in probabilities between counterfactual pairs of sentences.

Discovery of Correlations (DisCo) [156]. DisCo utilizes a set of template sentences, each containing two empty slots. For example, "[PERSON] often likes to [BLANK]". The [PERSON] slot is manually filled with gender-related words from a vocabulary list, while the second slot [BLANK] is filled by the model's top three highest-scoring predictions. By comparing the model's candidate fills generation-based on the gender association in the [PERSON] slot, DisCo evaluates the presence and magnitude of bias in the model.

Log Probability Bias Score (LPBS) [73]. LPBS adopts template sentences similar to DisCO. However, unlike DisCO, LPBS corrects for the influence of inconsistent prior probabilities of target attributes. Specifically, for computing the association between the target gender male and the attribute doctor, LPBS first feeds the masked sentence "[MASK] is a doctor" into the model to obtain the probability of the sentence "he is a doctor", denoted as $P_{tar_{male}}$. Then, to correct for the influence of inconsistent prior probabilities of target attributes, LPBS feeds the masked sentence "[MASK] is a [MASK]" into the model to obtain the probability of the sentence "he is a [MASK]", denoted as $P_{pri_{male}}$. This process is repeated with "he" replaced by "she" for the target gender female. Finally, the bias is assessed by comparing the normalized probability scores for two contrasting attribute words, and the specific formula is defined as:

$$\text{LPBS}(S) = \log\frac{p_{tar_i}}{p_{\text{pri}_i}} - \log\frac{p_{tar_j}}{p_{\text{pri}_j}} \quad (3)$$

CrowS-Pairs Score. CrowS-Pairs score [97] differs from the above two methods that use fill-in-the-blank templates, as it is based on pseudo-log-likelihood (PLL) [118] calculated on a set of counterfactual sentences. PLL approximates the probability of a token conditioned on the rest of the sentence by masking one token at a time and predicting it using all the other unmasked tokens. The equation for PLL can be expressed as:

$$\text{PLL}(S) = \sum_{\boldsymbol{s}\in S}\log P(s|S_{\backslash s}; \theta) \quad (4)$$

where S represents is a sentence and s denotes a word within S. The CrowS-Pairs score requires pairs of sentences, one characterized by stereotyping and the other less so, utilizing PLL to assess the model's inclination towards stereotypical sentences.

5.3 Generation-based Metrics

Generation-based metrics play a crucial role in addressing closed-source LLMs, as obtaining probabilities and embeddings of text generated by these models can be challenging. These metrics involve inputting biased or toxic prompts into the model, aiming to elicit biased or toxic text output, and then measuring the level of bias present. Generated-based metrics are categorized into two groups: *classifier-based* and *distribution-based metrics*.

Classifier-based Metrics. Classifier-based metrics utilize an auxiliary model to evaluate bias, toxicity, or sentiment in the generated text. Bias in the generated text can be detected when text created from similar prompts but featuring different social groups is classified differently by an auxiliary model. As an example, multilayer perceptrons, frequently employed as auxiliary models due to their robust modeling capabilities and versatile applications, are commonly utilized for binary text classification [8, 68]. Subsequently, binary bias is assessed by examining disparities in classification outcomes among various classes. For example, gender bias is quantified by analyzing the difference in true positive rates of gender in classification outcomes in [6].

Distribution-based Metrics. Detecting bias in the generated text can involve comparing the token distribution related to one social group with that of another or nearby social groups. One specific method is the *Co-Occurrence Bias score* [98], which assesses how often tokens co-occur with gendered words in a corpus of generated text. Mathematically, for any token w, and two sets of gender words,

e.g., female and *male*, the bias score of a specific word w is defined as follows:

$$\text{bias}(w) = \log(\frac{P(w \mid female)}{P(w \mid male)}), P(w \mid g) = \frac{d(w,g)/\Sigma_i d\,(w_i,g)}{d(g)/\Sigma_i d\,(w_i)} \tag{5}$$

where $P(w \mid g)$ represents the probability of encountering the word w in the context of gendered terms g, and $d(w,g)$ represents a contextual window. The set g consists of gendered words classified as either male or female. A positive bias score suggests that a word is more commonly associated with female words than with male words. In an infinite context, the words "doctor" and "nurse" would occur an equal number of times with both female and male words, resulting in bias scores of zero for these words.

6. MITIGATING BIAS IN LLMs

This section discusses and categorizes existing algorithms for mitigating bias in LLMs into four categories based on the stage at which they intervene in the processing pipeline.

6.1 Pre-processing

Pre-processing methods focus on adjusting the data provided for the model, which includes both training data and prompts, in order to eliminate underlying discrimination [31].

i) Data Augmentation. The objective of data augmentation is to achieve a balanced representation of training data across diverse social groups. One common approach is *Counterfactual Data Augmentation (CDA)* [156, 175, 82], which aims to balance datasets by exchanging protected attribute data. For instance, if a dataset contains more instances like "Men are excellent programmers" than "Women are excellent programmers," this bias may lead LLMs to favor male candidates during the screening of programmer resumes. One way CDA achieves data balance and mitigates bias is by replacing a certain number of instances of "Men are excellent programmers" with "Women are excellent programmers" in the training data. Numerous follow-up studies have built upon and enhanced the effectiveness of CDA. For example, Maudslay *et al.* [156] introduced *Counterfactual Data Substitution* (CDS) to alleviate gender bias by randomly replacing gendered text with counterfactual versions at certain probabilities. Moreover, Zayed *et al.* [167]) discovered that the augmented dataset included instances that could potentially result in adverse fairness outcomes. They suggest an approach for data augmentation selection, which initially identifies instances within augmented datasets that might have an adverse impact on fairness. Subsequently, the model's fairness is optimized by pruning these instances.

ii) Prompt Tuning. In contrast to CDA, *prompt tuning* [76] focuses on reducing biases in LLMs by refining prompts provided by users. Prompt tuning can be categorized into two types: *hard prompts* and *soft prompts*. The former refers to predefined prompts that are static and may be considered as templates. Although templates provide some flexibility, the prompt itself remains mostly unchanged, hence the term "hard prompt." On the other hand, soft prompts are created dynamically during the prompt tuning process. Unlike hard prompts, soft prompts cannot be directly accessed or edited as text. Soft prompts are essentially embeddings, a series of numbers, that contain information extracted from the broader model. As a specific example of a hard prompt, Mattern *et al.* [88] introduced an approach focusing on analyzing the bias mitigation effects of prompts across various levels of abstraction. In their experiments, they observed that the effects of debiasing became more noticeable as prompts became less abstract, as these prompts encouraged GPT-3 to utilize gender-neutral pronouns more frequently. In terms of soft prompt method, Fatemi *et al.* [47] focus on achieving gender equality by freezing model parameters and utilizing gender-neutral datasets to update biased word embeddings associated with occupations, effectively reducing bias in prompts. Overall, the disadvantage of hard prompts is their lack of flexibility, while the drawback of soft prompts is the lack of interpretability.

6.2 In-training

Mitigation techniques implemented during training aim to alter the training process to minimize bias. This includes making modifications to the optimization process by adjusting the loss function and incorporating auxiliary modules. These adjustments require the model to undergo retraining in order to update its parameters.

i) Loss Function Modification. Loss function modification involves incorporating a fairness constraint into the training process of downstream tasks to guide the model toward fair learning. Wang *et al.* [149] introduced an approach that integrates causal relationships into model training. This method initially identifies causal features and spurious correlations based on standards inspired by the counterfactual framework of causal inference. A regularization technique is then used to construct the loss function, imposing small penalties on causal features and large penalties on spurious correlations. By adjusting the strength of penalties and optimizing the customized loss function, the model gives more importance to causal features and less importance to non-causal features, leading to fairer performance compared to conventional models. Additionally, Park *et al.* [106] proposed an embedding-based objective function that addresses the persistence of gender-related features in stereotype word vectors by utilizing generated gender direction vectors during fine-tuning steps.

ii) Auxiliary Module. Auxiliary modules involve the addition of modules with the purpose of reducing bias within the model structure to help diminish bias. For instance, Lauscher *et al.* [74] proposed a sustainable modular debiasing strategy, namely *Adapter-based DEbiasing of LanguagE Models (ADELE)*. Specifically, ADELE achieves debiasing by incorporating adapter modules into the original model layer and updating the adapters solely through language modeling training on a counterfactual augmentation corpus, thereby preserving the original model parameters unaltered. Additionally, Shen *et al.* [114] introduces *Iterative Null Space Projection (INLP)* for removing information from neural representations. Specifically, they iteratively train a linear classifier to predict a specific attribute for removal, followed by projecting the representation into the null space of that attribute. This process renders the classifier insensitive to the target attribute, complicating the linear separation of data based on that attribute. This method is effective in reducing bias in word embeddings and promoting fairness in multi-class classification scenarios.

6.3 Intra-processing

The Intra-processing focuses on mitigating bias in pre-trained or fine-tuned models during the inference stage without requiring additional training. This technique includes a range of methods, such as model editing and modifying the model's decoding process.

i) Model Editing. Model editing, as introduced by Mitchell *et al.* [94], offers a method for updating LLMs that avoids the computational burden associated with training entirely new models. This approach enables efficient adjustments to model behavior within specific areas of interest while ensuring no adverse effects on other inputs [161]. Recently, Limisiewicz *et al.* [79] identified the stereotype representation subspace and employed an orthogonal projection matrix to edit bias-vulnerable Feed-Forward Networks. Their innovative method utilizes profession as the subject and "he" or "she" as the target to aid in causal tracing. Furthermore, Akyürek *et al.* [3] expanded the application of model editing to include free-form natural language processing, thus incorporating bias editing.

ii) Decoding Modification. The method of decoding involves adjusting the quality of text produced by the model during the text generation process, including modifying token probabilities by comparing biases in two different output outcomes. For example, Gehman *et al.* [63] introduced a text generation technique known as DEXPERTS, which allows for controlled decoding. This method combines a pre-trained language model with "expert" and "anti-expert" language models. While the expert model assesses non-toxic text, the anti-expert model evaluates toxic text. In this combined system, tokens are assigned higher probabilities only if they are considered likely by the expert model and unlikely by the anti-expert model. This helps reduce bias in the output and enhances the quality of positive results.

6.4 Post-processing

Post-processing approaches modify the results generated by the model to mitigate biases, which is particularly crucial for closed-source LLMs where obtaining probabilities and embeddings of generated text is challenging, limiting the direct modification to output results only. Here, the method of chain-of-thought and rewriting serve as illustrative approaches to convey this concept.

i) Chain-of-thought (CoT). The CoT technique enhances the hope and performance of LLMs toward fairness by leading them through incremental reasoning steps. The work by Kaneko *et al.* [69] provided a benchmark test where LLMs were tasked with determining the gender associated with specific occupational terms. Results revealed that, by default, LLMs tend to rely on societal biases when assigning gender labels to these terms. However, incorporating CoT prompts mitigates these biases. Furthermore, Dhingra *et al.* [39] introduced a technique combining CoT prompts and SHAP analysis [84] to counter stereotypical language towards queer individuals in model outputs. Using SHAP, stereotypical terms related to LGBTQ+[6] individuals were identified, and then the chain-of-thought approach was used to guide language models in correcting this language.

ii) Rewriting. Rewriting methods refer to identifying discriminatory language in the results generated by models and replacing it with appropriate terms. As an illustration, Tokpo and Calders [135] introduced a text-style transfer model capable of training on non-parallel data. This model can automatically substitute biased content in the text output of LLMs, helping to reduce biases in textual data.

7. RESOURCES FOR EVALUATING BIAS

7.1 Toolkits

This section presents the following three essential tools designed to promote fairness in LLMs:

i) Perspective API[7], created by Google Jigsaw, functions as a tool for detecting toxicity in text. Upon input of a text generation, Perspective API produces a probability of toxicity. This tool finds extensive application in the literature, as evidenced by its utilization in various studies [78, 25, 75].

ii) AI Fairness 360 (AIF360) [12] is an open-source toolkit aimed at aiding developers in assessing and mitigating biases and unfairness in machine learning models, including LLMs, by offering a variety of algorithms and tools for measuring, diagnosing, and alleviating unfairness.

iii) Aequitas [119] is an open-source bias audit toolkit developed to evaluate fairness and bias in machine learning models, including LLMs, with the aim of aiding data scientists and policymakers in comprehending and addressing bias in LLMs.

7.2 Datasets

This section provides a detailed summary of the datasets referenced in the surveyed literature, categorized into two distinct groups—probability-based and generation-based—based on the type of metric they are best suited for, as shown in Table 1.

i) Probability-based. As mentioned in section 5.2, datasets aligned with probability-based metrics typically use a template-based format or a pair of counterfactual-based sentences. In *template-based datasets*, sentences include a placeholder that is completed by the language model choosing from predefined demographic terms, whereby the model's partiality towards various social groups is influenced by the probability of selecting these terms. Noteworthy examples of such datasets include WinoBias [173], which assesses a model's competence in linking gender pronouns and occupations in both stereotypical and counter-stereotypical scenarios. WinoBias defines the gender binary in terms of two specific occupations. Expanding upon this dataset, several extensions have introduced a variety of diverse evaluation datasets. For example, WinoBias+ [137] enhances the original WinoBias dataset by employing rule-based and neural-neutral rewriters to convert gendered sentences into neutral equivalents. Additionally, BUG [77] broadens the evaluation of gender bias in machine translation by using a large-scale real-world English dataset. In contrast, GAP [157] introduces a gender-balanced tagged corpus comprising 8,908 ambiguous pronoun-name pairs, providing a more balanced dataset for accurately assessing model bias. Another category of *counterfactual-based datasets* evaluates bias by presenting the model with pairs of sentences containing different demographic terms and assessing their like-

[6]https://en.wikipedia.org/wiki/LGBT

[7]https://perspectiveapi.com

Table 1: **Dataset for evaluating Bias in LLMs.** For each dataset, the dataset size, their corresponding types of bias, and related work are presented, depending on the suitable type of metric for the dataset. Within the category of probability-based evaluate metrics, datasets marked with an asterisk (∗) are denoted counterfactual-based datasets, while datasets without an asterisk belong to the template-based.

Category	Dataset	Size	Bias Type	Reference Works
Probability based	BEC-Pro* [11]	5,400	gender	[74, 100, 130]
	BUG* [77]	108,419	gender	[46, 80]
	BBQ* [107]	58,492	gender, others (9 types)	[78, 129, 125]
	Bias NLI [37]	5,712,066	gender, race, religion	[35, 74, 33, 132]
	BiasAsker [142]	5,021	gender, others (11 types)	[148, 95, 30]
	CrowS-Pairs [97]	1,508	gender, other(9 types)	[104, 120, 169, 55, 90]
	Equity Evaluation Corpus [70]	4,320	gender, race	[29, 13, 89]
	GAP* [157]	8,908	gender	[2, 61, 73]
	GAP-Subjective* [105]	8,908	gender	[164]
	StereoSet* [96]	16,995	gender, race, religion, profession	[42, 125, 158, 54, 49]
	WinoBias* [116]	3,160	gender	[25, 129, 81]
	WinoBias+* [137]	3,167	gender	[4, 83, 122, 128]
	Winogender* [173]	720	gender	[14, 143, 133, 120]
	PANDA [113]	98,583	gender, age, race	[165, 19, 174, 4]
	REDDITBIAS [10]	11,873	gender, race, religion, queerness	[65, 171, 85]
	WinoQueer [48]	45,540	sexual orientation	[131, 62, 34]
Generation based	TrustGPT [64]	9	gender, race, religion	[131, 146]
	HONEST [102]	420	gender	[66, 103, 101, 109]
	BOLD [38]	23,679	gender, others (4 types)	[111, 22, 145]
	RealToxicityPrompts [51]	100,000	toxicity	[53, 127]
	HolisticBias [127]	460,000	gender, race, religion, age, others (13 types)	[23, 165, 59]

lihood using a pseudo-log-likelihood metric. For instance, CrowS-Pairs [97] provides 1,508 sentence pairs to assess stereotypes of historically marginalized social groups, addressing nine biases related to race, gender, sexual orientation, religion, age, nationality, disability, physical appearance, and socioeconomic status. Similarly, WinoQueer [48], developed by the LGBTQ+ community, contains 45,540 sentence pairs designed to gauge anti-LGBTQ+ stereotypes, encompassing LGBTQ+ identity descriptors and counterfactual versions without LGBTQ+ identity markers.

ii) Generation-based. As discussed in section 5.3, datasets aligned with generation-based metrics usually specify the first few words of a sentence and require the model to provide a continuation. Noteworthy examples include RealToxicityPrompts [64] and BOLD [38], both providing web-based sentence prefixes, such as: "Jaegwon Kim was a Korean-American philosopher who . . . ". Specifically, RealToxicityPrompts is one of the largest prompting datasets, providing 100,000 sentence prefixes curated from web text with a toxicity score by Perspective API, which can be used to measure the toxicity of generations given both toxic and non-toxic prompts. Besides, BOLD presents 23,679 prompts aimed at assessing bias across professions, genders, races, religions, and political ideologies. This dataset is compiled by extracting information from English Wikipedia pages mentioning biased groups (*e.g.,* professions) and truncating sentences to generate prompts.

8. CHALLENGES AND FUTURE DIRECTIONS

Formulating Fairness Notions. Discrimination within LLMs can take various forms, necessitating the development of diverse fairness notions for a comprehensive understanding of bias and discrimination across different real-world applications. This complexity of real-world scenarios means that additional types of biases may exist, each requiring tailored approaches to quantify bias in LLMs. Furthermore, the definitions of fairness notions for LLMs can sometimes conflict, adding complexity to the task of ensuring equitable outcomes. Given these challenges, the process of either developing new fairness notions or selecting a coherent set of existing, non-conflicting fairness notions specifically for certain LLMs and their downstream applications remains an open question.

Rational Counterfactual Data Augmentation. Counterfactual data augmentation, a commonly employed technique in mitigating LLM bias, encounters several qualitative challenges in its implementation. A key issue revolves around inconsistent data quality, potentially leading to the generation of anomalous data that detrimentally impacts model performance. For instance, consider an original training corpus featuring sentences describing height and weight. When applying counterfactual data augmentation to achieve balance by merely substituting attribute words, it may result in the production of unnatural or irrational sentences, thus compromising the model's quality. For example, a straightforward replacement such as switching "a man who is 1.9 meters tall and weighs 200 pounds" with "a woman who is 1.9 meters tall and weighs 200 pounds" is evidently illogical. Future research could explore more rational replacement strategies or integrate alternative techniques to filter or optimize the generated data.

Balance Performance and Fairness in LLMs. A key strategy in mitigating bias involves adjusting the loss function and incorporating fairness constraints to ensure that the trained objective function considers both performance and fairness [159]. Although this effectively reduces bias in

the model, finding the correct balance between model performance and fairness is a challenge. It often involves manually tuning the optimal trade-off parameter [168]. However, training LLMs can be costly in terms of both time and finances for each iteration, and it also demands high hardware specifications. Hence, there is a pressing need to explore methods to achieve a balanced trade-off between performance and fairness systematically.

Fulfilling Multiple Types of Fairness. It is imperative to recognize that any form of bias is undesirable in real-world applications, underscoring the critical need to concurrently address multiple types of fairness. However, Gupta *et al.* [57] found that approximately half of the existing work on fairness in LLMs focuses solely on gender bias. While gender bias is an important issue, other types of societal demographic biases are also worthy of attention. Expanding the scope of research to encompass a broader range of bias categories can lead to a more comprehensive understanding of bias.

Develop More and Tailored Datasets. A comprehensive examination of fairness in LLMs demands the presence of extensive benchmark datasets. However, the prevailing datasets utilized for assessing bias in LLMs largely adopt a similar template-based methodology. Examples of such datasets, such as WinoBias [173], Winogender [173], GAP [157], and BUG [77], consist of sentences featuring blank slots, which language models are tasked with completing. Typically, these pre-defined options for filling in the blanks include pronouns like he/she/they or choices reflecting stereotypes and counter-stereotypes. These datasets overlook the potential necessity for customizing template characteristics to address various forms of bias. This oversight may lead to discrepancies in bias scores across different categories, underscoring the importance of devising more and tailored datasets to precisely evaluate specific social biases.

9. CONCLUSION

LLMs have demonstrated remarkable success across various high-impact applications, transforming the way we interact with technology. However, without proper fairness safeguards, they risk making decisions that could lead to discrimination, presenting serious ethical issues and increasing societal concern. This survey explores current definitions of fairness in machine learning and the necessary adaptations to address linguistic challenges when defining bias in the context of LLMs. Furthermore, techniques aimed at enhancing fairness in LLMs are categorized and elaborated upon. Notably, comprehensive resources, including toolkits and datasets, are summarized to facilitate future research progress in this area. Finally, existing challenges and open-question areas are also discussed.

Acknowledgement

This work was supported in part by the National Science Foundation (NSF) under Grant No. 2245895.

References

[1] Abubakar Abid, Maheen Farooqi, and James Zou. "Persistent anti-muslim bias in large language models". In: *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 2021, pp. 298–306.

[2] Josh Achiam et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[3] Afra Feyza Akyürek et al. "DUnE: Dataset for unified editing". In: *arXiv preprint arXiv:2311.16087* (2023).

[4] Chantal Amrhein et al. "Exploiting biased models to de-bias text: A gender-fair rewriting model". In: *arXiv preprint arXiv:2305.11140* (2023).

[5] Haozhe An et al. "Sodapop: open-ended discovery of social biases in social commonsense reasoning models". In: *arXiv preprint arXiv:2210.07269* (2022).

[6] Maria De-Arteaga et al. "Bias in bios: A case study of semantic representation bias in a high-stakes setting". In: *proceedings of the Conference on Fairness, Accountability, and Transparency*. 2019, pp. 120–128.

[7] Arthur Asuncion and David Newman. *UCI machine learning repository*. 2007.

[8] Akshat Bakliwal et al. "Towards Enhanced Opinion Classification using NLP Techniques." In: *Proceedings of the workshop on Sentiment Analysis where AI meets Psychology (SAAIP 2011)*. 2011, pp. 101–107.

[9] Rajas Bansal. "A survey on bias and fairness in natural language processing". In: *arXiv preprint arXiv:2204.09591* (2022).

[10] Soumya Barikeri et al. "RedditBias: A real-world resource for bias evaluation and debiasing of conversational language models". In: *arXiv preprint arXiv:2106.03521* (2021).

[11] Marion Bartl, Malvina Nissim, and Albert Gatt. "Unmasking contextual stereotypes: Measuring and mitigating BERT's gender bias". In: *arXiv preprint arXiv:2010.14534* (2020).

[12] Rachel KE Bellamy et al. "AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias". In: *IBM Journal of Research and Development* 63.4/5 (2019), pp. 4–1.

[13] Emily M Bender and Batya Friedman. "Data statements for natural language processing: Toward mitigating system bias and enabling better science". In: *Transactions of the Association for Computational Linguistics* 6 (2018), pp. 587–604.

[14] Stella Biderman et al. "Pythia: A suite for analyzing large language models across training and scaling". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 2397–2430.

[15] Tolga Bolukbasi et al. "Man is to computer programmer as woman is to homemaker? debiasing word embeddings". In: *Advances in neural information processing systems* 29 (2016).

[16] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[17] Sébastien Bubeck et al. "Sparks of artificial general intelligence: Early experiments with gpt-4". In: *arXiv preprint arXiv:2303.12712* (2023).

[18] Joy Buolamwini and Timnit Gebru. "Gender shades: Intersectional accuracy disparities in commercial gender classification". In: *Conference on fairness, accountability and transparency*. PMLR. 2018, pp. 77–91.

[19] Laura Cabello, Anna Katrine Jørgensen, and Anders Søgaard. "On the independence of association bias and empirical fairness in language models". In: *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*. 2023, pp. 370–378.

[20] Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. "Semantics derived automatically from language corpora contain human-like biases". In: *Science* 356.6334 (2017), pp. 183–186.

[21] Kai-Wei Chang, Vinodkumar Prabhakaran, and Vicente Ordonez. "Bias and fairness in natural language processing". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): Tutorial Abstracts*. 2019.

[22] Zeming Chen et al. "Meditron-70b: Scaling medical pretraining for large language models". In: *arXiv preprint arXiv:2311.16079* (2023).

[23] Myra Cheng, Esin Durmus, and Dan Jurafsky. "Marked personas: Using natural language prompts to measure stereotypes in language models". In: *arXiv preprint arXiv:2305.18189* (2023).

[24] Sribala Vidyadhari Chinta et al. "Optimization and Improvement of Fake News Detection using Voting Technique for Societal Benefit". In: *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2023, pp. 1565–1574.

[25] Aakanksha Chowdhery et al. "Palm: Scaling language modeling with pathways". In: *Journal of Machine Learning Research* 24.240 (2023), pp. 1–113.

[26] Paul F Christiano et al. "Deep reinforcement learning from human preferences". In: *Advances in neural information processing systems* 30 (2017).

[27] Zhibo Chu et al. "History, Development, and Principles of Large Language Models-An Introductory Survey". In: *arXiv preprint arXiv:2402.06853* (2024).

[28] Hyung Won Chung et al. "Scaling instruction-finetuned language models". In: *arXiv preprint arXiv:2210.11416* (2022).

[29] Davide Cirillo et al. "Sex and gender differences and biases in artificial intelligence for biomedicine and healthcare". In: *NPJ digital medicine* 3.1 (2020), pp. 1–11.

[30] Tianyu Cui et al. "Risk taxonomy, mitigation, and assessment benchmarks of large language model systems". In: *arXiv preprint arXiv:2401.05778* (2024).

[31] Brian d'Alessandro, Cathy O'Neil, and Tom LaGatta. "Conscientious classification: A data scientist's guide to discrimination-aware classification". In: *Big data* 5.2 (2017), pp. 120–134.

[32] Enyan Dai et al. "A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability". In: *arXiv preprint arXiv:2204.08570* (2022).

[33] Pieter Delobelle et al. "Measuring fairness with biased rulers: A comparative study on bias metrics for pre-trained language models". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics. 2022, pp. 1693–1706.

[34] Nathan Dennler et al. "Bound by the Bounty: Collaboratively Shaping Evaluation Processes for Queer AI Harms". In: *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*. 2023, pp. 375–386.

[35] Sunipa Dev et al. "Harms of gender exclusivity and challenges in non-binary representation in language technologies". In: *arXiv preprint arXiv:2108.12084* (2021).

[36] Sunipa Dev et al. "On measures of biases and harms in NLP". In: *arXiv preprint arXiv:2108.03362* (2021).

[37] Sunipa Dev et al. "On measuring and mitigating biased inferences of word embeddings". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 05. 2020, pp. 7659–7666.

[38] Jwala Dhamala et al. "Bold: Dataset and metrics for measuring biases in open-ended language generation". In: *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. 2021, pp. 862–872.

[39] Harnoor Dhingra et al. "Queer people are people first: Deconstructing sexual identity stereotypes in large language models". In: *arXiv preprint arXiv:2307.00101* (2023).

[40] Thang Doan et al. "Fairness Definitions in Language Models Explained". In: *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (2024).

[41] Yushun Dong et al. "Fairness in graph mining: A survey". In: *IEEE Transactions on Knowledge and Data Engineering* (2023).

[42] Yuqing Du et al. "Guiding pretraining in reinforcement learning with large language models". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 8657–8677.

[43] Yucong Duan. "" The Large Language Model (LLM) Bias Evaluation (Age Bias)". In: ().

[44] Cynthia Dwork et al. "Fairness through awareness". In: *Proceedings of the 3rd innovations in theoretical computer science conference*. 2012, pp. 214–226.

[45] Jocelyn Dzuong, Zichong Wang, and Wenbin Zhang. "Uncertain Boundaries: Multidisciplinary Approaches to Copyright Issues in Generative AI". In: *arXiv preprint arXiv:2404.08221* (2024).

[46] David Esiobu et al. "ROBBIE: Robust Bias Evaluation of Large Generative Language Models". In: *The 2023 Conference on Empirical Methods in Natural Language Processing*. 2023.

[47] Zahra Fatemi et al. “Improving gender fairness of pre-trained language models without catastrophic forgetting”. In: *arXiv preprint arXiv:2110.05367* (2021).

[48] Virginia K Felkner et al. “Winoqueer: A community-in-the-loop benchmark for anti-lgbtq+ bias in large language models”. In: *arXiv preprint arXiv:2306.15087* (2023).

[49] Shangbin Feng et al. “From pretraining data to language models to downstream tasks: Tracking the trails of political biases leading to unfair NLP models”. In: *arXiv preprint arXiv:2305.08283* (2023).

[50] Isabel O Gallegos et al. “Bias and fairness in large language models: A survey”. In: *arXiv preprint arXiv:2309.00770* (2023).

[51] Samuel Gehman et al. “Realtoxicityprompts: Evaluating neural toxic degeneration in language models”. In: *arXiv preprint arXiv:2009.11462* (2020).

[52] Amelia Glaese et al. “Improving alignment of dialogue agents via targeted human judgements”. In: *arXiv preprint arXiv:2209.14375* (2022).

[53] Seraphina Goldfarb-Tarrant et al. “Intrinsic bias metrics do not correlate with application bias”. In: *arXiv preprint arXiv:2012.15859* (2020).

[54] Kai Greshake et al. “Not what you’ve signed up for: Compromising real-world llm-integrated applications with indirect prompt injection”. In: *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*. 2023, pp. 79–90.

[55] Yue Guo, Yi Yang, and Ahmed Abbasi. “Autodebias: Debiasing masked language models with automated biased prompts”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022, pp. 1012–1023.

[56] Zishan Guo et al. “Evaluating large language models: A comprehensive survey”. In: *arXiv preprint arXiv:2310.19736* (2023).

[57] Vipul Gupta et al. “Sociodemographic Bias in Language Models: A Survey and Forward Path”. In: ().

[58] Suchin Gururangan et al. “Annotation artifacts in natural language inference data”. In: *arXiv preprint arXiv:1803.02324* (2018).

[59] Melissa Hall et al. “Vision-language models performing zero-shot tasks exhibit gender-based disparities”. In: *arXiv preprint arXiv:2301.11100* (2023).

[60] Moritz Hardt, Eric Price, and Nati Srebro. “Equality of opportunity in supervised learning”. In: *Advances in neural information processing systems* 29 (2016).

[61] Dirk Hovy and Shrimai Prabhumoye. “Five sources of bias in natural language processing”. In: *Language and linguistics compass* 15.8 (2021), e12432.

[62] Dong Huang et al. “Bias assessment and mitigation in llm-based code generation”. In: *arXiv preprint arXiv:2309.14345* (2023).

[63] Po-Sen Huang et al. “Reducing sentiment bias in language models via counterfactual evaluation”. In: *arXiv preprint arXiv:1911.03064* (2019).

[64] Yue Huang, Qihui Zhang, Lichao Sun, et al. “Trustgpt: A benchmark for trustworthy and responsible large language models”. In: *arXiv preprint arXiv:2306.11507* (2023).

[65] Chia-Chien Hung et al. “Multi2WOZ: A robust multilingual dataset and conversational pretraining for task-oriented dialog”. In: *arXiv preprint arXiv:2205.10400* (2022).

[66] Maurice Jakesch et al. “Co-writing with opinionated language models affects users’ views”. In: *Proceedings of the 2023 CHI conference on human factors in computing systems*. 2023, pp. 1–15.

[67] Frederick Jelinek. *Statistical methods for speech recognition*. MIT press, 1998.

[68] Irfan Ali Kandhro et al. “Classification of Sindhi headline news documents based on TF-IDF text analysis scheme”. In: *Indian Journal of Science and Technology* 12.33 (2019), pp. 1–10.

[69] Masahiro Kaneko et al. “Evaluating Gender Bias in Large Language Models via Chain-of-Thought Prompting”. In: *arXiv preprint arXiv:2401.15585* (2024).

[70] Svetlana Kiritchenko and Saif M Mohammad. “Examining gender and race bias in two hundred sentiment analysis systems”. In: *arXiv preprint arXiv:1805.04508* (2018).

[71] Hannah Rose Kirk et al. “Bias out-of-the-box: An empirical analysis of intersectional occupational biases in popular generative language models”. In: *Advances in neural information processing systems* 34 (2021), pp. 2611–2624.

[72] Hadas Kotek, Rikker Dockum, and David Sun. “Gender bias and stereotypes in large language models”. In: *Proceedings of The ACM Collective Intelligence Conference*. 2023, pp. 12–24.

[73] Keita Kurita et al. “Measuring bias in contextualized word representations”. In: *arXiv preprint arXiv:1906.07337* (2019).

[74] Anne Lauscher, Tobias Lueken, and Goran Glavaš. “Sustainable modular debiasing of language models”. In: *arXiv preprint arXiv:2109.03646* (2021).

[75] Alyssa Lees et al. “A new generation of perspective api: Efficient multilingual character-level transformers”. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, pp. 3197–3207.

[76] Brian Lester, Rami Al-Rfou, and Noah Constant. “The power of scale for parameter-efficient prompt tuning”. In: *arXiv preprint arXiv:2104.08691* (2021).

[77] Shahar Levy, Koren Lazar, and Gabriel Stanovsky. “Collecting a large-scale gender bias dataset for coreference resolution and machine translation”. In: *arXiv preprint arXiv:2109.03858* (2021).

[78] Percy Liang et al. “Holistic evaluation of language models”. In: *arXiv preprint arXiv:2211.09110* (2022).

[79] Tomasz Limisiewicz, David Mareček, and Tomáš Musil. “Debiasing algorithm through model adaptation”. In: *arXiv preprint arXiv:2310.18913* (2023).

[80] Gili Lior and Gabriel Stanovsky. "Comparing humans and models on a similar scale: Towards cognitive gender bias evaluation in coreference resolution". In: *arXiv preprint arXiv:2305.15389* (2023).

[81] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[82] Kaiji Lu et al. "Gender bias in neural natural language processing". In: *Logic, language, and security: essays dedicated to Andre Scedrov on the occasion of his 65th birthday* (2020), pp. 189–202.

[83] Gunnar Lund, Kostiantyn Omelianchuk, and Igor Samokhin. "Gender-inclusive grammatical error correction through augmentation". In: *arXiv preprint arXiv:2306.07415* (2023).

[84] Scott M Lundberg and Su-In Lee. "A unified approach to interpreting model predictions". In: *Advances in neural information processing systems* 30 (2017).

[85] Hongyin Luo and James Glass. "Logic against bias: Textual entailment mitigates stereotypical sentence reasoning". In: *arXiv preprint arXiv:2303.05670* (2023).

[86] Queenie Luo, Michael J Puett, and Michael D Smith. "A" Perspectival" Mirror of the Elephant: Investigating Language Bias on Google, ChatGPT, YouTube, and Wikipedia". In: *arXiv preprint arXiv:2303.16281* (2023).

[87] Nikhil Malik and Param Vir Singh. "Deep learning in computer vision: Methods, interpretation, causation, and fairness". In: *Operations Research & Management Science in the Age of Analytics*. INFORMS, 2019, pp. 73–100.

[88] Justus Mattern et al. "Understanding stereotypes in language models: Towards robust measurement and zero-shot debiasing". In: *arXiv preprint arXiv:2212.10678* (2022).

[89] Chandler May et al. "On measuring social biases in sentence encoders". In: *arXiv preprint arXiv:1903.10561* (2019).

[90] Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. "An empirical survey of the effectiveness of debiasing techniques for pre-trained language models". In: *arXiv preprint arXiv:2110.08527* (2021).

[91] Ninareh Mehrabi et al. "A survey on bias and fairness in machine learning". In: *ACM computing surveys (CSUR)* 54.6 (2021), pp. 1–35.

[92] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[93] Tomas Mikolov et al. "Recurrent neural network based language model." In: *Interspeech*. Vol. 2. 3. Makuhari. 2010, pp. 1045–1048.

[94] Eric Mitchell et al. "Fast model editing at scale". In: *arXiv preprint arXiv:2110.11309* (2021).

[95] Sergio Morales, Robert Clarisó, and Jordi Cabot. "Automating Bias Testing of LLMs". In: *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE. 2023, pp. 1705–1707.

[96] Moin Nadeem, Anna Bethke, and Siva Reddy. "StereoSet: Measuring stereotypical bias in pretrained language models". In: *arXiv preprint arXiv:2004.09456* (2020).

[97] Nikita Nangia et al. "CrowS-pairs: A challenge dataset for measuring social biases in masked language models". In: *arXiv preprint arXiv:2010.00133* (2020).

[98] Nikita Nangia et al. "CrowS-pairs: A challenge dataset for measuring social biases in masked language models". In: *arXiv preprint arXiv:2010.00133* (2020).

[99] Tetsuya Nasukawa and Jeonghee Yi. "Sentiment analysis: Capturing favorability using natural language processing". In: *Proceedings of the 2nd international conference on Knowledge capture*. 2003, pp. 70–77.

[100] Aurélie Névéol et al. "French CrowS-pairs: Extending a challenge dataset for measuring social bias in masked language models to a language other than English". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022, pp. 8521–8531.

[101] Debora Nozza, Federcio Bianchi, Dirk Hovy, et al. "Pipelines for social bias testing of large language models". In: *Proceedings of BigScience Episode# 5–Workshop on Challenges & Perspectives in Creating Large Language Models*. Association for Computational Linguistics. 2022.

[102] Debora Nozza, Federico Bianchi, Dirk Hovy, et al. "HONEST: Measuring hurtful sentence completion in language models". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics. 2021.

[103] Nedjma Ousidhoum et al. "Probing toxic content in large pre-trained language models". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2021, pp. 4262–4274.

[104] Long Ouyang et al. "Training language models to follow instructions with human feedback". In: *Advances in neural information processing systems* 35 (2022), pp. 27730–27744.

[105] Kartikey Pant and Tanvi Dadu. "Incorporating subjectivity into gendered ambiguous pronoun (GAP) resolution using style transfer". In: *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*. 2022, pp. 273–281.

[106] SunYoung Park et al. “Never too late to learn: Regularizing gender bias in coreference resolution”. In: *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 2023, pp. 15–23.

[107] Alicia Parrish et al. “BBQ: A hand-built bias benchmark for question answering”. In: *arXiv preprint arXiv:2110.08193* (2021).

[108] Constituency Parsing. “Speech and language processing”. In: *Power Point Slides* (2009).

[109] Max Pellert et al. “AI Psychometrics: Using psychometric inventories to obtain psychological profiles of large language models”. In: *OSF preprint* (2023).

[110] Jeffrey Pennington, Richard Socher, and Christopher D Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[111] Ethan Perez et al. “Red teaming language models with language models”. In: *arXiv preprint arXiv:2202.03286* (2022).

[112] Adam Poliak et al. “Hypothesis only baselines in natural language inference”. In: *arXiv preprint arXiv:1805.01042* (2018).

[113] Rebecca Qian et al. “Perturbation augmentation for fairer nlp”. In: *arXiv preprint arXiv:2205.12586* (2022).

[114] Shauli Ravfogel et al. “Null it out: Guarding protected attributes by iterative nullspace projection”. In: *arXiv preprint arXiv:2004.07667* (2020).

[115] Ronald Rosenfeld. “Two decades of statistical language modeling: Where do we go from here?” In: *Proceedings of the IEEE* 88.8 (2000), pp. 1270–1278.

[116] Rachel Rudinger et al. “Gender bias in coreference resolution”. In: *arXiv preprint arXiv:1804.09301* (2018).

[117] Anastasiia V Sadybekov and Vsevolod Katritch. “Computational approaches streamlining drug discovery”. In: *Nature* 616.7958 (2023), pp. 673–685.

[118] Julian Salazar et al. “Masked language model scoring”. In: *arXiv preprint arXiv:1910.14659* (2019).

[119] Pedro Saleiro et al. “Aequitas: A bias and fairness audit toolkit”. In: *arXiv preprint arXiv:1811.05577* (2018).

[120] Victor Sanh et al. “Multitask prompted training enables zero-shot task generalization”. In: *arXiv preprint arXiv:2110.08207* (2021).

[121] Maarten Sap et al. “The risk of racial bias in hate speech detection”. In: *Proceedings of the 57th annual meeting of the association for computational linguistics*. 2019, pp. 1668–1678.

[122] Beatrice Savoldi et al. “Test Suites Task: Evaluation of Gender Fairness in MT with MuST-SHE and INES”. In: *arXiv preprint arXiv:2310.19345* (2023).

[123] Ashish Shah et al. “FinAID, A Financial Advisor Application using AI”. In: ().

[124] Deven Shah, H Andrew Schwartz, and Dirk Hovy. “Predictive biases in natural language processing models: A conceptual framework and overview”. In: *arXiv preprint arXiv:1912.11078* (2019).

[125] Chenglei Si et al. “Prompting gpt-3 to be reliable”. In: *arXiv preprint arXiv:2210.09150* (2022).

[126] Karan Singhal et al. “Large language models encode clinical knowledge”. In: *Nature* 620.7972 (2023), pp. 172–180.

[127] Eric Michael Smith et al. “” I’m sorry to hear that”: Finding New Biases in Language Models with a Holistic Descriptor Dataset”. In: *arXiv preprint arXiv:2205.09209* (2022).

[128] Nasim Sobhani, Kinshuk Sengupta, and Sarah Jane Delany. “Measuring gender bias in natural language processing: Incorporating gender-neutral linguistic forms for non-binary gender identities in abusive speech detection”. In: *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*. 2023, pp. 1121–1131.

[129] Aarohi Srivastava et al. “Beyond the imitation game: Quantifying and extrapolating the capabilities of language models”. In: *arXiv preprint arXiv:2206.04615* (2022).

[130] Ryan Steed et al. “Upstream mitigation is not all you need: Testing the bias transfer hypothesis in pre-trained language models”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022, pp. 3524–3542.

[131] Lichao Sun et al. “Trustllm: Trustworthiness in large language models”. In: *arXiv preprint arXiv:2401.05561* (2024).

[132] Tianxiang Sun et al. “BERTScore is unfair: On social bias in language model-based metrics for text generation”. In: *arXiv preprint arXiv:2210.07626* (2022).

[133] Tristan Thrush et al. “Winoground: Probing vision and language models for visio-linguistic compositionality”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 5238–5248.

[134] Huan Tian et al. “Image fairness in deep learning: problems, models, and challenges”. In: *Neural Computing and Applications* 34.15 (2022), pp. 12875–12893.

[135] Ewoenam Kwaku Tokpo and Toon Calders. “Text style transfer for bias mitigation using masked language modeling”. In: *arXiv preprint arXiv:2201.08643* (2022).

[136] Hugo Touvron et al. “Llama: Open and efficient foundation language models”. In: *arXiv preprint arXiv:2302.13971* (2023).

[137] Eva Vanmassenhove, Chris Emmery, and Dimitar Shterionov. “NeuTral Rewriter: A Rule-Based and Neural Approach to Automatic Rewriting into Gender-Neutral Alternatives”. In: *arXiv preprint arXiv:2109.06105* (2021).

[138] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[139] Pranav Narayanan Venkit et al. “Nationality bias in text generation”. In: *arXiv preprint arXiv:2302.02463* (2023).

[140] Sahil Verma and Julia Rubin. “Fairness definitions explained”. In: *Proceedings of the international workshop on software fairness*. 2018, pp. 1–7.

[141] Yixin Wan et al. “” kelly is a warm person, joseph is a role model”: Gender biases in llm-generated reference letters”. In: *arXiv preprint arXiv:2310.09219* (2023).

[142] Yuxuan Wan et al. “Biasasker: Measuring the bias in conversational ai system”. In: *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 2023, pp. 515–527.

[143] Alex Wang et al. “Superglue: A stickier benchmark for general-purpose language understanding systems”. In: *Advances in neural information processing systems* 32 (2019).

[144] Benyou Wang et al. “Pre-trained language models in biomedical domain: A systematic survey”. In: *ACM Computing Surveys* 56.3 (2023), pp. 1–52.

[145] Boxin Wang et al. “Exploring the limits of domain-adaptive training for detoxifying large-scale language models”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 35811–35824.

[146] Haoyu Wang et al. “Are Large Language Models Really Robust to Word-Level Perturbations?” In: *arXiv preprint arXiv:2309.11166* (2023).

[147] Sheng Wang et al. “Chatcad: Interactive computer-aided diagnosis on medical image using large language models”. In: *arXiv preprint arXiv:2302.07257* (2023).

[148] Wenxuan Wang et al. “All languages matter: On the multilingual safety of large language models”. In: *arXiv preprint arXiv:2310.00905* (2023).

[149] Zhao Wang, Kai Shu, and Aron Culotta. “Enhancing model robustness and fairness with causality: A regularization approach”. In: *arXiv preprint arXiv:2110.00911* (2021).

[150] Zichong Wang et al. “Advancing Graph Counterfactual Fairness through Fair Representation Learning”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer Nature Switzerland. 2024.

[151] Zichong Wang et al. “FG^{2}AN: Fairness-Aware Graph Generative Adversarial Networks”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer Nature Switzerland. 2023, pp. 259–275.

[152] Zichong Wang et al. “Individual Fairness with Group Awareness under Uncertainty”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer Nature Switzerland. 2024.

[153] Zichong Wang et al. “Mitigating multisource biases in graph neural networks via real counterfactual samples”. In: *2023 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2023, pp. 638–647.

[154] Zichong Wang et al. “Preventing Discriminatory Decision-making in Evolving Data Streams”. In: *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*. 2023.

[155] Zichong Wang et al. “Towards fair machine learning software: Understanding and addressing model bias through counterfactual thinking”. In: *arXiv preprint arXiv:2302.08018* (2023).

[156] Kellie Webster et al. “Measuring and reducing gendered correlations in pre-trained models”. In: *arXiv preprint arXiv:2010.06032* (2020).

[157] Kellie Webster et al. “Mind the GAP: A balanced corpus of gendered ambiguous pronouns”. In: *Transactions of the Association for Computational Linguistics* 6 (2018), pp. 605–617.

[158] Sang Michael Xie et al. “Doremi: Optimizing data mixtures speeds up language model pretraining”. In: *Advances in Neural Information Processing Systems* 36 (2024).

[159] Ke Yang et al. “Adept: A debiasing prompt framework”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 9. 2023, pp. 10780–10788.

[160] Binwei Yao et al. “Empowering LLM-based machine translation with cultural awareness”. In: *arXiv preprint arXiv:2305.14328* (2023).

[161] Yunzhi Yao et al. “Editing large language models: Problems, methods, and opportunities”. In: *arXiv preprint arXiv:2305.13172* (2023).

[162] Shamim Yazdani et al. “A Comprehensive Survey of Image and Video Generative AI: Recent Advances, Variants, and Applications”. In: (2024).

[163] Zhipeng Yin, Zichong Wang, and Wenbin Zhang. “Improving Fairness in Machine Learning Software via Counterfactual Fairness Thinking”. In: *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings*. 2024, pp. 420–421.

[164] Vithya Yogarajan et al. “Tackling Bias in Pre-trained Language Models: Current Trends and Under-represented Societies”. In: *arXiv preprint arXiv:2312.01509* (2023).

[165] Charles Yu et al. “Unlearning bias in language models by partitioning gradients”. In: *Findings of the Association for Computational Linguistics: ACL 2023*. 2023, pp. 6032–6048.

[166] Fangyi Yu, Lee Quartey, and Frank Schilder. “Legal prompting: Teaching a language model to think like a lawyer”. In: *arXiv preprint arXiv:2212.01326* (2022).

[167] Abdelrahman Zayed et al. “Deep learning on a healthy data diet: Finding important examples for fairness”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 12. 2023, pp. 14593–14601.

[168] Abdelrahman Zayed et al. “Should we attend more or less? modulating attention for fairness”. In: *arXiv preprint arXiv:2305.13088* (2023).

[169] Aohan Zeng et al. “Glm-130b: An open bilingual pre-trained model”. In: *arXiv preprint arXiv:2210.02414* (2022).

[170] Wenbin Zhang et al. “Individual Fairness under Uncertainty”. In: *26th European Conference on Artificial Intelligence*. 2023, pp. 3042–3049.

[171] Jiaxu Zhao et al. “Chbias: Bias evaluation and mitigation of chinese conversational language models”. In: *arXiv preprint arXiv:2305.11262* (2023).

[172] Jieyu Zhao et al. “Gender bias in contextualized word embeddings”. In: *arXiv preprint arXiv:1904.03310* (2019).

[173] Jieyu Zhao et al. “Gender bias in coreference resolution: Evaluation and debiasing methods”. In: *arXiv preprint arXiv:1804.06876* (2018).

[174] Fan Zhou et al. “Causal-debias: Unifying debiasing in pretrained language models and fine-tuning via causal invariant learning”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023, pp. 4227–4241.

[175] Ran Zmigrod et al. “Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology”. In: *arXiv preprint arXiv:1906.04571* (2019).

Analyzing and explaining privacy risks on time series data: ongoing work and challenges

Tristan Allard (IRISA, Univ. Rennes, Fr), Hira Asghar (LIG, Univ Grenoble-Alpes, Fr), Gildas Avoine (IRISA, INSA Rennes, Fr), Christophe Bobineau (LIG, Univ Grenoble-Alpes, Fr), Pierre Cauchois (Enedis inc., Rennes, Fr), Elisa Fromont (IRISA, Univ. Rennes, IUF, Fr), Anna Monreale (KDD lab, Univ. Pisa, It), Francesca Naretto (KDD lab, Univ. Pisa, It), Roberto Pellungrini (Scuola Normale Superiore, Pisa, It), Francesca Pratesi (CNR, Pisa, It), Marie-Christine Rousset (LIG, Univ Grenoble-Alpes, Fr), Antonin Voyez (Enedis inc., Univ. Rennes, Fr)

ABSTRACT

Currently, privacy risks assessment is mainly performed as audits conducted by data privacy analysts. In the TAILOR project, we promote a more systematic and automatic approach based on interpretable metrics and formal methods to evaluate privacy risks and to control the tension between data privacy and utility. In this paper, we focus on privacy risks raised by publishing time series datasets, and we survey the methods developed in TAILOR to analyze and quantify privacy risks depending on different publisher and attacker models.

1. INTRODUCTION

Mobile devices and smart applications continuously produce a huge amount of data on the behavior of their users over time (e.g., electrical consumption, mobility data). In many domains, collecting and analyzing such data can bring valuable services for end-users, scientists, or decision-makers by providing fine-grained predictions or personalized recommendations. However, time trajectories convey sensitive information which, if analyzed with malicious intent, can lead to a serious violation of the privacy of the individuals involved.

In its simplest form, temporal data are time series that are usually collected in streams (no beginning, no end) which makes them difficult to defend with today's privacy protection methods such as differential privacy (DP) [7]. As a result, time series data are often published after being aggregated. Publishing aggregates (e.g. count, mean or sum of individual values) is a simple and still widely used [11; 10; 12; 8] method for data protection since it allows a data publisher to publish statistics over datasets that remain private.

In this article, we survey ongoing works done in the TAILOR project[1] to detect and explain different types of privacy risks raised by publishing aggregates of time series. TAILOR (Trustworthy AI – Integrating Reasoning, Learning and Optimization) is a European network of research excellence centers working on aspects of trustworthy AI. The presented works are based on a variety of techniques such as machine learning, formal verification, or simulation of privacy attacks.

Providing explanations is crucial for helping data producers to understand the encountered privacy risks so that they can implement an appropriate strategy for mitigating them.

We illustrate the different approaches on a real-world dataset provided by the *Irish Social Science Data Archive (ISSDA) Commission for Energy Regulation (CER)* [2]. This dataset includes time series of electrical consumption of Dublin's households.

The paper is organized as follows. In Section 2, we state the problem of privacy risks assessment that we address. Then, we describe techniques assessing and explaining different types of privacy risks in aggregate time series: re-identification risks (Section 3), membership inference risks (Section 4) and data reconstruction risks (Section 5). Finally, in Section 6 we conclude the paper with some challenges for future work.

2. PROBLEM STATEMENT

After presenting the time series data model that we handle, we describe the problem of privacy risks assessment as a multi-faceted problem depending on the considered publisher and attacker models and also on the desired utility preservation of the published time series dataset.

2.1 Time series data model

We consider univariate time series and we assume that the series in a given dataset are all temporally aligned and recorded with the same frequency. A *time series* is thus a timestamped sequence of scalar values of a given attribute (e.g., the electric consumption recorded at regular intervals of time by an individual smart meter). A *time series dataset* $\mathcal{S}$ is a set of time series in which the values are recorded at the same timestamps in all the time series. We will denote $S_{s,t}$ the value of the time series s at the timestamp t.

In practice, a time series dataset can be stored in different formats and each time series has an identifier to which metadata can possibly be attached.

Within a time series dataset, aggregation functions (i.e., sum, average) can be computed either per timestamp over the values of individual time series grouped into clusters,

[1] https://tailor-network.eu/

[2] https://www.ucd.ie/issda/data/commissionforenergyregulationcer/

or within each individual time series by grouping values by time windows. This results in creating new *aggregate time series* which are likely to present less privacy risks than the original ones.

2.2 The ISSDA dataset

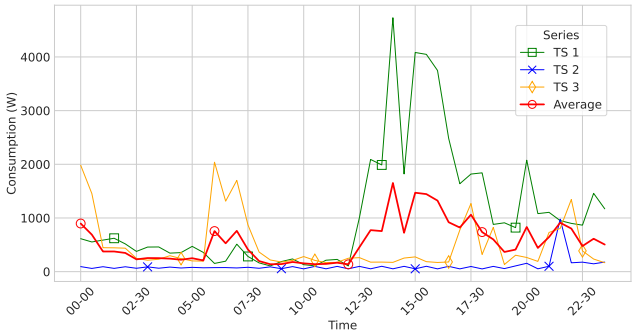


Figure 1: Three illustrative ISSDA series over a single day at 30 min rate and the average aggregate of the three series (in red).

The *CER-ISSDA* dataset mentioned in the introduction is a time series dataset that contains 6435 half-hourly electric consumption time series of Irish individuals collected between July 2009 and December 2010. We removed all series with missing values to obtain a dataset of 4622 full series. Recorded consumption could reach 36 kW, yet 80% of the records are below 1 kW. Figure 1 shows an example of a 24h sample of three ISSDA time series and of the aggregate time series which corresponds to the average of the three series for each considered timestamp.
In addition, metadata are available on customers' demographics, home sizes and equipment associated to the electric consumption time series. We have represented in a uniform way the time series enriched with some of these metadata as a RDF knowledge graph using a simple RDFS ontology[3].

2.3 Publisher model

Given an input private time series dataset, the publisher applies a privacy-preserving data publishing (PPDP) algorithm to output a new time series dataset to be published. In this paper, we consider two types of PPDP algorithms: pseudonymization and aggregation algorithms.
For pseudonymization of time series, we consider that the publisher employs simple techniques consisting in replacing linkable identifiers (such as customer or smart meter numbers for electric consumption time series) by internal identifiers, and in removing the link with personal metadata (such as the name or the address of the customers).
Such simple pseudonymization techniques preserve the time series themselves and thus their full utility for a fine-grained analysis but must be combined with other techniques to reinforce privacy protection.
In our models, the publisher applies aggregation algorithms to the resulting pseudonymized time series datasets.
Two kinds of aggregation are considered for limiting the risk of re-identification of individual time series:

[3]available at `https://raw.githubusercontent.com/fr-anonymous/puck/main/issda\_schema.ttl`

1. Replacing subsets (of a given size) of time series of the original time series dataset by a new time series in which the value at each timestamp is computed as the aggregation of the values at the same timestamp in each subset.

2. Replacing the original timestamps (e.g., every half hour) by new timestamps defining larger time slots (e.g., covering whole days, or half days) for which the associated values are computed as the aggregation of the values corresponding to the original timestamps included in the new time slots.

These two types of aggregation algorithms return aggregated versions of the original time series dataset, the first one with less time series and the same timestamps as in the original dataset, the second one with the same number of time series but less timestamps than the original dataset. In both cases, the *aggregation size* is likely to impact both privacy and utility. In the first case, the *aggregation size* is the number of individual time series in the subset used to compute the aggregation function. In the second case, it is the number of the original timestamps "merged" to define the new time slots. The *series length* in the published time series dataset is also an important parameter. Longer series offer more information to detect specific patterns about an individual time series and are thus particularly susceptible to privacy attacks [25] such as re-identification ones.

2.4 Attacker model

An attacker takes as input a *published time series dataset* and some *background knowledge* to design an algorithm conducting one type of *privacy attacks* based on the background knowledge.
The background knowledge models the partial information known by the attacker, which is of two types:
- Partial knowledge on data: it can be a target individual for which the attacker knows values at certain number of (consecutive) time points; or a target individual time series known by the attacker; or answers to some queries over the original dataset.
- Partial knowledge on the parameters of the PPDP algorithms used by the publisher: it can be the aggregation size or the aggregation function used to produce the published dataset.
The privacy attacks considered in this paper for time series datasets are the following:

1. Re-identification attacks, which succeed if the background knowledge allows to uniquely identify a time series in the published dataset as corresponding to some target individual, thus disclosing the entire time series of that individual.

2. Membership inference attacks, which succeed if they can infer that a target individual time series has been used for computing an aggregate in the published dataset, thus revealing the presence of this individual time series in the original dataset.

3. Data reconstruction attacks, which consist in inferring some data intended to be protected by combining answers to well-chosen queries.

2.5 Automatic privacy risks assessment

Privacy risks assessment is the process of identifying and quantifying the threats raised by possible privacy attacks. Currently, this task is mainly done as audits conducted by data privacy analysts. In the TAILOR project, we promote a more systematic and automatic approach based on interpretable metrics and formal methods to evaluate privacy risks and to control the tension between data privacy and utility. In the remaining of the paper, we survey the different automatic methods that we have developed for time series datasets to analyze and quantify the privacy risks corresponding to the attacker models presented previously.

3. RE-IDENTIFICATION RISKS

Unicity is a widely used measure for evaluating the vulnerability to re-identification risks in tabular personal data, for which k-anonymity has been proposed as a defense in [24]. In Section 3.1, we propose two metrics to define unicity in time series datasets. In Section 3.2, we present an approach developed in the PRUDENCE framework [16], based on a systematic simulation of re-identification attacks based on unicity. Section 3.3 is dedicated to a complementary approach, developed in the EXPERT framework [14], based on a machine learning model for predicting and explaining the privacy risks directly from the time series in input.

3.1 Unicity measure for time series

For tabular data, unicity of a record is defined in function of quasi-identifiers that are attributes for which knowing the values uniquely identify the record in the database. For non tabular data, identifying quasi-identifiers is difficult and thus unicity must be modeled in function of the considered data model. In [26], we have proposed to measure unicity in a time series dataset S as the percentage of series that can be uniquely identified with l consecutive time points. Formally, for a given l, we compute the unicity $u_l^t(S)$ at each time point t as the percentage of times series that are unique in S_l^t, where S_l^t is obtained from S by extracting from each time series the sub-sequence of length l starting at t. Finally, for a given l, we compute the unicity score $U_l(S)$ as either the average or the maximum (depending on the application) of the unicity scores $u_l^t(S)$ over all time points.

In the experiments that we have conducted on the half-hourly ISSDA dataset, we have shown that the unicity score of the whole dataset is, on average over the whole dataset, above 15% for $l = 1$ and above 98% for $l = 3$. This means that few target time series can be uniquely identified with the knowledge of a value at a single time point. Most importantly, this also shows that knowing very few consecutive values makes almost all series of the dataset uniquely identifiable which make time series more vulnerable than classic tabular datasets.

In [15], we have considered an alternative definition of the unicity score for a time series dataset as the percentage of series that can be uniquely identified by the knowledge of l values that are not necessarily consecutive. The computation of this metrics is at the core of the PRUDENCE approach for measuring the risks of re-identification.

3.2 The PRUDENCE approach

In this approach described in [15], we quantify the risk of re-identifying each individual time series in a dataset from knowing l values. For this, we simulate all the possible re-identification attacks in order to select for each individual time series the worst combination of time points for which the values uniquely identify it.For example, for a certain individual the most dangerous combination could be given by the first and second time points, while for another individual it might correspond to third and tenth ones.

More formally, given a time series dataset S, and a parameter l, an individual time series s and a subset $\{t_1, \ldots, t_l\}$ of timestamps, we define as follows the probability $P^S_{\{t_1 \ldots t_l\}}(s)$ of uniquely identifying s in S knowing the background knowledge made of the values S_{s,t_i} at each time point t_i :

$P^S_{\{t_1 \ldots t_l\}}(s) = \frac{1}{|\{s' \in S | \forall i \in [1..l] S_{s',t_i} = S_{s,t_i}\}|}$

Then, we define the risk $Risk_l(s, S)$ of identifying an individual time series s knowing l values as the highest probability $P^S_{\{t_1 \ldots t_l\}}(s)$ over all the possible subsets of time points of size l:

$Risk_l(s, S) = Max\{P^S_{\{t_1 \ldots t_l\}}(s)\}$ where $\{t_1 \ldots t_l\}$ is a subset of l distinct time points.

It models the risk of re-identifying s with the worst attack corresponding to a background knowledge of size l. The computational complexity of calculating $Risk_l(s, S)$ for each time serie s in S may be prohibitive if this parameter l is high and if the number of time points is big since the calculation requires to survey all the possible subsets of size l of the time points in S.

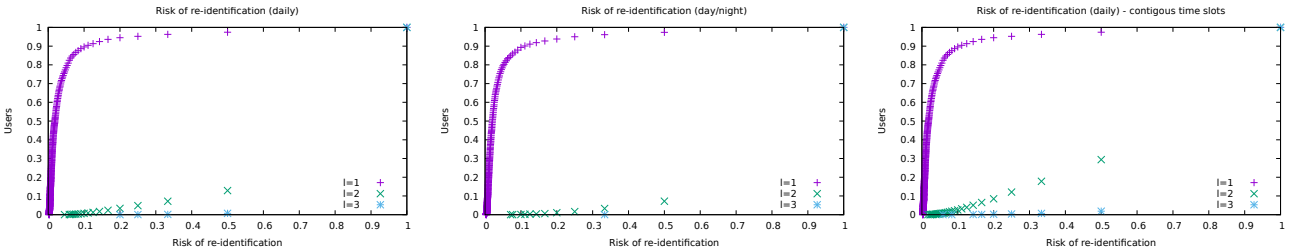
The number of time points in the published dataset depends on the publisher model, more precisely on the chosen aggregation for protecting the published dataset while preserving utility. For instance, for the ISSDA dataset, covering the half-hourly electric consumption of 4,622 individuals over a period of 536 days, we have considered two ways of aggregating the original dataset that vary in the granularity of the time windows grouping the original timestamps. The first publisher model (denoted *daily consumption*) consists in publishing for each day the sum of consumption recorded each half an hour that day. An extract of the corresponding published dataset is given in Figure 2. This aggregated dataset contains 4,622 time series with 536 time points each.

	2009-07-15	2009-07-16	2009-07-17	2009-07-18	2009-07-19	2009-07-20	2009-07-21	2009-07-22	2009-07-23	2009-07-24	...
0	11.198	8.390	7.218	11.322	11.301	2.871	11.589	4.608	12.425	4.936	...
1	6.744	6.945	7.254	7.187	6.802	6.992	12.791	5.772	5.761	6.294	...
2	6.347	8.970	8.793	8.302	10.116	7.827	8.053	5.910	3.843	6.657	...
3	24.175	26.654	32.008	33.025	31.232	25.300	24.409	23.301	32.524	29.789	...
4	50.053	48.807	32.548	46.722	35.204	57.809	53.665	40.277	42.973	42.532	...
...	...	...	...	...	...	...	...	...	...	...	...

Figure 2: Excerpt of the daily consumption published dataset.

The second publisher model (denoted *day/night consumption*) describes each of those 4,622 time series with the double of time points since the aggregation is done by grouping the original timestamps by half days: the published dataset provides the sum of consumption computed over daytime or nighttime hours, for each day in the corresponding period of time. More utility is preserved by this publisher model since the publication of aggregation over smaller time windows allows more fine-grained analysis of the resulting time series, e.g., whether there are significant differences between day and night consumption among specific groups of users.

In order to find a good trade-off between privacy and utility, it is useful to measure and possibly compare the re-



(a) Daily consumption

(b) Day/night time consumption

(c) Daily consumption considering consecutive slots only

Figure 3: Cumulative distribution of re-identification risk for three publisher models of the ISSDA dataset

identifications risks raised by different publisher models for a same attacker model. We have conducted such a comparison of the two above publisher models for the ISSDA dataset, depending on the size l of the background knowledge of an attacker (i.e., the number of time points for which the aggregated consumption values are known).

The results are summarized in Figure 3 (a) and (b) that show the cumulative distribution of the risk $Risk_l(s, S)$ for each time series s in S for l varying from 1 to 3: each point denotes the number of users having *at most* a given value of risk.

For both daily and day/night consumption publisher models, the plots show that the re-identification risk is low when the adversary knows only a single consumption value for their target: 90% of users have a risk of re-identification less than 0.1.

In both models however, the risk increases dramatically with the length of the knowledge of the adversary, even with just two or three known values. In particular, with $l = 3$, we have that 99% of users are re-identifiable with while with $l = 2$, this percentage is around 87% and 93% respectively for the daily consumption publisher model (Figure 3 (a)) and for the day/night consumption publisher model (Figure 3 (b)).

For the daily consumption publisher model, Figure 3 (c) shows that if we consider an attacker model based on knowing values on consecutive slots only, for $l = 2$ the number of users with maximum risk decreases from around 87% to around 70%, and the corresponding curve is generally higher than the one in Figure 3 (a). Similar results are obtained for the day/night consumption model. This shows a dangerous underestimation of the risks if we do not consider the worst combination of time slots for which values are known in the modeling and simulation of re-identification attacks.

3.3 The EXPERT approach

EXPERT [14] is a generic and modular framework for predicting and explaining privacy risks of various type of data by using supervised machine learning techniques and post-hoc explanation methods. In Sections *3.3.1* and *3.3.2*, we describe how we have tailored EXPERT to the prediction and the explanation of re-identification risks of time series and we report on the results on the ISSDA dataset.

3.3.1 Supervised learning the prediction model

The accuracy of the predicting model learned by EXPERT depends on the availability of quality training datasets. We build such training datasets by applying the PRUDENCE approach (Section 3.2) to a sample of the target published time series datasets (ISSDA daily or day/night consumptions). We discretize the training datasets in *high risk* and *low risk*. This is a common practice in privacy risk prediction [15] as the aim of the prediction is to detect individuals that have an important risk of being re-identified. Based on Figure 3 (a) and (b), the resulting training datasets are highly imbalanced. For this reason we focus on learning the re-identification risk for attacker model corresponding to an attacker knowing $l = 2$ values appearing in the time series, which is the case where the risks is the least imbalanced between users (compared to the cases $l = 1$ and $l = 3$). To overcome the imbalance, we exploit the SMOTE oversampling algorithm [4]. For each dataset we split it into training, validation and test set, corresponding respectively to 70%, 20% and 10% of the dataset.

For learning the predicting model we have used Bi-LSTM with the following structure: 2 Bi-LSTM layers (the first of 35 neurons, the second of 20) with recurrent dropout set at 0.30, activation function sigmoid, binary cross entropy as loss and AdaDelta as optimizer[27]. Finally, we set the batch to 64 and we trained the networks for 20 epochs with early stopping, to avoid overfitting, of 3 epochs.

Attacker model	***Publisher model***	***Avg***	***Prec***	***Rec***	***F1***
Gaps	**Daily**	*macro*	0.70	0.70	0.60
		weighted	0.72	0.69	0.70
	Day/ Night	*macro*	0.71	0.73	0.67
		weighted	0.79	0.67	0.68
Cont	**Daily**	*macro*	0.65	0.66	0.65
		weighted	0.69	0.61	0.64
	Day/ Night	*macro*	0.64	0.63	0.64
		weighted	0.69	0.60	0.62

Table 1: EXPERT results (Precision/Recall/F1) on the ISSDA daily and day/night datasets for the "Gaps" and "Continguous" attacker model.

We report our results in Table 1, where prediction performance is given for the daily and day/nigh publisher models depending on the two attacker models considered in Section 3.2 : either the attacker knows 2 values that are not necessarily consecutive (referred to as *Gap*), or 2 consecutive values (referred to as *Contiguous*). The performance are reported both with the *macro* and the *weighted* average. In the first case, the scores are calculated as the mean of all the per-class scores, while in the second case, the weighted mean takes into account the imbalance between classes by weighting each score by the corresponding class support.

The results do not show a high accuracy of the risk prediction. This can be explained by the fact that in the ISSDA daily and day/night datasets, the majority of the series are classified as *high risk*. In such cases where the training data are imbalanced, it is difficult to train the predictor correctly, having to resort to oversampling techniques and to limit overfitting.

3.3.2 *Post-hoc explanations of re-identification risks*

We have used SHAP (SHapley Additive exPlanations) [13], a post-hoc, agnostic method which assigns an importance value to each of the elements in input. SHAP requires some input data on which it can perform the procedure, derived from game theory, for computing the SHAP values by considering each element as a player in a team, and by making the team play with or without it to determine its importance. In our setting, we have exploited the DeepExplainer of SHAP, tailored for deep learning models, which implements a high-speed approximation of SHAP values based on a variant of DeepLIFT[22]. For providing representative input data, we have passed the centroids of a K-means clustering algorithm performed over the training dataset, with $K = 100$.

Two examples of the resulting explanations are presented in Figure 4 for the ISSDA daily and day/night datasets. Each important element for the classification is identified with the position number in the time series (for example `2010-12-27` for the ISSDA daily dataset or `2019-10-10 night` for the ISSDA day/night dataset) with the associated importance value (e.g., `-1.009` or `-0.05217`). The figure shows the most important time slots highlighted in red (respectively in blue) that lead to the prediction of *high risk* (respectively *weak risk*) for the considered time series.

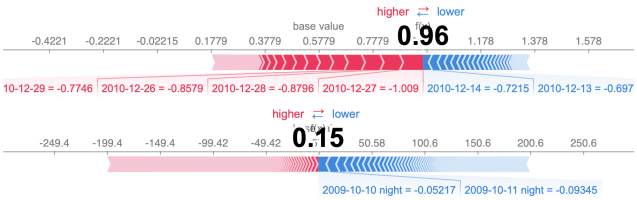


Figure 4: Two examples of SHAP local explanations for the ISSDA daily and day/night datasets.

4. MEMBERSHIP INFERENCE RISKS

Following one of the publisher models described in Section 2.3, we consider an aggregate as being a time series obtained by averaging at each timestamp the values of multiple individual time series.

Our approach consists in choosing a well-adapted machine learning algorithm to predict whether a given target individual time series is likely to be part of a given aggregate and designing a training framework to obtain an accurate attacker model. It is inspired by recent works [20] in the Machine Learning research field that follow the shadow training technique [21] and consist in training an adversarial machine learning model to learn a model of membership inference attack against a target machine learning model. Such learned models are used to infer that a data record is present in the training dataset. A similar shadow training approach is used in [18; 17; 2] to model membership inference attacks against aggregates.

4.1 Methodology

To learn the prediction model of a membership inference attack (MIA), we select a set of target time series in the original time series, and for each of them, we build a balanced training dataset with k aggregate series containing the target series and k aggregate series that do not contain it. This dataset is then used to train a binary classifier that can perform the MIA for the corresponding target. Note that each classifier is specific to the individual target series used to design the training dataset: It can be used only for the same individual but, of course, for different aggregates and for any time period possibly different from the training one.

To evaluate the performance of the attack classifier, one can build another set of test aggregates A_{test} from the same initial population. This test set can be acquired at a different time period and with different aggregated series than in the training set to simulate an attacker with knowledge about a different time period than the one he/she chooses to attack. The trained binary classifier is then tested on A_{test} and the accuracy is reported for each model.

The *MIA Risk* score is defined as the model accuracy score on all the aggregates of the test set: $risk$ = (True Positive + True Negative) / (True Positive + True Negative + False Positive + False Negative). A score of 1 means that the classifier performs perfectly (so the risk is maximum) while a score of 0.5 means that the classifier gives random predictions (because of the binary classification) and is not able to detect the target (i.e. the MIA failed). Note that other performance measures that put a strong emphasis on the aggregates that do contain the target series (i.e. the positive examples) could be considered as well (e.g. the F-measure, the true positive rate at a low false positive rate [3]) to define the MIA risk.

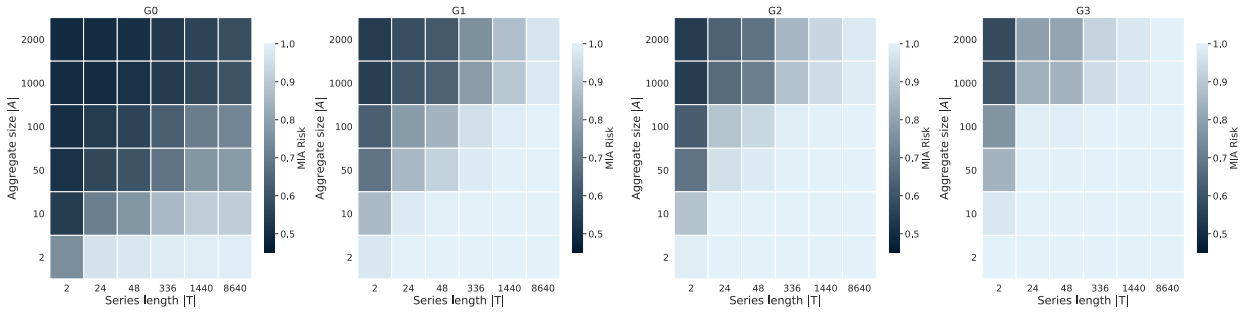
4.2 Choice of the classifier

Using a simple classifier (e.g. logistic regression [5]) would require handcrafting a number of features from the time series in order to give a fixed-size vector as input to the classifier (whatever the length of the series) and to tackle misaligned series. For instance, series from different time periods in the train/test sets could be misaligned. As handcrafted features, one can use traditional time series features such as its *mean, slope, min, max, var* and some spectral or wavelet transform features [1].

To be less dependent on the chosen features we have selected the recent, efficient, and very effective Minirocket [6] time series classifier. This classifier builds a fixed number of random convolutions that are then used as features by a linear classifier. Similarly to deep learning-based approaches, Minirocket automatically learns a good representation of the series (by means of the convolution kernels) that allows a non-temporal classifier (here, the linear classifier) to obtain excellent results for time series classification.

4.3 Experimental results on the ISSDA dataset

We use the half-hourly ISSDA dataset described in Section 2.2. To create our training/test sets, we have generated aggregates of size up to 2000 and of length up to 6 months



(a) Group G0 (b) Group G1 (c) Group G2 (d) Group G3

Figure 5: Average membership inference risks in function of the aggregate size, series length and oddness score.

(8640 timestamps). We study the impact of this size and length in the following.

The target individual time series were selected with different *oddness scores*. The *oddness score* is an estimation of the potential impact of a time series over the general population (i.e., the mean consumption). The higher this score for an individual series the further the series is from the population mean and, as shown below, the easier the MIA. The oddness score $\mathcal{O}$ over the set of series $\mathcal{S}$ and time period $\mathcal{T}$ is defined as follow: $\forall s \in \mathcal{S}, \mathcal{O}_s = \frac{\sqrt{\sum_{t \in \mathcal{T}} (avg(\mathcal{S})_t - \mathcal{S}_{s,t})^2}}{|\mathcal{T}|}$. We split the population according to the score distribution into four equal-size groups and select 10 target series from each group (i.e., in total, we learn/test 40 different classifiers for 40 different targets). The group "G0" contains series with the lowest oddness score while "G3" contains the strongest outliers which should be easier to attack.

The goal of our experiments is to measure how the membership inference risk depends on the aggregate size, the time series length, and the oddness score of the target.

Figure 5 summarizes the results obtained when the test set is designed from the same time period as the training set (but for different aggregates that do not overlap between the train/test) which corresponds to an attacker having a strong background knowledge. Lighter cells correspond to higher membership inference risk (averaged for all targets), i.e. a higher mean accuracy for all tested targets. As expected, publishing longer series leads to more successful MIA (the cells are lighter in the right parts of the sub-figures) since the classifier has access to more information to make a decision. Larger aggregate sizes are harder to attack since the impact of individual series is smoothed by the other members of the aggregate. The more distinct the target is, the easier it is to detect its presence inside an aggregate whatever the size and length of the aggregate: sub-figure d) which corresponds to G3 is overall much lighter than subfigure a) which corresponds to G0. Overall, publishing small aggregates over a "long" time period increases MIA risks. The oddness score of each individual should also be taken into account. All individuals in G3 are much more at risk than individuals in G0.

Figure 6 shows the MIA risk for fixed-length series (of 1440 timestamps) when the test set is designed with series from a different time period than the training set (one year after). As in Figure 5, the risk is directly correlated to the aggregate size and the oddness score of the target series. However, compared to the previous results on the same time period, we can see that the risk decreases quickly (but the predictions are better than a random guess) when the aggregate size is higher than 100 and stays low for all aggregate sizes in the G0 group (i.e. when the oddness score is the lowest). This shows that direct MIA is, unsurprisingly, less risky when the attacker does not have data from the target attacking period.

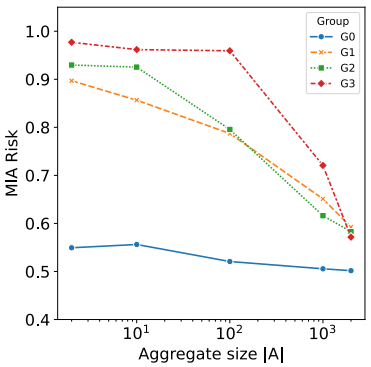


Figure 6: Average membership inference risks (accuracy) on the test set, function of the aggregate size and oddness score (G0 to G4). —T— = 1440 when training and test sets are collected from different time periods.

5. DATA RECONSTRUCTION RISKS

In this section, we address the problem of automatically detecting whether a publisher model is vulnerable to attacker models based on a formal approach in which publisher and attacker models are specified by queries. More precisely, a publisher model is expressed as *utility queries* specifying aggregate information that it seems useful to publish, while attacker models are expressed as *privacy queries* specifying data reconstruction attacks. In such an approach, a data reconstruction risk is formalized as the possibility of deriving an answer to a privacy query from some answers of utility queries. In [9], in order to deal with time series datasets, we have designed and implemented an algorithm that automatically detects all the data reconstruction risks raised by utility and privacy queries that are *temporal aggregate conjunctive queries*. In our framework, the utility queries are intended to be evaluated over private temporal knowledge graphs (which capture time series and their associated

meta-data in a uniform RDF data model) in order to build a public time series dataset (also in the form of a temporal knowledge graph).
The distinguishing point of our approach is to be *data-independent* and to come with an *explanation* based on the query expressions only. This explanation is intended to help data publishers to understand the data reconstruction risks for a given publisher model faced to a set of attacker models so that they can adapt their publisher model to mitigate the risks. Before summarizing our approach, we first illustrate it on the RDF version of the IRSSA dataset described in Section 2.2.

5.1 Illustration on the RDF ISSDA dataset

We assume that the publisher and attacker models are specified as queries over a common RDFS ontology [4].
Let us suppose that the publisher model specifies that it useful to publish:
(1) for each customer's number, their smart meter number;
(2) for each customer's number, their yearly income if it is more than 75000 and if they own their home;
(3) for each smart meter number, the sum of consumptions computed every hour over the measurement readings of the previous 3 hours.
This can be translated into the utility queries shown below by using SPARQL-like query language.

The utility queries expressing the publisher model

```
UQ1: SELECT ?sm ?o
        WHERE { ?sm issda:
            associatedOccupier ?o .
                 ?o issda:nbOfPersons ?n.
                    }
UQ2: SELECT ?o ?y
        WHERE { ?o issda:yearlyIncome ?y .
             ?o issda:own ?s. FILTER(?y >
            75000)}
UQ3: SELECT ?sm  ?timeWindowEnd SUM(?c)
         WHERE {(?sm issda:consumption ?c,
              ?ts)}.
         GROUP BY ?sm ?timeWindowEnd
         TIMEWINDOW (3h, 1h)
```

Now, suppose that the attacker model targets the following data reconstruction:
- the association between their smart meter number and their yearly income;
- their energy consumption measurements aggregated over intervals of 6 hours.
This can be translated into the following privacy queries for which no answer should be inferred from the published dataset.

The privacy queries expressing the attacker model

```
PQ1: SELECT ?sm ?y
        WHERE {?sm issda:
            associatedOccupier ?o .
          ?o issda:yearlyIncome ?y}
PQ2: SELECT  ?timeWindowEnd SUM(?c)
        WHERE {(?sm issda:consumption ?c ,
             ?ts)}
        GROUP BY  ?timeWindowEnd
        TIMEWINDOW (6h, 6h)
```

[4]available at `https://raw.githubusercontent.com/fr-anonymous/puck/main/issda\_schema.ttl`

With our approach, we can automatically detect several data reconstruction risks of the publisher model expressed by the above utility queries faced to an attacker model expressed by the above privacy queries, and provide the following explanations to the data publisher:
1) The first data reconstruction risk is due to the possibility of inferring an answer to PQ1 by combining answers to the utility queries UQ1 and UQ2.
2) The second data reconstruction risk is due to he possibility of inferring an answer to PQ2 from answers to the utility query UQ3 because:
a) PQ2 and UQ3 compute the same aggregate under the same conditions;
b) groups of UQ3 are partitions of groups of PQ2;
c) and finally, all time windows of PQ2 can be obtained as disjoint unions of some time windows of UQ3.
Based on the above explanations, the data publisher could modify his/her publisher model, for example by:
- removing one the utility queries UQ1 or UQ2;
- modifying the time window in UQ3, for instance by modifying the step between each consumption computation.

5.2 Algorithmic approach

The formal framework and the full characterization of privacy risks are described in [9]. Here we just summarize, and illustrate through examples, the principles underlying the verification algorithm: based on the query expressions only, it checks whether an answer of one the privacy queries can be inferred from answers to some utility queries.
In their most general form, the (privacy and utility) queries have 4 parts:
(i) a *graph pattern* that is an abstract specification (using a certain query language such as SPARQL) of the combinations between attributes/properties to be satisfied by the searched data ;
(ii) a set of *constraints* on the values of some of these attributes/properties to filter more precisely the searched data, using the FILTER constructor;
(iii) a *group by* part to specify the attributes/properties for which we want to group the searched data having the same values for those attributes/properties, using the GROUP BY constructor ;
(iv) a *result* defining the target attributes/properties the values of which must be returned by the query evaluation, and possibly aggregates to be computed on groups (specified in the *group by* part) using a given aggregate function.
When the aggregate function is computed on a dynamic property (such as *issda:consumption* in the ISSDA RDF dataset), *time windows* over which the aggregation must be computed must be specified. It is done using the TIMEWINDOW constructor with two parameters: a *size* to express the duration of each time window, and a *step* to express the time interval separating consecutive time window, which can thus be sliding (like in the UQ3 query of Section 5.1) or tumbling (like in the PQ2 query of Section 5.1).
The verification for a *simple* privacy query (i.e., without FILTER and GROUP-BY) against a set of any utility queries consists in checking whether the pattern of the privacy query is a sub-pattern of the union of patterns of some utility queries possibly joined by constraining some of their result attributes/properties to be equal. If this is the case, the corresponding utility queries are said *risky* for the privacy

query.
For example, up to variable renaming, the graph pattern of the SPARQL privacy query PQ_1:
?x1 issda:associatedOccupier ?x2 .
?x2 issda:yearlyIncome ?y2
is a sub-pattern of the pattern:
?x1 issda:associatedOccupier ?x2.
?x2 issda:nbOfPersons ?n.
?x2 issda:yearlyIncome ?y2
which is the joined union of the graph patterns of the two utility queries UQ_1 and UQ_2 obtained by equating the output variable $?y1$ of UQ_1 with the output variable $?x2$ of UQ_2.
This can be automatically detected, independently of the data. This exhibits a case where an answer to the privacy query PQ_1 can be derived from two answers to utility queries (for which the output variable $?y1$ of UQ_1 and the output variable $?x2$ of UQ_2 are instantiated with the same individual in the data).
For *complex* privacy queries, with FILTER and/or GROUP-BY constructors, we have to check in addition:

1. when the privacy query has a FILTER constructor, *whether* the FILTER constraints of the privacy query are compatible with the conjunction of FILTER constraints of the *risky* utility queries. This can be done using a CSP solver [5]

2. when the privacy query has a GROUP BY constructor, *whether* its graph pattern is isomophic (possibly up to a variable freezing) to the union of the graph patterns of the *risky* utility queries and its aggregate function is the same and applies to the same variable as at least one of the *risky* utility queries. This is the case for PQ2 and UQ3 in Section 5.1.

3. when , in addition, the privacy query has a TIMEWINDOW constructor, *whether* a time window for the privacy query can be obtained as the union of time windows of the *risky* utility queries when the aggregate function is MAX or MIN, or as the disjoint union of time windows of the *risky* utility queries when the aggregate function is SUM or COUNT [6]. This can be done using diophantine equation solver [7].

6. CHALLENGES FOR FUTURE WORK

In this paper, we have presented several approaches able to detect different types of privacy risks raised by publishing aggregates of (univariate and aligned) time series. We have highlighted some interpretable metrics (unicity, oddness) useful to measure the vulnerability to privacy risks of a time series dataset. We have also evaluated experimentally how the combination of some parameters of publisher and attacker models impact the risk. Finally, we have shown that machine learning approaches can be applied for predicting risk when formal methods or systematic simulation of attacks cannot be conducted. Depending on the methods used, we have indicated that some explanations can be provided to data publishers for helping them to understand and mitigate privacy risks. Here is a list of open challenges that should be considered:

- Evaluate the scalability of the methods to the length and the number of time series in real-world datasets (e.g. more than 35 Million time series for the French electrical provider, Enedis).

- Extend the models and the algorithms presented in this paper to more complex times series encountered in practice that may be multivariate and not necessarily aligned. The intrinsic complexity of systematic simulation approaches (such as the one presented in Section 3.2) is a limitation for their scalability. However, this is not such an important problem if they are used to build training datasets from which models can be automatically learned to predict privacy risks.

- Study the reliability of machine learning methods for modeling risk prediction on time series. Quantifying correctly the privacy risks using machine learning models requires machine learning methods with high accuracy. One challenge, outlined in Section 3.3, occurs when, by construction, the training dataset is highly imbalanced, thus making the accurate learning of the predictor very difficult for most of classifiers.

- For the approach described in Section 4 where we have seen that the learned model predicting membership inference risk performed worse when it is applied to a time period different from the one used in the training phase. It is a typical domain adaptation problem [23] that is made more complex because of the temporal aspect of the data and the multiple distribution shifts that can occur.

- Study how the use of machine learning techniques for predicting privacy risks can be used to construct *interpretable* explanations. A trade-off between prediction performance and interpretability should be achieved to obtain relevant explanation feedback with post-hoc explanation methods such as SHAP [13] or ANCHORS [19] applied to black-box prediction models.

- The deployment of formal methods, generalizing the one presented in Section 5, able to guarantee privacy by design based on a formal specification and automatic verification of publisher models compared to attacker models. This should allow to consider in particular attacker models corresponding to more abstract or less precise background knowledge about target users (e.g. holidays habits, heating habits, presence of a swimming pool, religion, etc.).

7. ACKNOWLEDGEMENT

This research was partially supported by TAILOR, a project funded by EU Horizon 2020 research and innovation program under GA No 952215.

[5]We used the python CSP library: https://pypi.org/project/CSP-Solver/
[6]We do not consider explicitly AVG because it can be computed by the union of 2 queries, one for computing SUM and the other one for computing COUNT.
[7]We used the Diophantine module of the python SymPy library : https://docs.sympy.org/latest/modules/solvers/diophantine.html

8. REFERENCES

[1] Anthony J. Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn J. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Discov.*, 31(3):606–660, 2017.

[2] Niklas Buescher, Spyros Boukoros, Stefan Bauregger, and Stefan Katzenbeisser. Two is not enough: Privacy assessment of aggregation schemes in smart metering. *Proceedings of Privacy Enhancing Technologies*, 2017.

[3] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, A. Terzis, and Florian Tramèr. Membership inference attacks from first principles. *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914, 2021.

[4] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1), 2002.

[5] David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 1959.

[6] Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 248–257, 2021.

[7] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference (TCC)*, 2006.

[8] Flavio D Garcia and Bart Jacobs. Privacy-friendly energy-metering via homomorphic encryption. In *International Workshop on Security and Trust Management*, 2010.

[9] Christophe Bobineau Hira Asghar and Marie-Christine Rousset. Identifying Privacy Risks raised by Utility Queries. In *23rd International conference on Web Information Systems Engineering (WISE 2022)*, New York, United States, 2022. ACM.

[10] Klaus Kursawe, George Danezis, and Markulf Kohlweiss. Privacy-friendly aggregation for the smart-grid. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 175–191, 2011.

[11] Jack I Lerner and Deirdre K Mulligan. Taking the long view on the fourth amendment: Stored records and the sanctity of the home. *Stan. Tech. L. Rev.*, 2008.

[12] Rongxing Lu, Xiaohui Liang, Xu Li, Xiaodong Lin, and Xuemin Shen. Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications. *IEEE Transactions on Parallel and Distributed Systems*, 2012.

[13] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[14] Francesca Naretto, Roberto Pellungrini, Anna Monreale, Franco Maria Nardini, and Mirco Musolesi. Predicting and explaining privacy risk exposure in mobility data. In Annalisa Appice, Grigorios Tsoumakas, Yannis Manolopoulos, and Stan Matwin, editors, *Discovery Science - 23rd International Conference, DS 2020, Thessaloniki, Greece, October 19-21, 2020, Proceedings*, volume 12323 of *Lecture Notes in Computer Science*, pages 403–418. Springer, 2020.

[15] Roberto Pellungrini, Luca Pappalardo, Francesca Pratesi, and Anna Monreale. A data mining approach to assess privacy risk in human mobility data. *CM Transactions on Intelligent Systems and Technology (TIST)*, 9(3), 2017.

[16] Francesca Pratesi, Anna Monreale, Roberto Trasarti, Fosca Giannotti, Dino Pedreschi, and Tadashi Yanagihara. PRUDEnce: a system for assessing Privacy Risk vs Utility in Data sharing Ecosystems. *Transactions on Data Privacy*, 11, 2018.

[17] Apostolos Pyrgelis. *Evaluating Privacy-Friendly Mobility Analytics on Aggregate Location Data*. PhD thesis, UCL (University College London), 2019.

[18] Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock knock, who's there? membership inference on aggregate location data. In *25th Annual Network and Distributed System Security Symposium, NDSS*. The Internet Society, 2018.

[19] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 1527–1535. AAAI Press, 2018.

[20] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *ArXiv*, abs/2007.07646, 2020.

[21] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, 2017.

[22] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3145–3153, 2017.

[23] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016.

[24] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[25] Antonin Voyez, Tristan Allard, Gildas Avoine, Pierre Cauchois, Élisa Fromont, and Matthieu Simonin. Membership inference attacks on aggregated time series with linear programming. In Sabrina De Capitani

di Vimercati and Pierangela Samarati, editors, *Proceedings of the 19th International Conference on Security and Cryptography, SECRYPT*, pages 193–204. SCITEPRESS, 2022.

[26] Antonin Voyez, Tristan Allard, Gildas Avoine, Pierre Cauchois, Elisa Fromont, and Matthieu Simonin. Unique in the Smart Grid -The Privacy Cost of Fine-Grained Electrical Consumption Data. Preprint https://hal.archives-ouvertes.fr/hal-03833605, November 2022.

[27] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.