MODEL SELECTION FOR SOCIAL NETWORKS USING GRAPHLETS

JEANNETTE JANSSEN, MATT HURSHMAN, AND NAUZER KALYANIWALLA

Abstract. Several network models have been proposed to explain the link structure observed in online social networks. This paper addresses the problem of choosing the model that best fits a given real world network. We implement a model selection method based on un-supervised learning. An alternating decision tree is trained using synthetic graphs generated according to each of the models under consideration. We use a broad array of features, with the aim of representing different structural aspects of the network. Features include the frequency counts of small subgraphs (graphlets) as well as features capturing the degree distribution and small world property. Our method correctly classifies synthetic graphs, and is robust under perturbations of the graphs. We show that the graphlet counts alone are sufficient in separating the training data, indicating that graphlet counts are a good way of capturing network structure. We tested our approach on four Facebook graphs from various American Universities. The models that best fit this data are those that are based on the principle of preferential attachment.

1. Introduction

Recent experimental studies of various types of on-line social networks have revealed many distinguishing features of the link structure of such networks. Examples are recent studies on on-line social networks such as MySpace and Facebook in [1, 30, 35, 39, 48], but also studies of social networks associated with other social media such as Youtube and Flickr [11, 33, 38]. The studies show that social networks share many characteristics of other complex networks, such as a power law degree distribution, high clustering coefficients and small hop distances between individuals.

A number of graph models have been developed to explain the observed link structure of social networks. Notable recent models that are specifically proposed to model social networks can be found in [9, 10, 30, 31, 33]. Most models can successfully replicate some of the observed features of the networks, and provide a mechanism for link formation which is based on plausible principles. As the number of proposed models increases, the question of model selection becomes more important. Our paper addresses this question. The goal of this work is to determine, given a set of models, which of the models is the most likely to have generated a given real online social network.

The best model obviously is the one that generates graphs that are most similar in structure to the observed network. However, no consensus exists on how to determine structural similarity of graphs. The graph features that are replicated

The authors gratefully acknowledge support from NSERC and MITACS grants.

by proposed models, most often are of a global nature: they characterize the network as a whole. Degree distribution, clustering coefficient and average hop distance are all global features. Our model selection method complements the global features with local features, which are features derived from the immediate neighbourhood of the vertex. A similar collection of features is used in the work [20], which uses clustering algorithms to determine if graphs from similar real networks get clustered together.

To characterize the local structure, we will use counts of small subgraphs, also called *graphlets*. Recent work on graph similarity has incorporated graphlet counts as a method of comparing networks, as in [20, 26, 47]. Graphlets have also been used to characterize biological networks, see [37, 43]. To test our hypothesis that graphlet counts characterize the structure of a network, we developed three versions of our model selection method: one based only on global features, one based only on graphlet counts, and a third one based on all features together. We found that the method based on graphlet counts alone performs as well as the full feature set, thus confirming our hypothesis. Our model also supports the conclusions of [20, 36]: that graphlet counts are an efficient way of characterizing networks. Though it is not immediately obvious why this should be so, the most likely explanation is that different models generate vastly different concentrations of small subgraphs even when the models share similar global properties.

Our model selection method is based on machine learning. More precisely, the model selection tool is a multi-class classifier, based on an alternating decision tree. This classifier is trained to distinguish synthetic graphs generated according to six different models. The parameters of the models are chosen randomly, but such that the synthetic graphs will have size and density approximately equal to the network data. Our model selection method is based on the model validation performed by Middendorf *et al.* [36] for protein-protein interaction networks. Our work is different in several ways: (*i*) the social networks we consider are much larger and denser than the PPI networks, (*ii*) we use a different type of decision tree, and (*iii*) we consider a different set of models. We also test the robustness of our classifier: to this end, we generate graphs according to one of the models, perturb some of its edges, and evaluate the performance of the classifier on the perturbed graph. Our results show that our classification method is robust up to a perturbation of 5–10% of the edges.

We apply our classification method to four social networks obtained from Facebook. The vertices correspond to users at four different American universities. Two vertices are connected if they are Facebook "friends". The data was obtained from Mason Porter's Facebook100 data set, first presented in [48]. The graphs have around 7000 vertices, and average degree ranging from 68 to 89. Because of the different average degrees, we generated different training sets, and built a different classifier for each of the four data sets. Our result show a clear preference for models that are based on preferential attachment. We complement the results of our classifier with a close analysis of the feature value profiles for each of the training sets, and a comparison with the feature values of the training data, which show a more nuanced picture.

We have selected our six models such that they represent the most prevalent principles that underlie graph models for complex networks. Of each type of model under consideration, many variations exist in the literature. Such variations are sometimes conceived with the aim of modelling a particular class of networks, or to obtain a particular graph-theoretic property. We note that there is always a trade-off between goodness-of-fit and complexity of the model. For this study, we have consciously chosen to keep the models as simple as possible, while still incorporating enough variability so that each training set contains graphs generated according to the same model with a range of different parameters. Each model has 2 or 3 tunable parameters. The models we consider are based on radically different generational principles. Thus, in finding the best fit of our data to this set of models, we aim to find the principles that drive link formation in on-line social networks.

An important ingredient of many models for complex networks is the principle of *preferential attachment* (PA), first proposed for complex networks by Barabási and Albert in [5]. Under this principle, vertices that already have high degree are more likely to receive an edge from any new vertices that join the network. Models based on the PA principle generally produce graphs with a power law degree distribution. Variations of the preferential attachment model from [5] are proposed and studied by others, see [7, 12] for a survey. We use a PA model proposed by Aiello *et al.* in [4], where the attachment strength, and thus, the exponent of the power law, can be tuned through the inclusion of an additional parameter.

A second principle that has been proposed to explain the specific structure of complex networks is *copy with error*. In a copy or duplication model, a new vertex copies some or all of its neighbours from the neighbourhood of an existing vertex. Copy models have been proposed for the Web graph in [25, 29], for citation graphs in [27], and for biological networks in [6, 13]. The forest fire model proposed in [33] also implicitly incorporates the idea of copying, since the neighbours of a new vertex are chosen from the local environment of an existing vertex.

A central question of this work is whether a *spatial* (or geometric) model is appropriate for our social network data. In a spatial model, vertices are assumed to be embedded in a metric space, and formation of edges is influenced by the metric distance between vertices. (For a recent overview of spatial models for complex networks, see [22]). The advantage of using a geometric model for a social network is that the metric space can be seen as the *social space* representing the interests, hobbies and other attributes of the individuals corresponding to the vertices of the social network. Assuming a geometric model gives the possibility for inference of the social space from the network, thus providing a basis for identifying communities or individuals with similar interests.

The simplest spatial model is the random geometric graph, where vertices can be connected only when their distance is below a given threshold. In the original geometric graph model (see [41]), vertices within threshold distance are always connected. To introduce more variability, we will use an adapted version where pairs of vertices that are within threshold distance are connected independently with a fixed probability. Geometric graphs have been proposed for biological

networks, for example in [42], and have shown to be a good fit in terms of graphlet structure.

We have included a spatial model which incorporates the preferential attachment principle: the Spatial Preferential Attachment (SPA) model proposed in [3]. Here, each vertex is surrounded by its sphere of influence, and new vertices can only link to it if they fall within this sphere of influence. The PA principle is incorporated indirectly, in that the size of the sphere of influence depends on the degree of the vertex. In [23], it is shown how, for graphs obtained from this model, the metric distance between pairs of vertices can be retrieved from the graph structure alone, by considering the degrees of the two vertices, and their number of common neighbours. If the metric embedding is interpreted as modelling the hidden reality of the vertices, then metric distance is a measure of how similar the vertices are. Thus, the SPA model gives a possibility of judging the similarity between vertices based on the graph structure.

The fact that spatial models can be used as a basis for estimating vertex similarity from the graph structure makes them superior to purely graph-theoretic models. We therefore feel that, in cases where the goodness of fit is approximately equal, spatial models should be preferred. The conclusion of this work is that, from among the selected models, those that are built on the principle of preferential attachment have the best fit for the Facebook data. Of the models that give the best fit, one is a standard PA model, the other is the Spatial Preferred Attachment model. In light of the previous discussion, we assert that the SPA model is the most appropriate to model the Facebook data.

The organization of this paper is as follows. In the next section we describe our method in detail, including descriptions of the models, the (Facebook) testing data, the features selected to represent the graphs and the classification algorithm. In Section 3 we present the results of our experiment. We first analyze the performance of our classifier using a test set containing synthetic graphs generated from our training models. We compare the performance of the classifier when only graphlet counts are used as features, when only global features are used, and when the full set of features is used. We also test how robust the classifier is by artificially creating noise in the test data through changes to some of the edges. Finally, we apply the classifier to the Facebook data. We present the results of the classifier, and also analyze the profiles of the features for the different models to understand what distinguishes the different models, and how the results for the Facebook data should be interpreted.

2. The Model Selection Classifier

Our model selection method follows three steps. First, we generate the training data, consisting of 1000 graphs generated according to each of the six models we have selected: the Preferential Attachment Model, the Copy Model, the Random Geometric Model (2D and 3D), and the Spatial Preferred Attachment Model (2D and 3D). The details of the models are given in Section 2.1 below. The parameters of the models are randomly sampled from a range such that the graphs generated are similar in size and density to the test data. The restriction of the sample

range of the parameters is necessitated by the fact that the graphlet counts depend heavily on the size and density of a graph, even for graphs generated by the same model. For this reason it is necessary to generate a new training set for each test graph.

Next, we use the training data to build a multi-class alternating decision tree (ADT). The details of the construction of the ADT are given in Section 2.3 below. We represent the graphs using features that capture both the local structure of the graph, through the graphlets, and the global structure. A description of the features is given in Section 2.2 below.

Finally, we compute the feature vectors corresponding to each network from our on-line social network data, in this case snapshots of Facebook. Running this feature vector through the classifier gives a score for each model corresponding to how well the model fits the test data. Our experimental procedure is repeated for four different Facebook networks taking from the following American universities: Princeton, American University, MIT and Brown. We obtained this data from the Facebook100 data set compiled by Porter *et al.* and introduced in [48].

2.1. **Models.** We have implemented six different graph models. As explained in the introduction, our choice of models was motivated by the desire to test a wide range of models commonly proposed for social networks, based on a number of different attachment principles. Special attention was given to spatial models, a class of models that is gaining support because of the ability to model vertex attributes through spatial representation. Wherever more than one variation of the model has been proposed in the literature, we have opted for the simpler versions. This choice was motivated by the wish to avoid ambiguity in the classification.

All model generation algorithms are written in Python using the graph-tool module [45]. Our training set includes only undirected graphs without multiple edges. Some of the models allow for multiple edges; if this occurs, we remove the multiple copies. For all models under consideration, this is known to affect only a tiny fraction of the edges. The SPA model and COPY model are formulated to generate directed graph; here we ignore the direction of the edges after generation.

Preferential Attachment Model (PA). The Preferential Attachment model was first introduced by Barabási and Albert in [5] as a model for the World Wide Web. In the original version, the model has only one parameter, namely the initial degree of each vertex. We use here a more general model introduced by Aiello *et al.* in [4]. Preferential attachment models are built on the concept that a new user is more likely to join to a user that already has many incident edges.

Our PA model has two parameters, $d \in \mathbb{Z}^+$ and $\alpha \in [0, d]$. It generates a sequence of graphs $\{G_t : t \in \mathbb{N}\}$ where $G_0 = H$ is a small random seed graph. We take H to be an Erdös-Rényi random graph $G(100, ed)$, where ed is the edge density of the test graph. At each time step $t > 0$, G_t is formed by adding a new vertex v_t and adding d edges (v_t, w_i) where w_i is chosen randomly, with probability proportional to its degree plus α. The probability that w_i is chosen is given by:

$$P(w = w_i) = \frac{deg_{G_{t-1}}(w) + \alpha}{2d(t-1) + 2|E(H)| + \alpha(t-1)},$$

where $deg_{G_{t-1}}(w)$ is the degree of w in G_{t-1}. Thus, vertices of high degree are more likely to accumulate more edges.

Copy Model (COPY). The Copy Model was originally proposed in [25] as a model for the World Wide Web. The idea behind this model is that a person tends to meet friends through a currently existing friend, and thus their frienship neighbourhood will have large overlap with that of the friend he or she "copied" from. Our version allows for a number of neighbours that are chosen at random, not copied. This version of the copy model is used in [2, 8].

The COPY model has two parameters $p \in (0, 1)$ and $d \in \mathbb{Z}^+$. It generates a sequence of directed graphs $\{G_t : t \in \mathbb{N}\}$; the direction of the edges will be ignored for the model selection. Again, $G_0 = H$ is a small random seed graph. We take H to be the directed version of the Erdös-Rényi random graph with 100 vertices and edge probability equal to the edge density of the test graph. At each time step $t > 0$, we add a new vertex v_t. We then choose a vertex w uniformly at random from G_{t-1} and for each out-neighbour u of w, independently, we add an edge from v_t to u with probability p. We then choose d more vertices uniformly at random from G_{t-1}, and add directed edges from v_t to each of these vertices.

Random Geometric Model (GEO). The Random Geometric Model is a model where the vertices are embedded in a metric space and edges are determined by a threshold on the distance between two vertices. Our model is close to the one used to model protein-protein interaction networks in [43]; the difference is that, in our version, vertices that are within threshold distance of each other have a probability p of being connected. This model is sometimes referred to as the percolated random geometric graph.

The GEO model has two parameters: a threshold $r \in (0, 1)$ and a link probability $p \in (0, 1)$. A prescribed number of vertices are embedded uniformly at random into a metric space S. If the distance between vertices in this space is less than the threshold r, then an edge is added with probability p, independently for each pair of vertices. We consider a two dimensional (GEO2D) and a three dimensional (GEO3D) version of this model where the metric space S is $[0, 1]^2$ and $[0, 1]^3$ respectively, equipped with the torus metric. In addition, we randomly select a small number d of pairs of vertices uniformly at random, and add an edge between them. In our experiments, we take $d = \lfloor \log |V| \rfloor$.

Our model differs from the traditional random geometric graph by the addition of a small number of random edges. The reason is the following. It is well-known that social networks have the "small world" property, so the average shortest path length between vertices is small. However, to obtain the desired density we need to take the threshold r fairly small. This implies that each edge can bridge at most a distance r in the metric space. Thus, the path length between vertices that are at opposite ends of the metric space will be large, and as a result the average path length in random geometric graphs will be too large to make them a suitable model for social networks. The random edges remedy this problem by providing "shortcuts" between vertices that are far away in the metric space. On the other hand, the number of edges added is so small that the other features, such as graphlet counts and degree distribution, are not significantly affected.

Spatial Preferential Attachment Model (SPA). The Spatial Preferential Attachment model introduced in [3] is a spatial model which incorporates the preferential attachment principle. The model has three parameters $A_1 \in (0,1)$, $A_2 \geq 0$, and $p \in (0,1]$. We form a sequence of directed graphs $\{G_t\}$, $t \in \mathbb{N} \cup \{0\}$ with G_0 as the empty graph. We define a region of influence around a vertex v at time $t \geq 1$, written $R(v,t)$, with area

$$|R(v,t)| = \frac{A_1 \deg^-(v,t) + A_2}{t}$$

or $R(v,t) = S$ if the above is greater than 1. In the above, $deg^-(v,t)$ is the in-degree of v at time t. At each time step $t \geq 1$, a point in S is uniformly at randomly chosen to be the new vertex v_t. For each vertex $u \in V(G_{t-1})$ such that $v \in R(u, t-1)$, we independently add an edge from v_{t-1} to u with probability p. In this model, the influence regions are proportional to the in-degree of the vertex but decrease over time. Again, after model generation, we ignore the direction of the edges.

The vertices will be placed in the same metric spaces as the two GEO models above giving us a two dimensional (SPA2D) and three dimensional (SPA3D) version of the SPA model.

2.2. **Features.** We represent our graphs by 17 features in a vector representation. These features include information about the global properties of the graphs, specifically the degree distribution, the assortativity coefficient and the average path length between vertices. In addition, we capture the local structure through the raw graphlets counts for the connected subgraphs of size 3 and size 4. Below is a description of each of the features.

Degree Distribution Percentiles. The degree distribution is a favourite property studied for most "real world" networks. A distribution with a power law tail is a distinguishing property of many such networks, including the friendship network of Facebook (see [48]). The most logical feature to use here would be the coefficient of the power law degree distribution. Unfortunately, not all the models generate graphs with a power law degree distribution (*e.g.* random geometric models), and it is often difficult to establish whether the data exhibits a real power law, or to determine its coefficient. Instead, to measure the spread of the degree distribution, we consider the percentiles of the distribution formed by breaking it evenly into 8 different pieces. This give us 7 features, called $deg_1, deg_2, deg_3, deg_4, deg_5, deg_6$, and deg_7.

Assortativity Coefficient. The assortativity coefficient $r \in [-1,1]$ is a measurement of how well vertices of similar degree link to one another in the network. An assortativity coefficient close to -1 indicates that vertices tend to link to vertices of different degrees and a value close to 1 indicates that vertices tend to link to vertices of similar degrees. It is shown in [38] that online social networks have positive assortativity coefficients while the World Wide Web and biological networks have negative assortativity coefficients. We compute the assortativity coefficient in graph-tool using the following equation from [40],

$$r = \frac{\sum_i e_{ii} - \sum_i a_i b_i}{1 - \sum_i a_i b_i},$$

where e_{ij} is the fraction of edges from a vertex of degree i to a vertex of degree j and $a_i = \sum_j e_{ij}$ and $b_j = \sum_i e_{ij}$.

Average Path Length. The small world property, implying a small average hop distance between vertices, is another distinguishing aspect of social networks. It is shown in [38] that online social networks have small average path length. Here we will compute the average path length between vertices by selecting 100 random pairs of vertices and calculate the length of the shortest path between them using a breadth-first-search which is implemented in graph-tool.

Graphlets.

To characterize local structure, we include as features all the counts of connected subgraphs of size 3 (two nonisomorphic graphs) and 4 (six nonisomorphic graphs), as shown in Figure 1 below. Unfortunately, no algorithm is known which computes the full counts for all of these subgraphs efficiently for the size of graphs we are considering (some new algorithms to compute triangles are being developed). As a compromise, we use the sampling algorithm of Wernicke [49] to sample the number of these graphlets. The advantage of Wernicke's algorithm is that it can be used to give an unbiased sample of a specified portion of the subgraphs.

Wernicke's algorithm is based on a systemetic exploration of k-neighbourhood of the neighbourhood using DFS (cut off when level k is reached). The sampling works by probabilistically skipping steps of this exploration. To achieve unbiased sampling which obtains a prescribe proportion of all small subgraphs, the probability that a step is skipped is adjusted to the depth in the DFS where the step occurs. Additional details can be found in [49].

For our experiments, we sample 1% of the size 3 graphlets and 0.01% of the size 4 graphlets. Since the sampling rate for size 4 graphlets is very low (0.01%), we tested the assertion of Wernicke that the algorithm indeed leads to an unbiased sample of the graphlets. To this end, we computed the exact counts for the size 3 and 4 graphlets for a number of randomly chosen graphs, and compared them to the estimates obtained by sampling. In Table 2.2 below, we show the performance of the algorithm for one of the randomly generated graphs. You can see that there is good agreement between the real counts and the estimates. In light of this almost perfect agreement between sampling and exhaustive counts, we deemed the achieved sampling rate more than sufficient for our purposes.

The computation of the graphlets is by far the most time consuming of all the features. For the size and density of graphs we are considering it was not feasible to include subgraphs of size greater than 4. In [36], the authors consider subgraphs up to size 7 but this is only possible because the graphs are much smaller and sparser then those considered here. Inclusion of graphlets of larger size will only be possible for graphs of the size and density we consider here if new methods are developed to compute or estimate graphlets counts which show a dramatic increase in efficiency. However, our results show that the graphlets of size 3 and 4 are highly efficient in separating the models. Based on our results,

%	g1	g2	g3	g4	g5	g6	g7	g8
100	2323538	320097	18389736	65090655	22256380	3115254	4317267	434608
10	232335	32075	1837970	6508583	2227640	310958	431961	43176
1	23142	3243	184115	650031	222899	30905	43062	4378
0.1	2368	343	18341	65156	22381	3143	4281	453
0.01	224	33	1804	6524	2163	315	422	49

TABLE 1. Performance of Wernicke's Algorithm on a graph with 3000 vertices and 70270 edges.

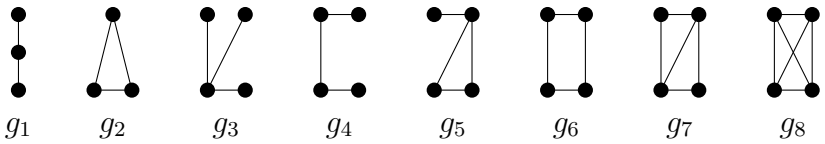

FIGURE 1. The graphlet features

we do not expect that inclusion of higher order graphlets will lead to a significant improvement the model selection method.

2.3. **Classification.** To classify our data we use the multi-class alternating decision tree (ADT) algorithm LADTree of Holmes *et al.* [19]. ADTs are a class of boosted decision trees which were introduced by Freunde and Mason in [14]. Boosting [15] is a well established classification technique which combines so called "weak classifiers" to form a single powerful classifier. In successive steps called *boosting steps*, weighted combinations of the weak classifiers are applied to the training data, and the weights are adjusted in each step to improve the classification.

The first ADTs were built using the AdaBoost boosting algorithm [15]. The ADT used here, LADTree, is built on the lesser known LogitBoost boosting algorithm of Friedman, Hastie and Tibshirani [17]. Friedman *et al.* show in their work that both boosting algorithms are fitting an additive logistic regression model. They argue that LogitBoost is the more appropriate algorithm, because it fits the regression model using the more typical maximum likelihood minimization criteria, whereas AdaBoost uses an exponential minimization criteria.

In Figure 2, we show a partial LADTree which was constructed during our experiment. An ADT has two types of nodes, *decision nodes* (rectangles in Figure 2) and *prediction nodes* (ellipses in Figure 2). Decision nodes contain a boolean predicate which corresponds to a threshold on one of the features in the feature vectors for the training data. The prediction nodes contain real-valued scores, one for each of the classes in the training set. In our case, we have six different classes or models so each prediction node contains six scores.

The LADTree begins with a prediction node which has a score of zero for each of the models. In each boosting iteration, a decision node is added to the tree along with two prediction nodes as its children in the tree. The new decision node can be added as a child to any existing prediction node in the tree. The placement of the decision node and its Boolean predicate is the one that gives the best separation of the training data. The exact criteria for this is provided by the LogitBoost algorithm [17].

Once the LADTree has been formed, new instances, typically called the test data in machine learning, can be classified by the tree. For us, the test data is the feature vector for the Facebook graph we wish to classify. The feature vector for the Facebook graph will determine its flow through the tree. The test instance travels through all possible paths it can reach in the tree resulting in a classification score which is the sum of all prediction nodes reached along the way. This results in six different scores, one for each of the six different models, F_j, $j = 1, 2, 3, 4, 5, 6$. A positive score is a good fit, a negative score is a bad fit. The model which obtains the highest score is deemed to be the model that best describes the test data. The absolute values of the scores provide the level of confidence in the prediction. Thus, a large positive F_j indicates that model j is a good model for the test instance and a large negative F_j indicates that model j is a bad model for the test instance. The scores F_j can be readily interpreted as class probabilities p_j by the equation $p_j = \frac{e^{F_j}}{\sum_{j=1}^{6} e^{F_j}}$ which results in inverting the additive logistic model which is fitted by the LADTree algorithm [19].

The advantage of using ADTs is that they require no specific assumption about the geometry of the input space for the features. Thus, we are free to incorporate any range of features such as degree distribution percentiles, average path length and subgraph counts without considering any potential dependence amongst them. The importance of each feature is based on how well it separates the 6 different models. We use the Weka software package for Java [50] to train all the LADTrees used in our experiments.

3. Results

We tested our approach on four different social network graphs taking from Mason Porter's Facebook100 data set [46]. Each graph in the data set corresponds to users at different universities. For our test data we take: Princeton University which has 6596 vertices and 293329 edges, American University which has 6386 vertices and 217661 edges, MIT which has 6440 vertices and 251252 edges and Brown University which has 8600 vertices and 384525 edges. In these graphs,

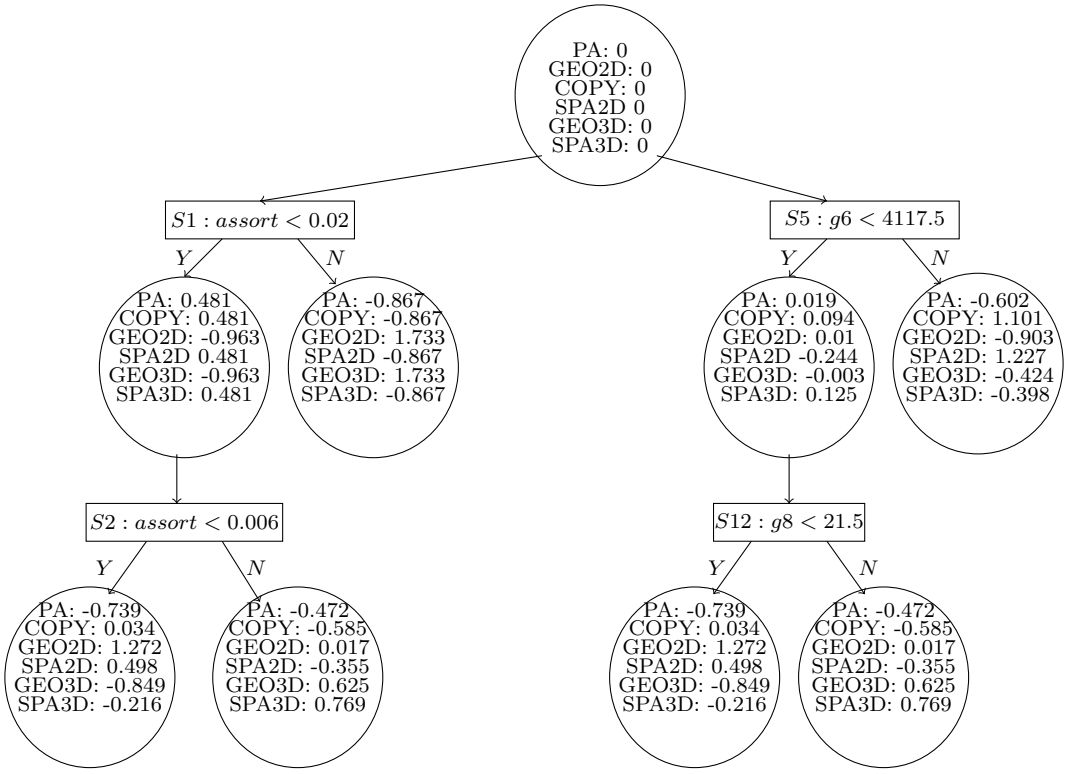

FIGURE 2. Partial LADTree using the full feature vector with 200 boosting iterations.

each vertex corresponds to a Facebook user, and two vertices are connected if they are Facebook friends.

For each of these graphs, the process is as follows. First, we generate a training set of 6000 graphs which are of the same size as the Facebook graph, and have edge density which differs by at most 5% from that of the Facebook graph. In order to test the effect of different features and different number of boosting iterations, we build 9 LADTree classifiers. The classifiers are built using 3 different types of feature vectors; the *full feature vector* which incorporates all 17 of the features described in Section 2.2, the *graph feature vector* using only the graphlet features and the *non-graph feature vector* which uses only the non-graphlet based features. For each of the feature vectors under consideration we build a classifier using 50, 100 and 200 boosting iterations, giving 9 classifiers in total for each experiment. To build the classifiers we use the well known machine learning software package Weka [50]. Finally, we use the classifiers to classify the Facebook graph. The model which produces the graphs which get the best scores is considered to be the best model for the data.

3.1. **Testing the Classifier.** Before performing our experiments on the actual Facebook data, it is important to test the classifier to find out how we should interpret the results. To this end, we generate an additional 100 graphs for each of the models, and apply the classifier to this known data set. Since we know exactly

which class these synthetic graphs belong to, this will establish a baseline for the maximum and minimum possible scores achievable by each model. We generate the initial 600 graphs with the same density as the Princeton network and classify them using the LADTree classifiers we have generated for the Princeton data.

First, we evaluate the performance of the classifier on these graphs, and test the effect of the number of iterations of the ADT tree. We use the full set of features. Consider the scores generated by the classifier for the unchanged synthetic graphs, shown in Tables 2, 3, 4, and 5. The rows in these tables correspond to the 100 additional graphs generated by each model and the column entry corresponds to the average score with standard deviation that each model scored for that row. As expected, the graphs are overwhelmingly assigned to the class corresponding to the model that generated them. The scores range roughly between -10 and 10 for 50 boosting iterations, -15 and 15 for 100 boosting iterations and -25 and 25 for 200 boosting iterations for both the full and graph features. The performance of the classifier is consistent over a different number of boosting iterations. Our other experiments confirm that the number of iterations does not make a significant difference. Therefore, from here on we present only the result for 100 boosting iterations.

Models	PA	COPY	GEO2D	SPA2D	GEO3D	SPA3D
PA	8.96 ± 1.18	-3.91 ± 2.39	-4.16 ± 1.18	0.17 ± 1.69	-2.38 ± 1.25	1.32 ± 0.82
COPY	-2.3 ± 0.34	7.02 ± 0.24	-2.19 ± 0.27	-0.19 ± 0.23	-3.2 ± 0.28	0.85 ± 0.27
GEO2D	-6.78 ± 1.59	-7.82 ± 3.55	9.13 ± 2.89	2.65 ±2.42	3.57 ± 1.47	-0.76 ± 1.55
SPA2D	-5.51 ± 2.5	-11 ± 3.86	2.89 ± 2.27	10.16 ± 3.05	-2.36 ± 2.04	5.81 ± 1.76
GEO3D	-6.14 ± 1.31	-8.42 ± 3.18	3.58 ± 1.61	-0.73 ± 1.05	9.04 ± 2.94	2.67 ± 2.32
SPA3D	-4.09 ± 2.48	-9.97 ± 4.54	0.03 ± 2.2	5.22 ± 2.06	-0.26 ± 2.79	9.07 ± 2.84

TABLE 2. Full Feature 50 Boosting iterations. Average over 100 test graphs, with standard deviation

Models	PA	COPY	GEO2D	SPA2D	GEO3D	SPA3D
PA	11.92 ± 0.89	-4.61 ± 1.25	-4.61 ± 1.93	2.39 ± 1.65	-5.11 ± 1.46	0.03 ± 1.03
COPY	-5.5 ± 1.67	11.73 ± 0.80	-0.34 ± 1.11	1.37 ± 1.02	-8.25 ± 1.93	0.99 ± 1.4
GEO2D	-10.83 ± 1.96	-10.64 ± 4.54	12.59 ± 3.19	4.07 ± 2.42	6.02 ± 1.91	-1.2 ± 1.84
SPA2D	-8.08 ± 3.57	-13.61 ± 4.57	3.04 ± 2.88	13.79 ± 3.72	-2.38 ± 3.45	7.25 ± 1.99
GEO3D	-10.72 ± 2.21	-12.55 ± 5.11	5.81 ± 2.30	1.56 ± 2.07	13.25 ± 3.79	2.66 ± 2.4
SPA3D	-6.62 ± 3.79	-13.04 ± 5.68	-0.09 ± 2.97	6.94 ± 2.17	-0.45 ± 4.13	13.26 ± 4.05

TABLE 3. Full Feature 100 Boosting iterations. Average over 100 test graphs, with standard deviation

To determine the importance of the graphlet features, we compare the performance of the classifiers built using the full feature vector with the ones built using only the graph feature vector. Table 5 shows the performance on the synthetically generated test graphs when only the graph feature vector is used. Again, almost all

Models	PA	COPY	GEO2D	SPA2D	GEO3D	SPA3D
PA	16.69 ± 1.5	-5.9 ± 1.55	-6.62 ± 2.75	2.42 ± 2.21	-8.90 ± 2.15	2.31 ± 1.66
COPY	-8.2 ± 4.9	18.31 ± 0.8	-6.31 ± 1.61	1.95 ± 2.03	-9 ± 2.46	3.24 ± 2.05
GEO2D	-16.79 ± 4.09	-18.23 ± 7.03	19.27 ± 3.01	5.48 ± 3.05	9.41 ± 3.44	0.86 ± 3.18
SPA2D	-10.75 ± 5.6	-20.41 ± 6.61	5.46 ± 4.24	19.53 ± 4.40	-3.71 ± 5.34	9.88 ± 2.56
GEO3D	-17.57 ± 4.34	-21.78 ± 8.46	8.92 ± 3.44	2.32 ± 2.96	20.5 ± 4.98	7.6 ± 4.12
SPA3D	-8.73 ± 5.76	-20.74 ± 7.99	0.82 ± 4.55	9.59 ± 2.7	0.07 ± 5.94	18.99 ± 4.06

TABLE 4. Full Feature 200 Boosting iterations. Average over 100 test graphs, with standard deviation

graphs are classified correctly. In comparing Tables 3 and 5, we can observe that the test graphs receive similar scores regardless of rather the full feature vector or the graph feature vector is used. In some cases, using the graph feature vector only produced higher scores for the geometric based models but not significantly higher. Thus we can conclude that graphlets alone are sufficient to recognize the graph structure of the models under consideration.

Models	PA	COPY	GEO2D	SPA2D	GEO3D	SPA3D
PA	10.36 ± 0.51	0.45 ± 0.44	-5.26 ± 0.72	-0.6 ± 1.02	-6.19 ± 0.78	1.24 ± 0.98
COPY	0.07 ± 0.95	9.71 ± 0.85	-5.27 ± 0.36	0.35 ± 1.16	-5.95 ± 0.64	1.1 ± 0.36
GEO2D	-12.18 ± 3.11	-14.9 ± 4.77	13.86 ± 3.98	5.97 ± 3.11	6.53 ± 2.49	0.72 ± 1.85
SPA2D	-11.71 ± 2.89	-13.74 ± 4.32	2.35 ± 2.55	15.41 ± 3.44	-0.78 ± 3.31	8.47 ± 2.27
GEO3D	-13.21 ± 3.28	-15.36 ± 4.39	7.45 ± 1.86	3.27 ± 2.02	13.39 ± 2.84	4.46 ± 2.67
SPA3D	-11.41 ± 3.34	-14.22 ± 4.61	-0.03 ± 2.32	9.06 ± 2.17	1.62 ± 3.56	14.99 ± 3.03

TABLE 5. Graph Feature 100 Boosting iterations. Average over 100 test graphs, with standard deviation

Finally, we test the robustness of the classifier with respect to perturbations of the graph structure. To do this, we take the 600 synthetic graphs and change a percentage of the edges. This is done by randomly choosing a set of edges from the graph, removing them, and replacing each edge with a new edge whose endpoints are chosen uniformly at random. The goal is to see how fast the classification accuracy deteriorates as the number of edge changes increases. Overall, we have 6 test data sets of 600 graphs each, with 0%, 5%, 10%, 15% , 20% and 25% of the edges randomly changed.

Tables 6 and 7 give the classification accuracy for each of the six test data sets using the full feature vector and graph feature vector respectively. As seen before, the classification accuracy on the original unchanged test data is very high for both the full and graph feature vectors, and the classification accuracy is slightly but not significantly higher when only the graph feature is used. When 5% of the edges are changed the classification accuracy for the full feature drops to just below 75% while for the graph feature vector the accuracy is just below 80%. In this case, the graph feature vector alone performs significantly better than the full feature vector. For all other percentages of edge changes, the difference between

the two is not significant. The conclusion of this experiment is that the graph features alone provide just as much information as the full feature set. In fact, as is the case when 5% of the edges were changed, including additional non-graph information can decrease the accuracy of the classifier. When 10% of the edges are changed, both feature vectors give classification accuracies around 65% which is still a fair performance. When 15% of the edges are changed, the accuracy for both feature vectors drops to around 55%. At 20% and 25%, the accuracy dips below 50%. The accuracy at this level is not good but there clearly still is information present in the link structure, since classifying the graphs completely at random would give the correct classification less than 17% of the time.

Another interesting observation is that the overall classification accuracy does not necessarily increase with the number of boosting iterations. It is the case that increasing the number of boosting iterations improves the classification accuracy on the unchanged data but this is not necessarily the case for the changed data. For most of the test data sets the difference is not significant but when 25% of the edges are changed, the classification accuracy is about 3% better when only 50 boosting iterations are performed as compared to 200 boosting iterations. We suspect that increasing the number of boosting iterations leads to overfitting of the perturbed data.

Edge Changes %	Boosting Iterations 50	100	200
0	94.67	95.67	97.17
5	73.83	71.5	74.33
10	64	63.33	65.17
15	57.33	56.17	56.33
20	51.17	48.67	48.83
25	44.17	43	41.17

TABLE 6. Full Feature Classification Accuracy

Edge Changes %	Boosting Iterations 50	100	200
0	94.83	96.67	97.83
5	78.67	79.83	79.67
10	64	63.5	63.67
15	56.17	55.67	54.8
20	49.33	48	48.17
25	44	40.5	40.67

TABLE 7. Graph Feature Classification Accuracy

To find out how exactly the graphs are misclassified, we present the complete classification results for the classifier trained with the graph feature vector, in Table 8. Here we can see that the 3D (GEO3D and SPA3D) models are very robust against the changing of edges while their 2D (GEO2D and SPA2D) counterparts are not. Precisely, a large part of the misclassification of perturbed graphs is due to the classification of GEO2D and SPA2D graphs as GEO3D and SPA3D, respectively. Even with the lowest level of perturbation, 5%, roughly half of the 2D models are classified as their 3D counterparts. When 25% of the edges have been changed, only around 5% of the 2D models are classified correctly, with most of the graphs being classified as the 3D counterpart. Meanwhile, the 3D models maintain a good classification accuracy even when 25% of the edges are changed.

Another interesting observation is that the COPY model is also somewhat robust against the changing of edges. Even with 5% of the edges switched, all the

COPY graphs are classified correctly. The accuracy dips to around 95% occurs when 10% of the edges are switched. Even when 25% of the graph is changed, the classification accuracy stays within 50%–70%. The PA model on the other hand is not robust against the changing of edges. The classification quickly decreases as edge changes start to accumulate. Interestingly, PA graphs are only confused with COPY model, not with any geometric model. Especially, PA graphs are never confused with the SPA models, even though both models incorporate the preferential attachment principle.

Change	PA	COPY	GEO2D	SPA2D	GEO3D	SPA3D	Models
0%	100	0	0	0	0	0	PA
	0	100	0	0	0	0	COPY
	0	0	92	2	6	0	GEO2D
	0	1	0	97	0	2	SPA2D
	0	0	5	0	95	0	GEO3D
	0	0	0	4	0	96	SPA3D
5%	88	2	0	0	0	10	PA
	0	100	0	0	0	0	COPY
	0	0	49	2	49	0	GEO2D
	0	0	0	47	0	53	SPA2D
	0	0	4	0	96	0	GEO3D
	0	0	0	1	0	99	SPA3D
10%	78	14	0	0	0	8	PA
	0	94	0	6	0	0	COPY
	0	0	11	1	88	0	GEO2D
	0	0	0	3	1	96	SPA2D
	0	0	3	0	97	0	GEO3D
	0	0	0	1	1	98	SPA3D
15%	51	45	0	0	0	4	PA
	0	82	0	18	0	0	COPY
	0	0	7	2	91	0	GEO2D
	0	0	0	2	6	92	SPA2D
	0	0	2	0	98	0	GEO3D
	0	0	0	1	5	94	SPA3D
20%	24	76	0	0	0	0	PA
	0	69	0	31	0	0	COPY
	0	0	8	2	90	0	GEO2D
	0	0	0	2	12	86	SPA2D
	0	0	4	0	96	0	GEO3D
	0	0	0	1	10	89	SPA3D
25%	2	98	0	0	0	0	PA
	0	53	0	46	1	0	COPY
	0	0	6	2	92	0	GEO2D
	0	0	1	4	18	77	SPA2D
	0	0	4	0	96	0	GEO3D
	0	0	0	1	17	82	SPA3D

TABLE 8. Classification of perturbed graphs. Graph feature vector with 100 boosting iterations

One purpose of testing the robustness of the classifier is to attempt to simulate the behaviour of the classifier on noisy data. One conclusion we have, is that even if a little bit of noise is introduced into the data, the 2D models are more likely to get classified as a 3D model. The conclusion is that if unknown data is classified as a 3D model, it is possible that the correct model should be the 2D model. We also can conclude that using the graph feature vector is at least as reliable as using the full feature vector.

3.2. **Classification of the Facebook networks.** After verifying the quality of the classifier, we now apply the classifiers to the data sets for which they were designed. Recall that, for each data set, we have built a different classifier, based on test data from synthetic graphs generated according to the different models, with parameters tuned so that the resulting graphs have approximately the same edge density as the graph from the data set. In Table 9, we present the classification scores for each of the four data sets, for classifiers using the full feature vector, graph feature vector, and the non-graph feature vector. The highest score is given in boldface; when the two highest scores are close, both are highlighted.

Classifier	PA	COPY	GEO2D	SPA2D	GEO3D	SPA3D
full Princeton	-0.303	-14.551	4.599	**11.287**	-5.451	4.42
graph Princeton	**6.699**	-2.227	-3.914	3.085	-3.676	0.033
non-graph Princeton	-0.858	-3.622	-7.447	**8.022**	-5.029	**8.941**
full American	-0.414	-12.164	-0.183	8.307	-5.578	**10.025**
graph American	0.779	-10.639	0.381	5.834	-7.693	**11.332**
non-graph American	-4.612	-2.442	-3.627	**6.517**	-3.348	**7.512**
full MIT	2.956	-12.512	2.715	**13.528**	-8.561	1.873
graphs MIT	4.097	-9.49	3.061	**5.304**	-2.91	-0.063
non-graph Brown	-0.197	-3.58	-2.61	**4.549**	-1.606	3.44
full Brown	4.998	-15.163	-0.305	1.733	-6.161	**14.897**
graphs Brown	**6.283**	-0.085	-3.774	1.827	-3.771	-0.479
non-graph MIT	1.956	-7.305	-2.458	2.518	-2.901	**8.192**

TABLE 9. Scores for each data set, for each of the classifiers with 100 boosting iterations

The first clear conclusion of the outcome is that all significant high scores are for models that incorporate the preferential attachment principle: PA, SPA2D and SPA3D. In most cases, both the SPA2D and SPA3D give fairly high positive scores. From results presented in the previous subsection on the perturbed graphs, we know that misclassification between SPA2D and SPA3D is common. Thus, only the general conclusion that the SPA model fits the data well, is justified. Other techniques will be needed to determine the dimension. The PA model gives the highest score for two of the data sets, but *only with the classifier that only uses the graph features.* For our synthetic graphs, we showed graphlet features are at least as efficient as the full feature set in distinguishing the classes. For real data, the situation clearly differs, and we see a fairly large discrepancy between graph feature and full feature results. Since the full feature set is based on the widest range of features, it makes sense to base our final conclusion on the classifier built using all available features. In this light, the SPA model clearly gives the best fit for the Facebook data.

Classification algorithms are built under the assumption that the test data actually belongs to one of the classes the classifier is trained to distinguish. This assumption is often not entirely justified in realistic applications, as is the case

here. However, it is common practice to evaluate unknown data using a classification algorithm. With this in mind, we performed a more detailed analysis of the classification, to help interpret our results from the Facebook data. In our analysis, we do not consider only the final score of the classifier on the data set, but we also consider how each feature contributes to the score for each model. Specifically, we extract information about the features which appear in the first layer of nodes (of depth 1) in the ADT. Since the first layer of the ADT is most important in separating the data, the features occurring in the first layer are the most influential in the classification. Furthermore, we consider how often each feature is visited when the classifier is applied to the Facebook data. Combined with our knowledge about the different models, and their typical behaviour with respect to the various features, our analysis gives a more detailed picture of the classification results.

In this section we give a general discussion of our analysis; a precise discussion of the performance of the classifier on each of the data sets can be found in our technical report [21]. Our general analysis is based on comparing how well the models were able to match the most important features in the classification scores to those present in the Facebook graphs. For this analysis, we generated box-plots to visualize how well each feature generated in the models compared to the feature present in the Facebook graphs. (See Figure 3 for an example; all box plots can be found in [21].) We also analyzed the ADTs to see which feature was most represented among the decision nodes.

Our first observation is that, in every classifier built using the full feature vector, the first node in the ADT corresponds to the assortativity coefficient. Thus, the assortativity coefficient is the most significant feature in separating the classes. From the box plots of the feature values, we have observed that the assortativity coefficient of the GEO models is significantly higher than all the other models, as shown in Figure 3. This can also be explained theoretically; it is easy to see that the vertices in a GEO model have degrees which are binomially distributed, which implies that many vertices will have similar degrees, which leads to a higher assortativity coefficient. Note that the assortativity coefficient is not included in the graph feature vector, while the results for the synthetic test graphs show that the graph feature vector is equally successful in separating the models. Thus, the information conveyed by the assortativity coefficient should be implicitly contained in the graphlet counts.

The most important graphlet feature was g_6, which corresponds to the 4-cycle. The 4-cycle feature tends to be the most important feature overall. That is, it appears frequently in the first layer of nodes in the ADT and it is usually the feature which is most visited by the Facebook data when it is put through the classifier. In some cases, the outcome of the classification can be deduced by only considering the feature g_6. In most cases, the SPA models were the models which were able to generate 4 cycle counts which were the closest to the 4-cycle counts in the Facebook graphs. This can be seen in the box plot for the Princeton network in Figure 3. The box-plots for the other Facebook graphs can be seen in [21]. This is a major factor in why the SPA models performed so well in our experiments.

An important difference between the models is that the PA and COPY models tend not to generate highly connected subgraphs whereas the GEO models do tend to generate highly connected subgraphs. Conversely, the PA and COPY models generate many sparse subgraphs whereas the GEO models do not. By highly connected subgraphs we mean those that contains a triangle, namely: g_2, g_5, g_7. Sparse subgraphs are those without a triangle: g_1, g_3, g_4. In particular, for some experiments, the ability of the PA model to generate a high number of 3 and 4-paths which fit extraordinary well with the 3 and 4-paths generated in the Facebook data resulted in the PA model having the best performance. This phenomena was especially significant for the Princeton and Brown experiments when only the graph feature vector was used. Overall, the SPA models are able to generate a mixture of dense subgraphs and sparse subgraphs, which explains the good performance overall of the SPA model.

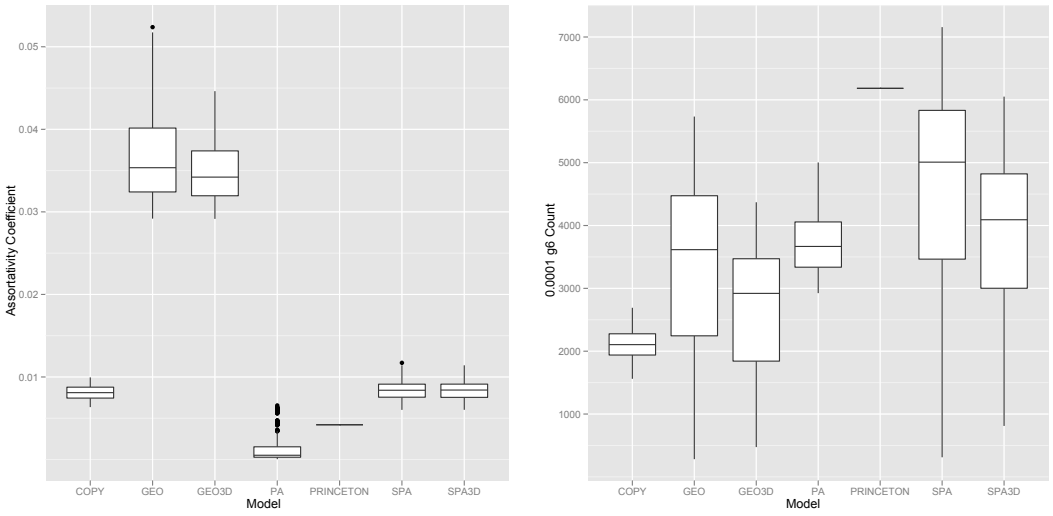

FIGURE 3. Box-plots representing the spread of the assortativity coefficient (left) and the $g6$ graph feature (right) for the Princeton network.

A final interesting observation comes from comparing the experiments for the Princeton and Brown networks. Though the Princeton network has 6596 vertices and the Brown network has 8600 vertices, they both have almost the same edge density. The conclusions of the two experiments are similar and for the graph feature vector in particular they are almost identical. Moreover, the ADTs produced for each of the networks are very similar. They have the exact same first layer of nodes for both the full and graph feature vector. This suggests that training sets with graphs of the same density generate similar ADTs, and that the same classifier could be used for observed networks of comparable density. This also implies that, if the appropriate normalization factor could be found for comparing subgraph counts for the graphs of different size but similar density then the building of the classifier would only have to be done once. The same classifier could then be applied to suitably normalized feature vectors of the data. Since the generation of the samples of each model, and the computation of the graphlet

counts for these samples takes a lot of computation time, this would improve our method significantly.

4. Conclusions and Further Work

The main goal of this work was to determine which of our 6 models is the most appropriate for a social network such as Facebook. The results of our experiments show clearly that the models incorporating a preferential attachment mechanism give the better fit. However, based on our work, it is difficult to determine whether the PA or the SPA model is better. This is because we have not performed enough experiments to develop a statistically significant sample size. However, the fact that in all four experiments, for almost every classifier generated, the PA and SPA models generally received positive scores indicate that the models do fit the test data quite well. On the other hand, the COPY model generally gave high negative scores for almost all the classifiers generated, indicating that the model is a poor fit for the Facebook graphs considered.

Our work has shown conclusively that our classification procedure works well at separating graphs produced by each of our models even when the models generate graphs with similar degree distributions and average path lengths. This gives evidence to our claim that local structure is important in developing models for real world networks. Furthermore, we saw that the classification accuracy using all of the features and using only the graphlet features were not significantly different. We can conclude from this that considering graphlets are sufficient to separate the models.

Our results show that graphlets corresponding to paths, cycles and highly connected subgraphs are the most influential in distinguishing between different models. This is not a surprising conclusion because a high count of paths and a low count of complete subgraphs are characteristic of sparse models such as PA and COPY while a low count of paths and a high count of complete graphs is characteristic of denser models such as GEO2D and GEO3D. The ability of the SPA models to generate a good mixture of all the subgraphs, in particular the 4-cycle, resulted in the SPA models performing well across all experiments.

Currently, it is necessary to generate a new training set and classifier for each test network of a given size and density. This is because graphlet counts are highly dependent on the size and density of the graph. We are interested in determining a method to normalize the graphlet counts so that graphlet counts for graphs of varying sizes and densities can be compared. Such a normalization would make it possible to build a single classifier which can test networks for a range of sizes and densities. In this work, the amount of time to perform one experiment took about two weeks so the existence of a universal classifier through the normalization of graphlet counts would make testing various real world networks more tractable.

References

[1] Y. Ahn, S. Han, H. Kwak, S. Moon, H. Jeong, *Analysis of topological characteristics of huge on-line social networking services*, In: Proceedings of the 16th International Conference on World Wide Web (WWW), 2007.

[2] M. Adler and M. Mitzenmacher, *Toward Compressing Web Graphs*, In: Proceedings of the 2001 Data Compression Conference, 2002, pp. 203–212.
[3] W. Aiello, A.Bonato, C.Cooper, J. Janssen, P. Prałat, *A Spatial Web Graph Model with Local Influence Regions*, Proceedings of the 5th Workshop on Algorithms and Models for the Web-Graph, 2007.
[4] W. Aiello, F. Chung, L. Lu, *Random Evolution of Massive Graphs*, in: Handbook of Massive Data Sets, J. Abello *et al.*, Eds., Kluwer, 2002, pp. 97–122.
[5] A. Barabási, R. Albert, *Emergence of Scaling in Random Networks*, Science **286**, 1999, 509-512.
[6] G. Bebek, P. Berenbrink, C. Cooper,T. Friedetzkyd, J. Nadeaue, S.C. Sahinalp, *The degree distribution of the generalized duplication model* Theoretical Computer Science 369, 2006, pp. 234–249.
[7] A. Bonato, *A Course on the Web Graph*, American Mathematical Society Graduate Studies Series in Mathematics, Providence, Rhode Island, 2008.
[8] A. Bonato and J. Janssen, *Infinite limits and adjacency properties of a generalized copying model*, A. Bonato, Internet Mathematics 4 (2009) pp. 199–223.
[9] A. Bonato, J. Janssen, P. Prałat, *The Geometric Protean Model for On-Line Social Networks*, in Proc. Workshop on Algorithms and Models of the Web Graph (WAW) 2010, Springer LNCS 6516, pp. 110–120.
[10] A. Bonato, N. Hadi, P. Horn, P. Prałat, C. Wang, Models of on-line social networks, accepted to *Internet Mathematics*, 2010.
[11] X. Cheng, C. Dale, J. Liu, *Statistics and Social Network of YouTube Videos*, in: Proc. 16th International Workshop on Quality of Service (IWQoS) 2008. IWQoS 2008, pp. 229–238.
[12] F.R.K. Chung, L. Lu, *Complex Graphs and Networks*, American Mathematical Society, U.S.A., 2004.
[13] F. Chung, L. Lu, T.G. Dewey, D.J. Galas, *Duplication models for biological networks*, J. Comput. Biol. 10, 2003, pp. 677-687.
[14] Y. Freunde, L. Mason, *The Alternating Decision Tree Learning Algorithm*, In Proc. 6th Int. Conf. on Machine Learning, 1999, 124-133.
[15] Y. Freunde, R. Schapire, *A Decision- Theoretic Generalization of Online Learning and an Application to Boosting*, 1995.
[16] Y. Freunde, R. Schapire, *Experiments with a new boosting algorithm*, Machine Learning: Proceedings of the 13th International Conference, 148-156.
[17] J. Friedman, T. Hastie, R. Tibshirani, *Additive logistic regression: a statistical view of boosting*, Annals of Statistics, 2000, 337–407.
[18] D. Higham, M. Rasajski, N. Przulj, *Fitting a geometric graph to a proteinprotein interaction network*, Bioinformatics, 2008, Vol. 24 (8), 1093–1099.
[19] G. Holmes, B. Pfahringer, R. Kirkby, E. Frank, M. Hall, *Multiclass Alternating Decision Trees*, ECML, 2002, 161–172.
[20] I. Bordino, D. Donato, A. Gionis, S. Leonardi, *Mining Large Networks with Subgraphs* ICDM 08, 737 742, 2008.
[21] M. Hurshman, J. Janssen, N. Kalyaniwalla, *Model Selection Using Graphlets*, Technical Report CS-2011-07, Faculty of Computer Science, Dalhousie University, 2011.
[22] J.Janssen, *Spatial models for virtual networks*, Computability in Europe, LNCS, 2010.
[23] J. Janssen, P. Prałat, R. Wilson, *Geometric Properties of the Spatial Preferred Attachment Model*, preprint, 2011, arXiv:1111.0508v1 [cs.SI].
[24] M. Kim and J. Leskovec, *Multiplicative Attribute Graph Model of Real-World Networks* in Proc. 8th Workshop on Algorithms and Models for the Web-Graph (WAW) 2010, R. Kumar, D. Sivakumar, Springer LNCS 6516, 62–73.
[25] J. Kleinberg, S.R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkin, *The Web as a Graph: Measurements, Models and Methods*, Proceedings of the International Conference on Combinatorics and Computing, 1999.
[26] R. Kondor, N.Shervashidze, K. Borgwardt, *The Graphlet Spectrum*, ICML, 2009.

[27] P. Krapivsky, S. Redner, *Network Growth by Copying*, Physical Review E, Vol. 71, 2005.
[28] B. Krishnamurthy, P. Gill, M. Arlitt *A few chirps about twitter*, In Proceedings of the first workshop on Online social networks (2008), pp. 19–24, ACM.
[29] R.Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, E. Upfal, *Stochastic models for the web graph*, Proceedings of the 41th IEEE Symposium on Foundations of Computer Science, 2000.
[30] R. Kumar, J. Novak, A. Tomkins, Structure and evolution of on-line social networks, In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.
[31] S. Lattanzi, D. Sivakumar, Affiliation Networks, In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009.
[32] J. Leskovec, C. Faloutsos, *Sampling from Large Graphs*, ACM SIGFDD International Conference on Knowledge Discovery and Data Mining, 2006.
[33] J. Leskovec, J. Kleinberg, C. Faloutsos, *Graphs over time: Densification laws, shrinking diameters and possible explanations*, ACM SIGKDD, 2005.
[34] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, Realistic, mathematically tractable graph generation and evolution, using Kronecker multiplication, In: *Proceedings of European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2005.
[35] M. McGlohon, L. Akoglu, C. Faloutsos, *Statistical Properties of Social Networks*, in: *Social Network Data Analytics*, C. Aggarwal, Ed., Springer 2011, 17–42.
[36] M. Middendorf, E. Ziv, C. Wiggins, *Inferring Network Mechanism: The Drosophila Melanogaster Protein Interaction Network* PNAS **102**, 2005, 3192–3197.
[37] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii and U. Alon, *Network Motifs: Simple Building Blocks of Complex Networks*, Science, 2002, Vol. 298 no. 5594, 824–827.
[38] A. Mislove, M. Marcon, K.Gummadi, P. Druschel, B. Bhattacharjee, *Measurement and Analysis of Online Social Networks*, In Proc. 7th ACM SIGCOMM Conference on Internet Measurement, 2007, 29-42.
[39] A. Nazir, S. Raza, and C.-N. Chuah. *Unveiling Facebook: a measurement study of social network based applications.* In Proceedings of the 8th ACM SIGCOMM conference on Internet measurement (IMC '08). ACM, New York, NY, USA, 43–56.
[40] M. Newman, *Mixing Patterns in Networks*, Phys Rev. **67**, 2003.
[41] M. Penrose, *Random Geometric Graphs*, Oxford Studies in Prob., Vol. 5, Oxford University Press, 2003
[42] N. Przulj, D.G. Corneil, I. Jurisica, *Modeling interactome: scale-free or geometric?*, Bioinformatics 20 (18), 2004, pp. 3508-3515.
[43] N. Pržulj, *Biological Network Comparison using Graphlet Degree Distribution*, Bioinformatics 23, 2007, 177-183.
[44] A. Sala, L. Cao, C. Wilson, R. Zablit, H. Zheng and B. Zhai, *Measurement-calibrated Graph Models for Social Network Experiments*, In WWW'10, 861-870, 2010.
[45] http://projects.skewed.de/graph-tool/ Home web page for the graph-tool python module.
[46] www.insna.org, Web page for the International Network for Social Network Analysis. Accessed Feb 3rd, 2011.
[47] N. Shervashidze, SVN. Vishwanathan, T. Petri, K. Mehlhorn, K. Borgwardt, *Efficient Graphlet Kernels for Large Graph Comparison*, AISTATS 2009.
[48] A. Traud, P. Mucha, M. Porter, *Social Structure of Facebook Networks*, arXiv:1102.2166, 2011.
[49] S. Wernicke, *Efficient Detection of Network graphlets*, IEEE/ACM Transaction on Computational Biology and Bioinformatics Vol 3, Issue 4, 2006, 347-359.
[50] http://www.cs.waikato.ac.nz/mi/weka/, Home page for the open source machine learning software Weka.

E-mail address: janssen@mathstat.dal.ca

Department of Mathematics and Statistics, Dalhousie University, PO Box 15000, Halifax, NS, Canada, B3H 4R2

E-mail address: mhursh@mathstat.dal.ca

Faculty of Computer Science, Dalhousie University, Halifax, Canada
E-mail address: nauzerk@cs.dal.ca