Supplementary Material for Embeddings of Networks Using Small-Size Graphlets: Comparisons and Analysis

Luce le Gorrec
Department of Mathematics and Statistics
University of Strathclyde
Glasgow G1 1XH, United Kingdom
luce.le-gorrec@strath.ac.uk

Philip A. Knight
Department of Mathematics and Statistics
University of Strathclyde
Glasgow G1 1XH, United Kingdom
p.a.knight@strath.ac.uk

Auguste Caen
School of GeoSciences and NCEO
University of Edinburgh
Edinburgh, EH9 3FF, UK
auguste.caen@gmail.com

I. Introduction

This document provides some additional detail to the paper *Embeddings of Networks Using Small-Size Graphlets: Comparisons and Analysis*, submitted to the jounal *Social Network Analysis and Mining*. Section II contains a thorough description of the network benchmarks used in Section 3 of the main paper. The key used to identify k-node graphlets is given inn Section III. The performance of our algorithms and of alternatives is sensitive to the particular choice of certain building blocks and to the values of parameters and hyper parameters. In Section IV we justify the algorithmic choices we made to build our algorithms while Section V and Section VI are dedicated to parameter tuning of existing methods. Finally, Section VII provides additional information concerning the unsupervised approach from Section 5.1 of the main paper.

II. The Network Benchmark

In our numerical experiments, we have used directed networks with more than 50 nodes, from 4 different application fields, namely food webs, electronic circuits, discourse structures and social networks. Here we give information about the networks, the repositories where we found them as well as any post-processing we have applied to them to obtain networks that are well suited to our analysis—unweighted directed static networks without self-loops.

A. Food webs.

A food web represents the consumer–resource relations between species in an ecosystem. That is, there exists a relation between species a and b if individuals from species a consume individuals from species b—most often, this means a eats b but other kinds of relations may exist, for instance, if species a is a parasite of the host species b.

In a network that represents a food web, the nodes are the species, and the edges represent the consumer–resource relations. We consider directed edges such that there is an edge from a to b if species a consumes species b. Networks from this field may be weighted, with weights representing either a level of confidence in the consumer–resource relation, or the quantity of species b that is consumed by species a. When meeting such weighted networks, we have kept all edges but removed the weight. Self-loops occasionally occur, in cannibal species for instance. These are excised. Finally, we have also met some food webs where prey and predators are partitioned into two disjoint groups of species. We have discarded such networks as the bipartite structure is the dominant characteristic.

In total we have collected 70 food webs from the following repositories.

- 33 food webs have been extracted from the website GlobalWeb [1]. We have filtered all the food webs to keep only the non-bipartite networks with more than 50 nodes. Among this first set, we have only kept the networks in which we, with limited biological knowledge, could interpret the relations—that is, numerical values between species if there were valued, as well as which species were preys, and which were predators.
- 21 food webs have been extracted from [2]. This article analyses different kinds of food web to understand the impact of taking into account parasitism. It first observes food webs without parasites and considers free-living species only (species that do not need a host). Then the study adds parasitic species, but without concomitant links (that is, the only links that are added are those that indicate that a species b is an host for a parasite species a). Finally, food webs with parasitic species and concomitant links are analysed (a concomitant link is a link that indicates that if predator a eats preys b, it also eats parasites hosted by b). The study analyses seven different ecosystems, and for each ecosystem, three food webs are provided.
- We have used the data from [3] in a similar way to [1], to generate 4 food webs. In the first we keep the

This project was supported by the Royal Academy of Engineering and the Office of the Chief Science Advisor for National Security under the UK Intelligence Community Postdoctoral Fellowship Programme.

links corresponding to predation only, the second keeps host–parasite relations, the third includes concomitant predation relations and a final network includes all the relations.
- 11 food webs have been extracted from Pajek [4]. In the Pajek dataset, edges are weighted, and some nodes are not formal species—for instance, in each network, there are three nodes labelled Input, Output and Respiration. We have decided to keep all these nodes when we extract the networks, and to take into account all the edges, whatever their weight.
- The last food web of our benchmark comes from the Konnect repository [5], taken from the study [6].

Food webs from the second and third items were found using the Icon Colorado Repository [7].

B. Electronic Circuits.

Here we consider an electronic circuit as an object composed of one or more inputs that provide one or more outputs by passing through logic gates (AND, NOR, etc.). An electronic circuit can be represented by a network where a node may be an input, an output or a logic gate, and an edge indicates that a signal enters or leaves a logic gate. For instance, the relation

$$\Sigma_1 = AND(\Sigma_2, \Sigma_3)$$

will be represented by one node, with two edges entering the node, representing signals Σ_2 and Σ_3, and one edge leaving the node that represents signal Σ_1. An example for the full electronic circuit s27 from [8] is provided in Figure 1.

For this field, 52 networks with more than 50 nodes have been extracted from two classical benchmarks, namely ISCAS'89 [8] and ITC'99 [9], and have been respectively found on repositories [10] and [7].

C. Discourse Structures.

In this application, Segmented Discourse Representation Theory [11] is used to represent the rhetorical relations between discourse units by means of networks. An example extracted from [12] and [13] is shown in Figure 2 to explain how the network representation of a discourse is built. The discourse units can be of two kinds: elementary discourse units (EDUs), or complex discourse units (CDUs) that correspond to discourse units composed of several EDUs.

For this field, we have extracted the networks from the STAC dataset[1] [14], which is a corpus of chat conversations between players from an online version of the game board "The Settlers of Catan". Two corpora are available to download: a linguistic one, in which there are only human being exchanges (players and observers), and a situated one, which is the linguistic corpus completed with information about the game given by the computer by means of canned messages[2]. For both corpora, we have extracted all the networks with more than 50 nodes, which has resulted in a dataset of 195 networks—22 from the linguistic corpus, 173 from the situated corpus. As shown in Figure 2, the edges are labelled. We do not consider these labels in our study.

[1] https://www.irit.fr/STAC/corpus.html

[2] Such as "It's <player id> to roll the dice", "<player id> ends its turn", etc.

D. Social relations.

The term "social relation" covers a wide range of relation types, from friendship relations on online social networks to physical contacts; and from commercial exchanges to genealogical relations, or even domination among a group of animals. In our study, we have limited ourselves to human relations that are directed and can be symmetrical—this excludes, for instance, genealogical or student–teacher relationships. The resulting dataset consisting of 81 networks can be roughly divided into two kinds of relations, namely feelings and exchanges.

Networks representing feeling relations are mainly based on surveys of groups of individuals, such as students from the same school or co-workers, who were asked to name their friends, or who they turn to to ask for advice or support, etc. In such networks, the edges may be weighted according to the strength of the feeling. These weights may sometimes be negative if the individuals can expressed both positive and negative feelings—trust and distrust, friend or foe. In these cases we only take into account the existence of a feeling, and we draw an edge whenever the individual a expresses a feeling—weak or strong, positive or negative—for individual b. The edges may also be labelled according to the nature of the relation if several kinds of relations have been tested. In this case, we have created one network per relation type, and one network merging all the relation types.

Networks representing exchanges (emails, phone calls, messages on online social websites) may be dynamic (the times of exchanges are known), may have multi-edges (several exchanges from individual a to individual b are possible). Self-loops are sometimes allowed since, for example, one can send an email to oneself. We have transformed these networks into static unweighted networks by adding an edge from a to b so long as there is at least one exchange at some time from a to b. Finally, we remove self-loops.

The sources of the social networks we have used is as follows. Many of our social networks representing feelings come from surveys.

- 29 networks come from the study [15], which is a survey held in several schools. Students were asked to name their closest friends. For each friend named, the student was asked whether he/she participated in any of five activities with the friend. In the resulting network, nodes are students, and there is an edge from a to b if student a named b as a close friend. The edges are labelled by the activities that a claims to have done with b. We limited ourselves to the studies of the 30 first schools, omitting school 3 which surveyed fewer than 50 students. In post-processing we removed edge labels. Our source for data is the repository [16].

l1: # 4 inputs
l2: # 1 outputs
l3: # 3 D-type flip-flops
l4: # 3 inverters
l5: # 8 gates(1 ANDs +1 NANDs +2 ORs +4 NORs)
l6: INPUT(G0)
l7: INPUT(G1)
l8: INPUT(G2)
l9: INPUT(G3)
l10: OUTPUT(G17)
l11: G5 = DFF(G10)
l12: G6 = DFF(G11)
l13: G7 = DFF(G13)
l14: G14 = NOT(G0)
l15: G17 = NOT(G11)
l16: G8 = AND(G14,G6)
l17: G15 = OR(G12,G8)
l18: G16 = OR(G3,G8)
l19: G9 = NAND(G15,G16)
l20: G10 = NOR(G14,G11)
l21: G11 = NOR(G5,G9)
l22: G12 = NOR(G1,G7)
l23: G13 = NOR(G2,G12)

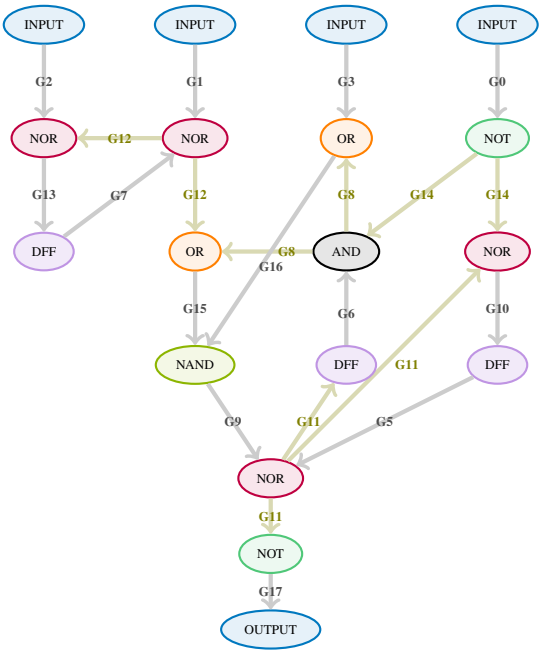

Fig. 1. Left: Netlist of the electronic circuit s27 from [8]. Right: The resulting network, with labels displayed. Each gate/input/output corresponds to one node. The edges are signal transfers. For instance, signal G14, which is the result of input signal G0 going through a NOT gate (*l14*), enters both a NOR gate with signal G11 to create signal G10 (*l20*), and an AND gate with signal G6 to create signal G8 (*l16*).

γ_1 : John had a lovely evening last night.
γ_2 : He had a great meal.
γ_3 : He ate salmon.
γ_4 : He devoured lots of cheese.
γ_5 : He won a dancing competition.
Γ_6 : $\gamma_2 \cup \gamma_5$
Γ_7 : $\gamma_3 \cup \gamma_4$

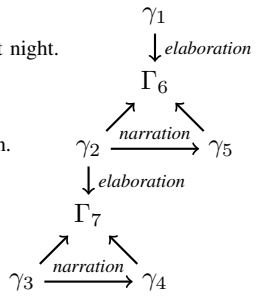

Fig. 2. Left: A short text divided into discourse units. The EDUs are identified by a γ, the CDUs by a Γ. Right: Discourse structure of the text with a network representation. The edge labels correspond to rhetorical relations. An edge with no label links an EDU to the CDU it is a member of.

- 2 networks come from the study [17], where students in a high school were asked to name their friends. Two surveys took place, one in 1957 and one in 1958. Source can be found in [16].
- 1 network from the study [18]. This is a survey based on the same principles as above: students living in a residence are asked to name their friends within the residence, giving a strength to their relationship. The initial network is available from [16].
- 9 networks come from the study [19], a survey of Dutch secondary schools. Pupils have been asked to name other pupils with whom they share an emotional support relationship (either give or receive such support). 36 classrooms were surveyed, resulting in 36 networks. We kept the 9 classrooms whose largest weak component contained at least 50 nodes. These networks are available at [20].
- 6 networks come from a survey by Vladis Krebs of the IT department of a Fortune 500 company. Workers were asked to name other workers with whom they have certain kinds of relationships (co-workers on multiple projects, sought for advice for decision making, sought for technical expertise, discussion of customers issues). Their relationships were assigned a value ("for each kind of [relation], each individual reported the frequency of contact with each of the other individuals (Yearly or less, Quarterly, Monthly, Weekly, Daily or more"). In the resulting networks (one per relationship type plus one for the merger of all relationships), nodes are workers, and there is an edge from a to b if b has been named by a for this type of relationship. These sources were taken from [16]. Note that for customer issues, the answers from the last individual have been only partially reported (11 answers on 55). We have kept the network with the incomplete answers.
- 4 networks come from a similar study to Krebs but in a law firm [21], with questions about advice, co-workers and friendship. The relations are not valued. Once again, we have one network per relation kind plus one that merges all the relations. These networks were found at [16].
- 4 networks come from another work place survey, this time of GPs in Illinois [22]. The questions asked were: "When you need information or advice about questions of therapy where do you usually turn?", "Who are the

three or four physicians with whom you most often find yourself discussing cases or therapy in the course of an ordinary week—last week for instance", "Would you tell me the first names of your three friends whom you see most often socially?". This resulted in 4 networks (advice, discussion, friendship, and the merger of all relationships). These sources were found at [16].

- 2 networks come from a survey of a research team in a manufacturing company [23]. Given a list of other coworkers, the researchers were asked to answer two questions. The first question was "Please indicate the extent to which the people listed below provide you with information you use to accomplish your work" on the scale : "0: I Do Not Know This Person/I Have Never Met this Person; 1: Very Infrequently; 2: Infrequently; 3: Somewhat Infrequently; 4: Somewhat Frequently; 5: Frequently; and 6: Very Frequently". The second question asked the researchers to rate the following assertion "I understand this person's knowledge and skills. This does not necessarily mean that I have these skills or am knowledgeable in these domains but that I understand what skills this person has and domains they are knowledgeable in" on the scale: "0: I Do Not Know This Person/I Have Never Met this Person; 1: Strongly Disagree; 2: Disagree; 3: Somewhat Disagree; 4: Somewhat Agree; 5: Agree; and 6: Strongly Agree." We have built two networks from this survey, one per question, where a node is a researcher and there is an edge from a to b if for researcher b, a has provided an answer greater or equal to 3. Initial data were found at [24].
- 1 network comes from a survey of drug users in Hartford, Connecticut [25]. Nodes are drug users, and directed edges represent acquaintance. Data can be found at [26].
- 1 network comes from a survey of prison inmates who were asked to name their closest friends within the prison [27]. In the resulting network, nodes are prison inmates and there is an edge from a to b if b is designated as a close friend by a. The network can been found at [4].
- 1 network comes from a survey that lists the frequent visits among families living in a same neighbourhood [28], [29]. The visits can be of 3 kinds, namely "visiting relation as ordinary", "visits among kin", and "visits among ritual kin". In the network, nodes are the families and there is an edge from a to b if family a frequently visits family b. A post-processing is applied to remove the labels on the edges (that is, we keep only the information about frequent visits, and do not take into account the type of visits). Initial data can be found at [4].
- 2 networks come from another survey based on the same principles as above [28]. One of the networks was built exactly in the same way (but on another neighbourhood), and has been treated in the same way. For the other, each family was asked which other families they would notify in case of a family member's death. In this network, each node is a family, and there is an edge from a to b if family a answered they would advise family b in case of death. Initial data were taken from [4].

We also have social networks extracted from websites in which edges represent feelings.

- 1 network, presented in [30], is an online social network called Anybeat, where people can choose to follow other people. Nodes are users, edges denote the fact a user follows another user. Initial data came from [31].
- 3 networks come from the technology website Slashdot-Zoo, where users can mark other users as friend or foe. The three networks correspond to three different time-stamps. One has been presented in [32] and can be found at [5]. The two others have been presented in [33] and can be found at [34]. In these networks, nodes are users, and an edge from a to b indicates that user a has labelled user b as his/her friend or his/her foe, resulting in labelled edges. We have post-processed the networks to remove these labels, so that we only keep the information that user a has expressed a feeling for user b.
- 2 networks, discussed in [35] and available at [34], focus on the hyperlinks between communities (called subreddits) of the social news website Reddit. A subreddit can make a post that mentions a post from another subreddit. Two networks are built, one for the mention of another post that is done in the title of the current post, the other if the mention is done in the post body. The mention can be done in a negative, neutral or positive way. In the networks, a node is a subreddit, and there is an edge from a to b if a post from subreddit a mentions a post from subreddit b in its title (respectively its body). Post-processing was required for these networks. There are weights (-1 if the post from a is negative toward b) and timestamps (time where the post appeared) associated with the edges in these networks, that we choose to ignore.
- 1 network, presented in [36] and available at [5], is a Who-trusts-whom social network taken from a consumer review website. Nodes are users and an edge from a to b means that user a has indicated he/she trusts user b.
- 1 network, presented in [37], is based on an online community platform for developers of free software. Users can give certification to other users or to themselves corresponding to the trust a user has to another user (or themselves). Three levels of certification are possible. In this network, nodes are users, edges are trust relations. In post-processing, self-loops and edge weights corresponding to certification level are removed. The network before post-processing can be found at [5].
- 2 networks, presented in [38], [39], are two Who-trusts-whom networks of traders extracted from two Bitcoin platforms. Users of the platforms can rate other members with integers from -10 (total distrust) to +10 (total trust). Post-processing was required for these networks. We chose to ignore the weights and time-stamps associated with the edges in these networks. That is, if member a rates a grade of $k \in \{\pm 1, ..., \pm 10\}$ to user b at time t, in

the resulting network we only see one static edge from a to b. The networks (before post-processing) have been extracted from [34].

Finally, we also have some social networks that represent exchanges between individuals. These exchanges are mainly emails or phone calls.

- 1 network, presented in [40], is a subset of the posts left by Facebook users on their friends' walls. Nodes are users, an edge from a to b indicates user a posts something on the wall of user b. Post-processing have been used to remove weight on edges (a user may post several times on the same wall), and self-loops (posting on one's own wall). The initial network has been found at [5].
- 1 network, presented in [41], is the list of email exchanges among members of a European research institution. Nodes are email addresses and there is an edge from a to b if a has sent at least one email to b. This network come from [34].
- 1 network, presented in [42], is based on the same principle as the network above, but in a mid-sized manufacturing company instead of a European research institution. Post-processing was applied because the initial network was dynamic: times when emails were sent were known. We create an edge from a to b if a sent at least one email to b. Original data are available at [5].
- 2 networks, presented in [43], represent the communication network of the Linux Kernel mailing list. Nodes are email addresses, there is an edges from a to b if a had made a reply to a request of b. A post-processing has been applied since the initial network is dynamic, with the time where a reply is made which is known. We create a simple unweighted edge from a to b if at least one reply has been made from a to a request from b. The resulting network has 3 connected components with more than 50 nodes, but the smallest one is a simple path (and hence, is bipartite). We have kept the two largest only. The original network can be found at [5].
- 1 network, presented in [44] and available at [5], documents messages sent on a social website among students from the University of California, Irvine. Nodes are students, and there is an edge from a to b if a sent a message to b. Again, post-processing has been applied since the initial network was dynamic: times when messages were sent were known. We create an edge from a to b if a sent at least one message to b.
- 2 networks, analysed in [45], are the results of an investigation of university students within the Copenhagen Networks Study. Two networks have been generated. One corresponds to phone calls, the other ones to text messages. Nodes are students and there is an edge from a to b if a has called/texted b. The networks are dynamic (times when messages or phone calls have been passed is known), and weighted (number of text messages, number of phone calls plus length of phone calls). We have post-processed these networks to make them unweighted. We did not take into account the missed calls.
- 1 network, presented in [46], is the result of an investigation of hashish and cocaine importers who were identified as members of the so-called Caviar network. Nodes represent people actively implicated in the Caviar network who were caught in a police surveillance net, and there is an edge from a to b if communications from a to b have been observed during the investigation. A post-processing has been applied because the initial network is valued by the strength of contact between two individuals. We have kept all edges, whatever the strength of the contact, and made them unweighted. Original dataset is available at [47].

In total we obtained 81 networks for this database. As for the food webs, the listing provided by [7] was very helpful for sourcing the data.

III. Graphlet Identifier

Studies that focus on graphlets need to identify these graphlets with a unique identifier. Below we give the general formula to derive an identifier for a graph, then we explain how it is applied to the graphlets.

We start with an unweighted directed graph $G = (V, E)$ with nodes $V = \{v_1, ..., v_k\}$ ($|V| = k$) and edges E where (v_i, v_j) indicates the presence of an edge from v_i to v_j in G. Let $\mathbf{A} \in \{0, 1\}^{k \times k}$ be the adjacency matrix of G,

$$a_{i,j} = 1 \iff (v_i, v_j) \in E$$

and $\mathbf{B} \in \mathbb{R}^{k \times k}$ be a matrix whose entry (i, j) is given by:

$$b_{i,j} = 2^{(i-1)\times k+(j-1)},$$

We define an identifier for G by the integer

$$id_G = \mathbf{e}^T(\mathbf{B} \circ \mathbf{A})\mathbf{e}, \tag{III.1}$$

with $\circ$ the Hadamard product, or pairwise matrix multiplication, and $\mathbf{e}$ a vector of ones.

When considering graphlets, we presume two isomorphic subgraphs correspond to the same graphlet. In other words, in a graphlet, node label is immaterial. To derive a graphlet identifier one chooses to label the nodes of the corresponding subgraph so that the identifier provided by (III.1) is as small as possible.

This method of identifying graphlets is used, amongst many others, in [48], [49] and in the 3-and 4-node graphlet detection routine from [34].

In our study, we focus on 3-node and 4-node graphlets. Since (III.1) can give the same value to different graphlets if they have different number of nodes, we specify the number of nodes as a prefix of graphlet identifier. That is, the 3-node graphlet identified by 14 will be labelled 3-14, and the 4-node graphlet identified by 14 will be labelled 4-14, as illustrated in Figure 3.

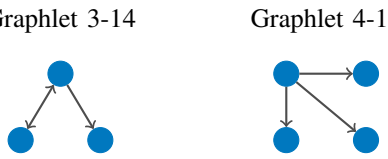

Fig. 3. Two graphlets with the same smallest value for Equation (III.1).

IV. Algorithmic Choices and Implementation

We had to make a number of choices when designing our first algorithm for network embeddings, such as the number of nodes within graphlets, the normalisation to use in (2.1) from the main paper, the feature extraction procedure, and the number of features to be kept. Our method is described in Algorithms 1 and 2.

Algorithm 1: Naive Classification

Input: $G \in \mathcal{G}$, $\widetilde{\mathbf{U}} \in \mathbb{R}^{M\times k}$, $\overline{\mathbf{x}} \in \mathbb{R}^M$, $\mathcal{C} = \{(\overline{\mathbf{c}_i}, \ell_i) \in \mathbb{R}^k \times \mathcal{L}, i = 1, \ldots, |\mathcal{L}|\}$
Output: $\ell^* \in \mathcal{L}$ the predicted class of G

begin
  compute $\{\phi_k(G)\}_{k\in\mathcal{K}}, \{\alpha_k(G)\}_{k\in\mathcal{K}}$
  $\mathbf{x} \leftarrow f(G)$ according to (2.1)
  $\mathbf{c} \leftarrow (\widetilde{\mathbf{U}}^k)^T \times (\mathbf{x} - \overline{\mathbf{x}})$
  $(\mathbf{c}^*, \ell^*) = \underset{(\overline{\mathbf{c}_i},\ell_i)\in\mathcal{C}}{\arg\min} \|\mathbf{c} - \overline{\mathbf{c}_i}\|_2$
return ℓ^*

Algorithm 2: Preprocessing of Algorithm 1

Input: A training set $\{(G_\sigma, \ell_\sigma) \in \mathcal{G} \times \mathcal{L}, \forall \sigma \in \mathcal{S}\}$, an integer $k < M$ ($k < |\mathcal{L}|$ if FDA)
Output: $\overline{\mathbf{x}} \in \mathbb{R}^M$, $\widetilde{\mathbf{U}}^k \in \mathbb{R}^{M\times k}$, $\mathcal{C} = \{(\overline{\mathbf{c}_i}, \ell_i) \in \mathbb{R}^k \times \mathcal{L}, i = 1, \ldots, |\mathcal{L}|\}$

begin
  for $\sigma \in \mathcal{S}$ **do**
    compute $\{\phi_k(G_\sigma)\}_{k\in\mathcal{K}}, \{\alpha_k(G_\sigma)\}_{k\in\mathcal{K}}$
    $\mathbf{x}_\sigma \leftarrow f(G_\sigma)$ according to (2.1)
  $\overline{\mathbf{x}} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{\sigma\in\mathcal{S}} \mathbf{x}_\sigma$
  apply PCA or FDA on $\mathbf{X} = \begin{bmatrix}\mathbf{x}_{\sigma_1} & \ldots & \mathbf{x}_{\sigma_{|\mathcal{S}|}}\end{bmatrix}^T$ to obtain $\widetilde{\mathbf{U}} \in \mathbb{R}^{M\times k}$
  $\mathbf{C} \leftarrow (\mathbf{X} - \mathbf{e} \times \overline{\mathbf{x}}^T) \times \widetilde{\mathbf{U}}$
  $\mathcal{C} \leftarrow \emptyset$
  for $\ell \in \mathcal{L}$ **do**
    $\overline{\mathbf{c}_\ell} \leftarrow \frac{1}{|\{\sigma\in\mathcal{S}:\ell_\sigma=\ell\}|} \sum_{\sigma\in\mathcal{S}:\ell_\sigma=l} \mathbf{c}_\sigma$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(\overline{\mathbf{c}_\ell}, \ell)\}$
return $\overline{\mathbf{x}}$, $\widetilde{\mathbf{U}}$, $\mathcal{C}$

In order to recover the labels of the networks in the benchmark described in Section II we divide our database into a training set and a testing set. The training set $\mathcal{S}$ is built by randomly selecting $N_B = 40$ networks from each class/field. Thus, $|\mathcal{S}| = |\mathcal{L}| \times N_B$. The remaining networks form the test set $\mathcal{T}$. We repeated this process on 50 pairs of sets $(\mathcal{S}, \mathcal{T})$ and we present the mean and standard deviation of the results. To improve presentation, the means and standard deviations have been multiplied by 10^{-1} and 10^{-2} respectively. In all the tables presenting our results, the labels *fw*, *elec*, *disc* and *soc* relate to the food web, electronic circuit, discourse structure and social relationship benchmarks respectively.

A. Feature Extraction Procedure

As stated in Section 2 of the main paper, in Algorithm 1, we use a feature extraction procedure to reduce the dimension of network embeddings. To do this, we have tested two such extraction procedures, namely Principal Component Analysis (PCA) and Fisher's Discriminant Analysis (FDA) [50], in a supervised fashion. We assume that we want to extract p features of samples from a dataset containing n samples, described by q real valued features. We also assume that we want to split the samples into c classes.

Given such a dataset, PCA can be seen as a change of basis of the feature space. In a nutshell, it finds an orthogonal basis where the new axes (called principal axes) are sorted such that the first principal axis maximises the variance of the samples if they were projected onto a 1D space, the second principal axis also maximises the variance, but is orthogonal to the first one, and so on. Thus the p first principal axes form the p-dimension space which provides the best representation of the dataset, in terms of variance. We use PCA as a supervised feature extraction procedure for our downstream classification task, by following the steps proposed in [51]. That is, we compute a matrix $\widetilde{\mathbf{U}} \in \mathbb{R}^{q\times p}$ containing the leading eigenvectors of the covariance matrix of the training set, and we use $\widetilde{\mathbf{U}}$ to get the shifted projections of samples from both training and test sets onto a lower dimension space whose directions are the principal directions provided by $\widetilde{\mathbf{U}}$.

PCA is not able to make use of the available knowledge about the training set labels. For this, FDA can be used instead. FDA aims to find a space in which the mean elements of different classes are well separated, while the variance within each class (that is, the sum of square distances between samples from a class and the mean element of this class) is small. This space is described by the matrix $\mathbf{W} \in \mathbb{R}^{q\times p}$ whose columns are the solution of the generalised eigenvalue problem $\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w}$ where $\mathbf{S}_B, \mathbf{S}_W \in \mathbb{R}^{q\times q}$ are the between- and within-classes scatter matrices, respectively. Since the rank of $\mathbf{S}_B$ is at most $c-1$, $\mathbf{W}$ has at most $c-1$ columns, constraining the dimension of the space produced by FDA [52]. A matrix $\widetilde{\mathbf{U}}$ is then built so that its columns form an orthogonal basis for the space described by $\mathbf{W}$'s columns. This matrix is used to project samples onto this space, in a similar fashion to PCA.

A comparison between PCA and FDA is provided in Table I. Precision, recall and $F1$-score of our method for $\mathcal{K} = \{3, 4\}$ are provided for both extraction procedures, along with different normalisation procedures—See Section IV-B. The number

of axes kept is 10 for PCA, and 3 for FDA. Best scores are highlighted.

It is noticeable that the scores are globally much lower using FDA than PCA. One could reason that this is because a 3-dimension space is not sufficient to well approximate the dataset. However, this alone cannot explain the bad results of FDA, since the method with PCA using only 3 principal axes still achieves much better scores[3]. However, we remark that the magnitude orders within the scatter matrices are wide (from 1 to 10^{-11}), and that the within-scatter matrix is rank deficient. Finally, the highest generalised eigenvalue has an order of magnitude of 10^8. A consequence is that the generalised eigenvalue problem is poorly resolved (the magnitude order of $\|\mathbf{S}_B\mathbf{w}_i - \lambda_i\mathbf{S}_W\mathbf{w}_i\|/\|\mathbf{S}_B\mathbf{w}_i\|$ is 10^{-i-2} for $i = 1, 2, 3$). This may explain the poor results of this feature extraction procedure for our problem. Furthermore, while for PCA the global variance and average of the dataset are estimated using the whole training set, FDA estimates within-class variances and averages using only the training samples that belong to the given class. Our training set has been built using 40 networks from each class, which may not be enough for FDA to accurately estimate these within-class variances and averages.

Thus, in the following, we choose to keep the first normalisation, and to use PCA as a procedure to reduce dimensions.

B. Normalisation

In order to mitigate against the difference of amplitudes that exist between the occurrences of graphlets of different sizes within a network, we compare six normalisation procedures.

1) $\alpha_k(G) = \frac{m_k}{k!}\binom{n}{k}^{-1}$

2) $\alpha_k(G) = m_k\binom{n}{k}^{-1}$

3) $\alpha_k(G) = \binom{n}{k}^{-1}$

4) $\alpha_k(G) = ||\phi_k(G)||^{-1}$

5) $\alpha_k(G) = \sqrt{m_k}||\phi_k(G)||^{-1}$

6) $\alpha_k(G) = 1$ (no normalisation)

Our rationale for the first three is that the maximum number of k-node graphlets that can occur in a network of n nodes is given by the binomial coefficient. For connected k-node graphlets this maximum is achieved if the shortest path between each pair of nodes in the network is at most k. Let us assume we have a graph G such that $\phi_k(G) = \beta\mathbf{e}$, that is, each non-isomorphic k-node graphlet has the same number of occurrences in G, thus $\beta = \binom{n}{k}m_k^{-1}$. In such a case, normalisation 2) results in a uniform $f(G)$ in (2.1) from the main paper. Normalisation 1) slightly boosts larger k, while 3) favours smaller k.

A comparison of the the six formulations of α_k are provided in Table I. For PCA the best scores tend to be achieved by the first normalisation, and only this latter and the second normalisation provide consistently high $F1$-scores. Thus we use Normalisation 1).

C. Number of Nodes Within Graphlets

A decision must be made on the number of nodes the graphlets must have to accurately classify the networks. We investigate graphlets with 3, 4, and 5 nodes. For 5-node graphlets, however, 10 networks from the social dataset have been removed (those with more than 10K nodes), since it takes too long to compute the number of occurrences of such graphlets within these networks. The scores are provided in Table II. Embeddings were built without feature extraction. From this table, we can see that if we were to just choose one size of graphlet, then 5-node graphlets provide the greatest accuracy, but size combinations generally work better. And when combining these graphlets, the results are very similar whether we use $\mathcal{K} = \{3, 4\}$ or $\mathcal{K} = \{3, 4, 5\}$: the former provides noticeably better results for food webs and social networks, while the latter detects electronic circuits more accurately. Hence, adding 5-node graphlets to 3-and-4-node graphlets in (2.1) from the main paper does not bring sufficient information about network structures for our classification task to justify the cost in the run-time for computing 5-node graphlets—see Section 2.3 for a discussion about the complexity. For these reasons we will focus on 3-and-4-node graphlets only.

D. Number of Principal Axes

In this section, we briefly justify our choice of reducing the dimension, and of applying normalisation factors α_k on our embeddings—see Section IV-B.

To show our rationale for the normalisation step, we plot the average $F1$-score and average standard deviation (that is, the sum of the F1-scores over the classes, divided by the number of classes, similarly for standard deviation) in the left panel of Figure 4. These curves have been obtained using a combination of 3-and 4-node graphlets, as justified in Section IV-C. We only plot the curves for the 50 first principal axes to improve the visibility. This has no impact as results are stabilised for larger number of components. The red curves correspond to the embedding technique when no normalisation is applied, while the blue ones include normalisation. We see that, both in terms of $F1$-score and of standard deviation, the normalisation improves results.

The right panel of Figure 4 provides justification for our rationale for dimension reduction, and to extract 18 features to build Tables 2 and 3 in the main paper. Here we plot the curve of average $F1$-score and average standard deviation, minus the average $F1$-score and standard deviation obtained by keeping all the graphlets. We observe that the results are stabilised

[3] $F1$-score of PCA with 3 principal components, using the first normalisation: fw: 0.95; $elec$: 0.56; $disc$: 0.99; soc: 0.81.

TABLE I
COMPARISON BETWEEN THE NORMALISATION PROCEDURES AND THE REDUCTION PROCEDURES.

		PCA						FDA					
		Precision		Recall		F1-score		Precision		Recall		F1-score	
		mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
(1)	fw	**9.50**	**2.5**	**10**	**0.3**	**9.74**	**1.3**	6.73	12.3	6.64	10.5	6.59	8.4
	$elec$	**10**	**0**	8.98	9.0	9.44	5.0	5.15	14.1	9.22	8.0	6.48	11.6
	$disc$	9.88	0.8	**9.95**	**0.3**	9.91	0.4	**9.66**	**3.7**	9.81	3.7	9.73	2.7
	soc	**10**	**0.2**	**9.62**	**2.2**	**9.81**	**1.1**	8.28	10.3	5.39	10.3	6.44	13.1
(2)	fw	9.01	3.3	**10**	**0.4**	9.48	1.9	7.02	13.6	6.88	11.8	6.83	9.0
	$elec$	9.97	1.6	9.44	6.4	**9.69**	**3.4**	5.41	17.6	9.20	7.7	6.63	13.3
	$disc$	9.93	0.6	9.90	0.4	9.91	0.4	9.45	4.16	9.72	3.6	9.58	3.1
	soc	**10**	**0.3**	9.47	2.9	9.72	1.6	8.50	10.35	5.25	15.1	6.39	14.1
(3)	fw	8.92	4.1	8.98	4.9	8.94	3.4	6.71	12.3	6.70	10.2	6.62	8.4
	$elec$	8.31	8.4	8.87	8.5	8.55	6.7	5.41	14.1	9.41	6.6	6.75	11.4
	$disc$	9.87	0.8	**9.95**	**0.3**	9.91	0.4	9.61	3.4	**9.87**	**2.1**	**9.74**	**2.3**
	soc	9.79	1.7	9.22	3.7	9.45	2.2	8.38	10.1	5.20	15.3	6.31	14.0
(4)	fw	8.46	4.2	8.67	5.6	8.55	3.9	7.41	12.6	6.82	10.2	**7.01**	**8.4**
	$elec$	7.87	8.6	9.35	8.0	8.52	6.9	5.09	14.5	**9.44**	**7.0**	6.48	12.0
	$disc$	9.92	0.7	9.90	0.4	9.91	0.4	9.63	3.4	9.74	3.3	9.67	2.7
	soc	9.79	1.8	9.10	3.4	9.43	2.1	**8.93**	**7.5**	6.21	11.6	7.27	9.7
(5)	fw	8.39	4.2	8.67	5.8	8.51	3.9	**7.44**	**13.0**	6.81	10.2	**7.01**	**8.5**
	$elec$	7.91	8.6	**9.60**	**5.7**	8.65	6.1	5.09	14.2	9.42	7.2	6.48	11.8
	$disc$	**9.96**	**0.5**	9.90	0.4	**9.93**	**0.3**	9.63	3.3	9.74	3.3	9.68	2.7
	soc	9.80	1.8	9.11	3.2	9.44	2.0	8.90	7.7	**6.24**	**11.8**	**7.28**	**9.9**
(6)	fw	8.52	3.8	8.80	5.7	8.65	3.8	6.97	12.6	**7.12**	**10.9**	6.93	7.9
	$elec$	8.03	8.5	9.46	7.3	8.66	6.5	**5.54**	**16.5**	9.27	7.7	**6.78**	**13.0**
	$disc$	9.93	0.7	9.90	0.4	9.91	0.4	9.47	4.1	9.73	3.1	9.59	2.9
	soc	9.83	1.8	9.13	3.1	9.46	1.9	8.46	11.8	5.13	16.2	6.26	15.5

TABLE II
COMPARISON BETWEEN THE GRAPHLETS KEPT FOR THE EMBEDDINGS.

		Precision		Recall		F1-score	
		mean	std	mean	std	mean	std
3-node only	fw	8.85	4.6	8.99	5.3	8.90	3.6
	$elec$	8.27	9.8	8.72	9.4	8.45	7.8
	$disc$	9.88	0.8	**9.94**	**0.4**	**9.91**	**0.4**
	soc	9.82	1.8	9.25	3.9	9.52	2.3
4-node only	fw	8.45	4.3	8.97	4.8	8.69	3.1
	$elec$	8.27	8.3	9.48	6.5	8.81	5.8
	$disc$	9.95	0.6	9.90	0.4	9.93	0.3
	soc	9.84	1.6	9.13	3.4	9.47	2.1
5-node only	fw	**8.90**	**4.3**	**9.43**	**4.1**	**9.15**	**3.1**
	$elec$	**8.67**	**8.4**	**9.58**	**4.8**	**9.07**	**5.0**
	$disc$	**9.97**	**0.4**	9.84	0.8	9.91	0.4
	soc	**9.99**	**0.4**	**9.56**	**3.1**	**9.77**	**1.7**
3-4 node	fw	**9.31**	**2.7**	**10**	**0**	**9.64**	**1.5**
	$elec$	**10**	**0**	8.97	9.6	9.43	5.6
	$disc$	9.89	0.8	**9.92**	**0.5**	**9.91**	**0.4**
	soc	**10**	**0.2**	**9.66**	**2.0**	**9.83**	**1.0**
3-4-5 node	fw	8.87	3.8	**10**	**0**	9.39	2.1
	$elec$	**10**	**0**	**9.17**	**8.2**	**9.55**	**4.8**
	$disc$	**9.92**	**0.6**	9.89	0.5	**9.91**	**0.4**
	soc	**10**	**0**	9.21	3.8	9.59	2.0

for 20 features. Indeed, there is no further variation in $F1$-score nor in standard deviation when extracting more than 20 features. We also observe that both the best $F1$-score and the best standard deviation are obtained by extracting 18 and 19 features (the results are practically identical). We thus choose to extract 18 features in our feature extraction procedure.

V. TUNING ALTERNATIVES

The purpose of this section is to describe how we have chosen the parameters in existing algorithms to build the comparisons in Table 3 in the main paper. We outline these methods, conduct tests on different parameter sets, and discuss the results.

A. *graph2vec*

This method, proposed in [53], is a neural embedding framework that aims to learn network embeddings in an unsupervised fashion. It is based on two fundamental tools: the Weisfeiler–Lehman (WL) test, used to define the context of a node, and an unsupervised learning process inspired by `doc2vec`, to derive graph embeddings. The WL test is a relabelling process that aims to decide whether two graphs are isomorphic. Each node starts with its own label (the standard choice for a directed graph is a pair containing the in- and out-degree), and then receives a list of its neighbours' labels. A new label is created for this node, two nodes having the same labels if and only if their previous labels and the list they received are equal. These steps are repeated d times, so that at the end of the process, two nodes have the same labels if and only if differences between their d-hop neighbourhoods are indiscernible.

`doc2vec` learns document embeddings by learning word embeddings from the word context (i.e. words coming before and/or after the word of interest), and uses these embeddings to derive a vector representation of a document that maximises the log-likelihood of words in this document.

By considering a network as a document with nodes playing the role of words, and their d-hop neighbourhood playing the role of context, the authors of [53] produce task-agnostic embeddings for networks within a corpus.

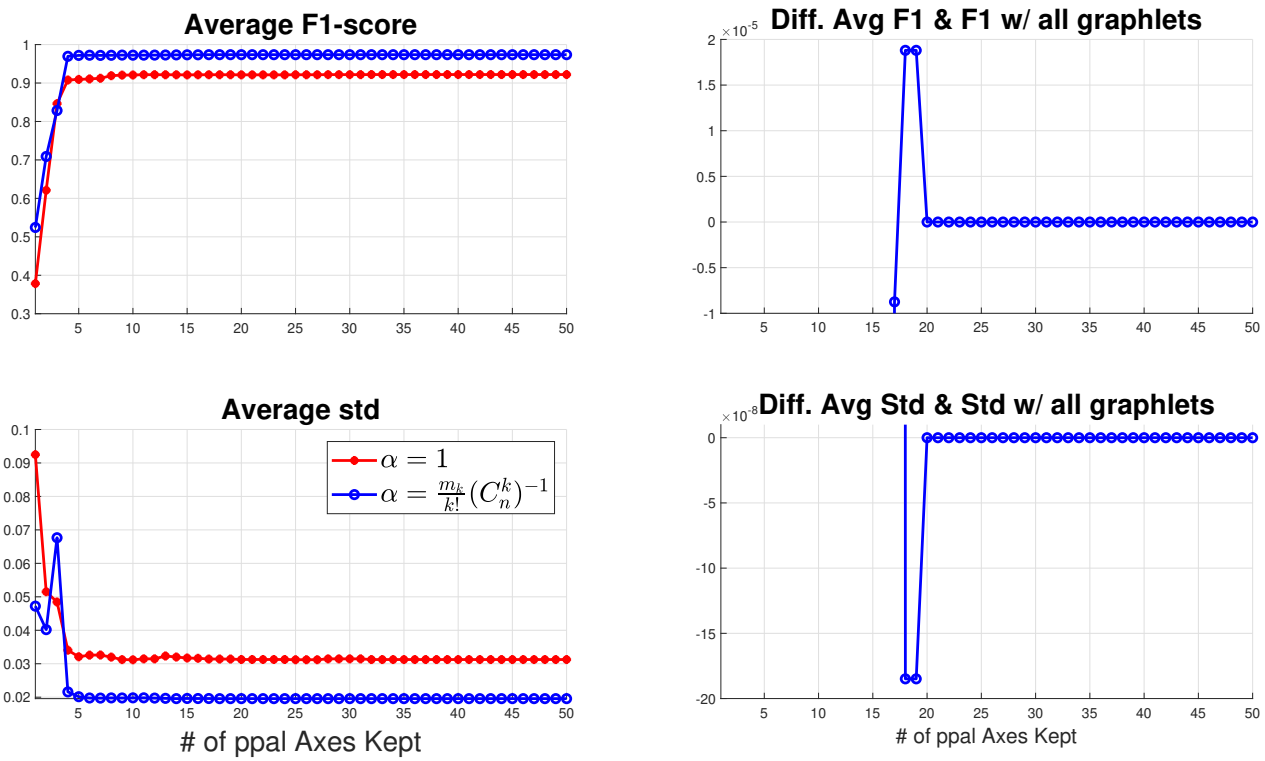

Fig. 4. Right: Improvement brought by the normalisation step. Left: Using 18 features achieve as good results as using 212.

To produce embeddings using `graph2vec`, we have used the implementation from [54]. Some parameters need to be set up in `graph2vec`, the most important being the depth of context to take into account (the d in the WL part), and the size of the embeddings produced by `doc2vec`. We tested with different sizes for both. For other parameters we have used the default values from [54]. The $F1$-scores of the classifier from (1.1) in the main paper applied to these embeddings are summarised in Table III. As for the previous score tables, we have computed the scores of the classifier for the 50 training/test splits.

From Table III, we observe that the best results are provided by a depth of 1 for the WL test. Except for the discourse structure benchmarks for which all depths provide very accurate results, other benchmarks are poorly detected by a 2-depth WL test, and this becomes worse with a 3-depth WL test, whatever the size of the embeddings. Our interpretation of these results is that deep WL tests are not able to capture similarity between networks from our benchmarks, which we believe is due to the wide range of number of nodes and edges of networks in a class—see Table 1 from the main paper. Indeed, the WL test is designed such that networks that are far from isomorphic have very different labelling [55]. The fact that similar networks have similar node labellings is a side product. However, two networks with different number of nodes cannot be isomorphic, even when they have a very similar structure—for instance, random graphs generated using the same random

TABLE III
$F1$-SCORE OF `graph2vec` FOR DIFFERENT EMBEDDING SIZES (ROWS) AND WL TEST DEPTHS(COLUMNS).

Size Emb.		depth 1 F1-score mean	depth 1 F1-score std	depth 2 F1-score mean	depth 2 F1-score std	depth 3 F1-score mean	depth 3 F1-score std
8	fw	9.45	3.1	7.63	5.0	4.39	7.5
	$elec$	9.75	2.9	8.94	5.8	4.42	11.2
	$disc$	9.94	0.4	9.95	0.3	9.82	0.6
	soc	9.38	3.7	8.35	3.3	7.03	8.4
16	fw	9.38	3.0	7.67	4.5	4.42	6.7
	$elec$	9.81	2.2	8.97	5.1	4.35	9.1
	$disc$	9.96	0.3	9.96	0.3	9.87	0.6
	soc	9.40	3.1	8.38	4.7	7.04	3.1
32	fw	9.60	2.5	7.71	4.6	4.14	6.3
	$elec$	9.88	2.0	8.61	5.9	4.15	9.7
	$disc$	9.94	0.3	9.97	0.3	9.87	0.6
	soc	9.52	2.6	8.39	4.2	7.11	2.19
64	fw	9.63	2.5	7.56	4.8	4.58	6.1
	$elec$	9.86	2.0	8.61	6.3	4.46	9.0
	$disc$	9.95	0.3	9.97	0.3	9.89	0.5
	soc	9.55	2.5	8.37	4.5	7.12	3.0
128	fw	9.62	2.5	7.39	5.0	4.62	6.1
	$elec$	9.86	2.0	8.34	7.0	4.47	9.2
	$disc$	9.95	0.3	9.97	0.3	9.89	0.5
	soc	9.57	2.4	8.32	4.6	7.11	3.0
256	fw	9.64	2.1	7.50	5.2	4.78	5.9
	$elec$	9.77	3.1	8.64	6.7	4.69	8.1
	$disc$	9.95	0.3	9.97	0.3	9.89	0.5
	soc	9.61	2.1	8.37	4.2	7.06	2.9

process but with different number of nodes. This interpretation is supported by the fact that the discourse structure benchmark, for which the 3-depth WL provides accurate results, is also the benchmark with the smallest range of network sizes. On the other hand, given a 1-depth WL test, increasing the size of the embeddings tends to provide more accurate results. We observe a stabilisation at size 32: the classification process of larger size embeddings only slightly improve the results.

B. (R)GCN

Graph convolutional networks (GCNs) are an instance of graph neuronal networks, proposed in [56]. They aim to produce node embeddings for some downstream classification task, such as clustering or community detection. A GCN is composed of k Graph Convolutional Layers, as shown in Figure 5. Each layer i can be written as:

$$\mathbf{H}^{(i+1)} = \sigma(\mathbf{M}\mathbf{H}^{(i)}\mathbf{W}^{(i)}) \tag{V.1}$$

with σ some non-linear function (typically $reLU$), $\mathbf{M} \in \mathbb{R}^{n\times n}$ represents the graph structure, $\mathbf{H}^{(i)} \in \mathbb{R}^{n\times d_i}$ whose rows contain the ith hidden embeddings of nodes, and $\mathbf{W}^{(i)} \in \mathbb{R}^{d_i\times d_{i+1}}$ the weight matrices to be learnt. Generally, $\mathbf{M}$ is some transformation of the adjacency matrix of the graph[4] $\mathbf{A} \in \mathbb{R}^{n\times n}$. Classical choices are $\mathbf{M} = \mathbf{D}_{in}^{-1/2}\mathbf{A}\mathbf{D}_{out}^{-1/2}$ (both-side normalisation) or $\mathbf{M} = \mathbf{A}\mathbf{D}_{in}^{-1}$ (right-side normalisation), where $\mathbf{D}_{in}$ and $\mathbf{D}_{out}$ are diagonal matrices of in- and out-degrees. $\mathbf{H}^{(0)}$ contains the initial node features. If no feature is known, $\mathbf{H}^{(0)}$ can be a vector containing nodes' degrees, an $n \times 2$ matrix containing node in/out degrees, or the identity matrix.

Relational GCNs (RGCNs) [57] have been adapted from GCNs to deal with graphs with different kinds of relations—and thus represented by a set of adjacency matrices $\{\mathbf{M}_r \in \mathbb{R}^{n\times n}\}_{r\in\mathcal{R}elations}$—by allowing some weights to be different for each relation. Here, we use RGCNs as a tool to take into account the fact that our networks are directed. That is, we consider that a directed edge from node u to node v results in two different kinds of relations: u "points to" v; and v "is pointed by" u. Thus, the adjacency matrices of our relations are actually the adjacency matrix of the directed graph and its transpose.

To classify graph instances instead of nodes from a (R)GCN, one has to add a layer to merge node embeddings, such as a pooling layer, and a dense layer that returns a vector whose dimension is equal to the number of classes [55]. Using a pooling layer has the benefit that we can avoid learning new variables. Several layers have been tested, as well as different numbers of Graph Convolutional Layers and hidden embedding sizes. As node features, we have taken the in/out degrees of nodes. Matrix $\mathbf{M}$ has been chosen as the both-side normalisation. Indeed, both normalisations have been tested and provide similar results. We also test our GCNs without normalisation, which provides a much less accurate model. Implementation has been done using the DeepGraph Library (https://www.dgl.ai/), with `pytorch` as backend. We use the same 40 networks per class for training the neuronal networks, 5 of which are kept for validation purpose (early stopping if the loss of the validation set does not decrease for 20 successive iterations). Other hyper parameters have been chosen as in [58], namely: the non-linear function (σ in (V.1)) is a $ReLU$, all hidden layers return embeddings of same dimension, we use the Adam algorithm for gradient descent, with a learning rate of 0.01 and weight decay for the L_2 penalty equal to 5×10^{-4}, and the dropout of each layer is 0.5.

Table IV summarises the average $F1$-scores obtained by different (R)GCN architectures in classifying the networks of our dataset. Due to the long runtime necessary for training the models, we test each architecture on just 10 training/test sets. For some sets, some models were unable to detect a class (that is, the classifier returns an empty class). Such failures are displayed in the Table. In the average of $F1$-scores, we do not take into account the iteration at which a model does not detect a class. If the model has failed for 8 iterations or more, only its maximal $F1$-score is displayed. Best and second best results for each class and each type of pooling are highlighted in by a dark, respectively bright, background. When the model has failed for 8 or more iterations, it is not highlighted..

We note the following. First, the sum pooling shows very poor results compared to maximum and mean pooling layers. Most of the time, it fails to detect at least one class (see the corresponding *KO* columns). This can be correlated to the observation about deep WL tests in `graph2vec`. Indeed, it has been proved in [55] that a sum pooling at the end of a GCN is more able to discriminate among non-isomorphic networks than mean or maximum poolings. As explained before, in our dataset, networks from the same class are not expected to be isomorphic. Hence, the more an architecture discriminates among non-isomorphic networks, the worse its results on our dataset. This is also confirmed by the fact that for RGCNs, the unique sum pooling architecture able to consistently detect all classes has a unique Graph Convolutional Layer, meaning that, before the pooling, embedding from a node only depends on its initial embedding and those from its neighbours. We also remark that for maximum and mean poolings the architectures with 3 hidden layers rarely provide the best results which, once again, should mean that embedding of a node that considers a wide portion of the network jeopardises the classifier consistency.

Both mean and maximum poolings poorly detect the electronic circuit benchmark but are quite efficient at classifying discourse structures. Globally, mean pooling is more accurate than maximum pooling (mean pooling achieves better results than maximum pooling at least twice as often as the opposite, for both GCNs and RGCNs), even if maximum pooling noticeably better detects electronic circuits.

Finally, we remark than RGCNs outperform GCNs, which was expected since we have used RGCNs to take advantage of the direction of edges in the networks from our dataset, which

[4]More specifically, the graph obtained by adding a self loop on each node, which allows to consider the previous embedding of a node when computing its new embedding in a layer.

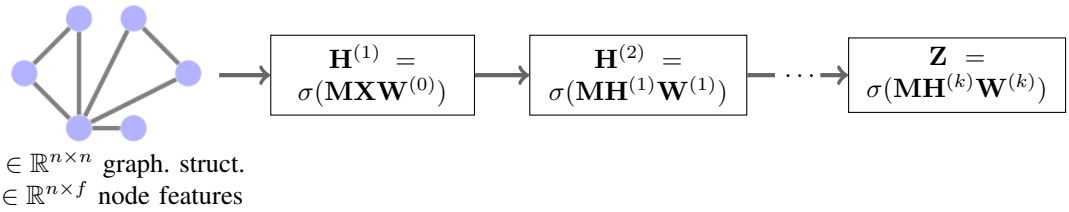

Fig. 5. Pipeline of a GCN.

a simple GCN is not able to do.

VI. Parameters for Feature Selection Based on Random Forests

In Section 2.2 of the main paper, we propose an algorithm based on a graphlet count followed by a feature selection procedure. We tested it against another feature selection procedure based on random forests (RF) [59] as described below.

First, we have embedded the networks $G_i, \forall i \in \mathcal{I}$ to obtain their vector representations $\mathbf{x}_i, \forall i \in \mathcal{I}$:

$$\forall i \in \mathcal{I}, \quad \mathbf{x}_i = \frac{\boxplus_{k=3,4} \alpha_k(G_i)\phi_k(G_i)}{\sqrt{\sum_{k=3,4} \alpha_k(G_i)^2 \phi_k(G_i)^T \phi_k(G_i)}}, \qquad \text{(VI.1)}$$

where $\phi_k(G) \in \mathbb{R}^{m_k}$ is the vector such that $\phi_k(G)(p)$ is the number of occurrences of the pth k-node graphlet in G and $\alpha_k(G)$ is the normalisation factors. We use the symbol $\boxplus$ to represent the concatenation of vectors.

We have chosen to apply the RF-based feature selection on network embeddings provided by (VI.1) because these are also the embeddings we use in our own feature selection procedure. The results when we applied RF-based feature selection simply on $\boxplus_{k=3,4} \phi_k(G)$ (normalised or not) were much worse than those we present in the main paper.

To measure a RF-based feature significance, for each training set $\mathcal{S}$, we have trained a RF and recovered the Gini importance—or mean decrease in impurity—of each graphlet according to the forest [60]. The Gini importance of a forest being the average Gini importance over all trees. We have used the random forest implementation proposed in the python library `scikit-learn` [61] with all default parameters—essentially, 100 trees are built in the forest, and the Gini impurity is used to measure the quality of a split in a tree.

We have kept the 9 graphlets with highest Gini importance, that is, the same number of graphlets we kept for our procedure.

VII. Towards Unsupervised Clustering

In Section 5.1 of the main paper, we predict that graphlets selected via our feature selection procedure may be able to discriminate fields of networks that were not used in the learning process. To support this claim, we supplement our database with unseen random network, and apply the unsupervised algorithm from [62] to recover the clusters. Some technical details are provided here.

A. LFR Benchmark

To support our claim, we supplement our database with 60 directed networks built using the graph generator from [63]. Namely, we used the "Package 3" implementation from https://www.santofortunato.net/resources. This generator was initially designed to test community detection algorithms by generating random modular networks with features observed in real-world networks (especially an heterogeneous distributions of node degrees and community sizes, described by power laws). Parameter choices are summarised in Table V.

B. Affinity matrix

To uncover the network clusters, we have applied the unsupervised clustering algorithm from [62]. This requires the dataset to be in an affinity matrix style. That is, a dataset containing n samples has to be represented by a matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ where $\mathbf{A}(i,j)$ expresses the proximity (or affinity) between sample i and sample j. To do this here we have used the affinity matrix of our embedded networks $\mathbf{x}_i, i \in \mathcal{I}$ built with the embedding technique from Section 2.2. The affinity matrix of a set of points $\{\mathbf{x}_i \in \mathbb{R}^p, i = 1, \ldots, n\}$ being the symmetric matrix with zeros on the diagonal and

$$\mathbf{A}(i,j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right),$$

elsewhere. The affinity matrix strongly depends on the choice of the Gaussian parameter σ. We follow the prescription in [64] to take into account both density and dimension of the dataset.

VIII. Remark

Tables and Figures from the main paper can be reproduced by using the Matlab code and dataset at http://github.com/luleg/DiscriminantMotifs. A curated version of the networks described in Section II is also provided.

References

[1] U. of Canberra. https://www.globalwebdb.com/. Access: February 2020.

[2] J. Dunne, K. Lafferty, A. Dobson, R. Hechinger, A. Kuris, N. Martinez, and e. a. McLaughlin, "Parasites affect food web structure primarily through increased diversity and complexity," *PLoS biology*, vol. 11, no. 6, 2013.

[3] C. D. Zander, N. Josten, K. C. Detloff, R. Poulin, J. P. McLaughlin, and D. W. Thieltges, "Food web including metazoan parasites for a brackish shallow water ecosystem in germany and denmark: Ecological archives e092-174," *Ecology*, vol. 92, no. 10, pp. 2007–2007, 2011.

[4] V. Batagelj and A. Mrvar, "Pajek datasets." http://vlado.fmf.uni-lj.si/pub/networks/data/, 2006. Access: March 2020.

GCNs		Mean Pooling					Max Pooling					Sum Pooling				
Lay.	Emb.	$F1$-scores				KO	$F1$-scores				KO	$F1$-scores				KO
		fw	$elec$	$disc$	soc		fw	$elec$	$disc$	soc		fw	$elec$	$disc$	soc	
1	4	8.07	3.46	8.11	8.07	5	7.06	5.56	7.71	5.49	2	×	×	×	×	10
	8	8.46	**5.53**	**9.85**	**9.05**	2	8.04	**7.92**	9.38	**8.17**	2	5.41	3.53	7.34	5.57	6
	12	8.17	5.17	9.82	**8.98**	0	**8.15**	**8.00**	**9.70**	**8.20**	0	8.52	1.38	1.13	7.19	8
2	4	8.12	3.00	8.80	8.00	2	7.22	5.52	9.64	6.97	7	2.63	7.5	0.97	3.32	9
	8	8.25	4.30	9.68	8.75	0	8.57	4.71	9.78	8.64	9	4.87	2.63	4.43	5.93	7
	12	**8.51**	**5.66**	**9.86**	**8.98**	1	8.45	5.0	7.32	4.76	8	8.39	0.43	3.33	8.51	8
3	4	5.46	2.93	9.36	7.01	4	6.12	2.67	8.99	6.87	8	4.50	3.53	9.63	4.48	9
	8	8.23	3.47	9.19	8.57	0	7.99	4.28	**9.78**	7.60	6	4.15	1.54	9.57	4.91	9
	12	**8.58**	5.33	9.82	8.88	0	**8.60**	4.80	9.25	7.39	4	8.28	6.67	9.63	8.72	8

RGCNs		Mean Pooling					Max Pooling					Sum Pooling				
Lay.	Emb.	$F1$-scores				KO	$F1$-scores				KO	$F1$-scores				KO
		fw	$elec$	$disc$	soc		fw	$elec$	$disc$	soc		fw	$elec$	$disc$	soc	
1	4	8.64	4.52	8.40	8.53	2	7.66	5.81	9.63	6.77	2	×	×	×	×	10
	8	9.20	5.82	9.66	8.86	1	8.06	**7.82**	9.57	7.97	0	9.15	9.23	9.54	6.67	9
	12	**9.45**	7.23	**9.90**	8.90	1	8.40	**8.33**	**9.81**	**8.42**	0	6.54	1.24	2.49	4.81	9
2	4	9.22	5.90	9.82	8.74	1	6.67	2.82	6.21	6.18	5	6.19	3.08	9.74	0.93	8
	8	**9.43**	**7.71**	**9.88**	**9.23**	2	7.82	5.51	7.77	7.47	3	0.65	1.62	3.11	4.66	9
	12	9.41	**7.28**	9.83	8.98	1	**8.67**	7.68	**9.73**	**9.04**	1	5.49	1.58	9.09	7.52	8
3	4	9.17	5.58	9.78	8.86	2	7.31	4.98	8.82	6.43	5	3.67	5.22	9.27	4.74	8
	8	9.33	6.34	9.36	9.12	0	8.62	3.72	9.49	7.66	2	3.64	3.85	9.13	2.25	8
	12	9.40	6.50	8.85	**9.33**	0	**8.99**	6.02	9.61	7.97	1	4.54	5.76	8.07	3.83	7

TABLE IV

$F1$-SCORES OF CLASSIFIERS BUILT ON DIFFERENT (R)GCNS ARCHITECTURES. COLUMNS *Lay.* AND *Emb.* INDICATE THE NUMBER OF GRAPH CONVOLUTIONAL LAYERS AND THE SIZE OF HIDDEN EMBEDDINGS, RESPECTIVELY. COLUMNS *KO* GIVE THE NUMBER OF TIME THE CLASSIFIER CONSIDERS AT LEAST ONE CLASS AS EMPTY.

nb of nodes	mixing param. μ	exp. of power law for community size: t_2	average in-deg. $\overline{k}$	max. in-deg. k_{max}
$n \times 1000$, $n = 1..5$	0.6	-1	5	100
	0.6	-2	5	100
	0.8	-1	5	100
	0.8	-1	5	50
	0.8	-1	10	50
	0.8	-1	10	100

TABLE V

TABLE OF PARAMETERS USED TO GENERATE RANDOM GRAPHS WITH LFR GENERATOR. GIVEN A SET OF PARAMETERS $(n, \mu, t_2, \overline{k}, k_{max})$, TWO RANDOM NETWORKS HAVE BEEN GENERATED. DETAILS ABOUT THE PARAMETERS CAN BE FOUND IN [63].

[5] J. Kunegis, "KONECT – The Koblenz Network Collection," in *Proc. Int. Conf. on World Wide Web Companion*, pp. 1343–1350, 2013.

[6] N. D. Martinez, "Artifacts or attributes? effects of resolution on the little rock lake food web," *Ecological monographs*, vol. 61, no. 4, pp. 367–392, 1991.

[7] A. Clauset, E. Tucker, and M. Sainz, "The Colorado index of complex networks." https://icon.colorado.edu/, 2016. Access: February 2020.

[8] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *IEEE International Symposium on Circuits and Systems*, pp. 1929–1934, 1989.

[9] F. Corno, M. S. Reorda, and G. Squillero, "Rt-level itc'99 benchmarks and first atpg results," *IEEE Design & Test of computers*, vol. 17, no. 3, pp. 44–53, 2000.

[10] M. Jenihhin. http://pld.ttu.ee/~maksim/benchmarks/iscas89/bench/, 2020. Access: February 2020.

[11] N. Asher and A. Lascarides, *Logics of Conversation*. Cambridge University Press, 2003.

[12] A. Lascarides and N. Asher, *Segmented Discourse Representation Theory: Dynamic Semantics With Discourse Structure*, pp. 87–124. Springer, 2007.

[13] M. Qi, "Use of deep learning approaches for the prediction of discourse structures," Master's thesis, Université Paul Sabatier, Toulouse, 2018.

[14] N. Asher, J. Hunter, M. Morey, F. Benamara, and S. Afantenos, "Discourse structure and dialogue acts in multiparty dialogue: the stac corpus," in *LREC*, pp. 2721–2727, 2016.

[15] J. Moody, "Peer influence groups: Identifying dense clusters in large networks," *Social Networks*, vol. 23, pp. 261–283, 10 2001.

[16] L. C. Freeman. http://moreno.ss.uci.edu/data.html, 2020. Access: February 2020. Out-of-date.

[17] J. S. Coleman, *Introduction to Mathermatical Sociology*, pp. 450–451. New York: Free Press, 1964.

[18] C. M. W. L. C. Freeman and D. M. Kirke, "Exploring social structure using dynamic three-dimensional color images," *Social Networks*, vol. 20, pp. 109–118, 1998.

[19] C. Baerveldt, "Pupil's networks in high schools. network sampling, program and some results from a theory-oriented research project on petty crime of pupils," in *Paper for the 2nd International Network Sampling Workshop*, 2000.

[20] http://www.stats.ox.ac.uk/~snijders/siena/BaerveldtData.html, 2020. Access: March 2020.

[21] E. Lazega, *The Collegial Phenomenon: The Social Mechanisms of Cooperation Among Peers in a Corporate Law Partnership*. Oxford University Press, 2001.

[22] J. Coleman, E. Katz, and H. Menzel, "The diffusion of an innovation among physicians," *Sociometry*, vol. 20, no. 4, pp. 253–270, 1957.

[23] R. Cross and A. Parker, *The Hidden Power of Social Networks*. Harvard Business School Press, Boston, MA, 2004.

[24] http://toreopsahl.com/datasets, 2020. Access: March 2020.

[25] M. R. Weeks, S. Clair, S. P. Borgatti, K. Radda, and J. J. Schensul, "Social networks of drug users in high-risk sites: Finding the connections," *AIDS and Behaviour*, vol. 6, pp. 193–206, 2002.

[26] http://sites.google.com/site/sfeverton18/research/cohesion-and-clustering, 2020. Access: March 2020.

[27] M. J., "Direct factor analysis of sociometric data," *Sociometry*, vol. 23, pp. 360–371, 1960.

[28] C. P. Loomis, J. O. Morales, R. A. Clifford, and O. E. L. Turrialba, *Social Systems and the Introduction of Change*. Glencoe (Ill.): The Free Press, 1953.

[29] . V. B. W. de Nooy, A. Mrvar, *Exploratory Social Network Analysis with Pajek*, ch. 3. Cambridge: Cambridge University Press, 2004.

[30] M. Fire, R. Puzis, and Y. Elovici, "Link prediction in highly fractional data sets," *Handbook of Computational Approaches to Counterterrorism*, 2012.

[31] http://proj.ise.bgu.ac.il/sns/, 2020. Access: March 2020.

[32] J. Kunegis, A. Lommatzsch, and C. Bauckhage, "The slashdot zoo: Mining a social network with negative edges," in *Proceedings of the 18th International Conference on World Wide Web*, p. 741–750, 2009.

[33] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters," *Internet Mathematics*, vol. 6, no. 1, pp. 29–123, 2009.

[34] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection." http://snap.stanford.edu/data, June 2014. Access: February 2020.

[35] S. Kumar, W. L. Hamilton, J. Leskovec, and D. Jurafsky, "Community interaction and conflict on the web," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pp. 933–943, International World Wide Web Conferences Steering Committee, 2018.

[36] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *The Semantic Web-ISWC*, pp. 351–368, Springer, 2003.

[37] P. Massa, M. Salvetti, and D. Tomasoni, "Bowling alone and trust decline in social network sites," in *Proc. Int. Conf. Dependable, Autonomic and Secure Computing*, pp. 658–663, 2009.

[38] S. Kumar, F. Spezzano, V. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Data Mining (ICDM), 2016 IEEE International Conference on*, 2016.

[39] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, (New York, NY, USA), p. 333–341, Association for Computing Machinery, 2018.

[40] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *ACM Workshop on Online Social Networks*, p. 37–42, 2009.

[41] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 555–564, 2017.

[42] R. Michalski, S. Palus, and P. Kazienko, "Matching organizational structure and social network extracted from email communication," in *Int. Conf. on Bus. Inf. Syst.*, pp. 197—-206, 2011.

[43] D. Homscheid, J. Kunegis, and M. Schaarschmidt, "Private-collective innovation and open source software: Longitudinal insights from linux kernel development," in *Open and Big Data Management and Innovation* (M. Janssen, M. Mäntymäki, J. Hidders, B. Klievink, W. Lamersdorf, B. van Loenen, and A. Zuiderwijk, eds.), (Cham), pp. 299–313, Springer International Publishing, 2015.

[44] T. Opsahl and P. Panzarasa, "Clustering in weighted networks," *Soc. Netw.*, vol. 31, pp. 155—163, 2009.

[45] P. Sapiezynski, A. Stopczynski, L. David Dreyer, and S. Lehmann, "Interaction data from the Copenhagen Networks Study," *Scientific Data*, vol. 6, no. 1, 2019.

[46] C. Morselli, *Inside criminal networks*. New York: Springer, 2009.

[47] http://sites.google.com/site/ucinetsoftware/datasets/covert-networks/caviar, 2020. Access: March 2020.

[48] A. Topirceanu, A. Duma, and M. Udrescu, "Uncovering the fingerprint of online social networks using a network motif based approach," *Computer Communications*, vol. 73, pp. 167–175, 2016.

[49] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon, "Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs," *Bioinformatics*, vol. 20, no. 11, pp. 1746–1758, 2004.

[50] S. Theodoridis, *Machine learning: a Bayesian and optimization perspective*. Academic press, 2015.

[51] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[52] Q. Gu, Z. Li, and J. Han, "Linear discriminant dimensionality reduction," in *Machine Learning and Knowledge Discovery in Databases* (D. Gunopulos, T. Hofmann, D. Malerba, and M. Vazirgiannis, eds.), pp. 549–564, Springer Berlin Heidelberg, 2011.

[53] N. Annamalai, C. Mahinthan, V. Rajasekar, C. Lihui, L. Yang, and J. Shantanu, "graph2vec: Learning distributed representations of graphs," in *International Workshop on Mining and Learning with Graphs*, 2017.

[54] B. Rozemberczki, O. Kiss, and R. Sarkar, "Karate Club: An API Oriented Open-source Python Framework for Unsupervised Learning on Graphs," in *ACM International on Conference on Information and Knowledge Management*, 2020.

[55] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *International Conference on Learning Representations*, 2019.

[56] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017.

[57] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European semantic web conference*, pp. 593–607, Springer, 2018.

[58] T. N. Kipf *et al.*, *Deep learning with graph-structured representations*. 2020.

[59] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[60] S. Nembrini, I. R. König, and M. N. Wright, "The revival of the Gini importance?," *Bioinformatics*, vol. 34, pp. 3711–3718, 05 2018.

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

[62] L. le Gorrec, S. Mouysset, I. S. Duff, P. A. Knight, and D. Ruiz, "Uncovering hidden block structure for clustering," in *Machine Learning and Knowledge Discovery in Databases*, pp. 140–155, 2020.

[63] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Phys. Rev. E*, vol. 80, p. 016118, Jul 2009.

[64] S. Mouysset, J. Noailles, and D. Ruiz, "Using a global parameter for gaussian affinity matrices in spectral clustering," in *VECPAR*, vol. 5336 of *Lecture Notes in Computer Science*, pp. 378–390, Springer, 2008.