**HNet: Graphical Hypergeometric Networks.**

Erdogan Taskesen1\*

1Datalab, Ministry of Infrastructure and Water Management, Delft, 2622 HA, the Netherlands

ET: erdogan.taskesen@rws.nl

\* Corresponding author

# **Abstract**

**Background:** Real-world data sets often contain measurements with both continues and categorical values for the same sample. Despite the availability of many libraries, data sets with mixed data types require intensive pre-processing steps, and it remains a challenge to describe the relationships of one variable on another. The data understanding part is crucial but without making any assumptions on the model form, the search space is super-exponential in the number of variables and therefore not a common practice.

**Result:** We propose graphical hypergeometric networks (HNet), a method where associations across variables are tested for significance by statistical inference. The aim is to determine a network with significant associations that can shed light on the complex relationships across variables. HNet processes raw unstructured data sets and outputs a network that consists of (partially) directed or undirected edges between the nodes (i.e., variables). To evaluate the accuracy of HNet, we used well known data sets, and generated data sets with known ground truth by Bayesian sampling. In addition, the performance of HNet for the same data sets is compared to Bayesian structure learning.

**Conclusions:** We demonstrate that HNet showed high accuracy and performance in the detection of node links. In the case of the Alarm data set we can demonstrate an average MCC score 0.33 + 0.0002 (*P*<1x10-6), whereas Bayesian structure learning showed an average MCC score of 0.52 + 0.006 (*P*<1x10-11), and randomly assigning edges resulted in a MCC score of 0.004 + 0.0003 (*P*=0.49). Although Bayesian structure learning showed slightly better results, HNet overcomes some of the limitations of existing methods as it processes raw unstructured data sets, it allows analysis of mixed data types, it easily scales up in number of variables, and allows detailed examination of the detected associations.

**Key words:** hypergeometric test, structure learning, networks, unstructured data.

**Availability**: HNet is available at git XXX.

# **Introduction**

In recent years, there has been much effort in the progress of network-learning. The importance is stressed by many applications that are complex interaction systems such as social networks, collaboration networks but also biological networks. By revealing the patterns within the network, we can better understand the organizational and structural functions of network systems. In global, the field of network learning can be dissected into two groups; the generative and discriminative models. The challenges for generative models is to learn a network representation for an existing network (e.g., social network). Commonly used techniques are graph/knowledge embeddings that transform node-links in a low-dimensional vector, which can then be used in applications with supervised or unsupervised models. Such approaches capture the complex associations between node-links for which some popular methods are Splitter1, Deepwalk2, node2vec3 and LINE4. On the other hand, the challenges for a discriminative model is to learn the network structure or its associations (node links) given the data set. In such case, unstructured data is used as an input into the model and the goal is to determine the representing network by means of (strong) relationships between the variables. Questions can be addressed such as; does variable X (in)directly influence Y, or might they have a common cause? Representations can be learned using Bayesian Network structure learning5,6 which aims to determine the directed acyclic graphs (DAG) given the data. Bayesian learning has been successfully applied in many fields such as insurance7, health8, and biological networks9. However, the search space of DAGs is super-exponential in the number of variables for which the typical scoring functions can result in a local suboptimum. This is especially the case for large data sets (e.g., with many node links to be determined) where an exhaustive search is intractable due to computational burden. It is known that Bayesian approach has NP-complete11 complexity which requires a quantum solution12 for very large data sets. Nevertheless, for small data sets, an exhaustive search for DAGs can be computed, whereas for medium data sets, the use of heuristics (e.g., hill-climbing10) in combination with Bayesian approaches can provide a good solutions. In addition to Bayesian learning, there are also rule-based machine learning techniques (association rules) to discover co-occurrence relationships between variables (item sets) in the large search space. The use of rule-based techniques, such as Apriori13, Eclat14 and FP-Growth, is successfully applied in many fields such as marketing (e.g., product placements, promotional pricing), retail (e.g., loyalty programs, sales promotions), security (e.g., intrusion detection15, malicious activities), web usage mining (e.g., advertisements). A drawback is however the risk of finding many spurious associations, and the limitation of only modelling categorical values (item lists).

With HNet, we developed a discriminative model with the aim to discover statistically significant associations across categorical as well as continues variables. We control for the risk of finding spurious associations by multiple test correction and the detected associations form the edges in a network. HNet does not force variables into static item sets but instead we created variable item sets that allows deep examination by our interactive network. To test HNet, we evaluated the detection of networks, i.e., sprinkler, and Alarm16 data set with known ground truth and demonstrate high accuracy of our solutions. In addition we analysed a well-known unstructured data sets, the Titanic17. Here we demonstrate the goodness of fit, the ease of use and how to deeper examine the discovered associations.

# **Material and Methods**

**HNet.** Detection of significant edge probabilities between pairs of vertices (node-links) given a data set using HNet is a multi-step process (Figure 1, A-F). The first step is pre-processing the data set by feature typing (Figure 1A). This step types every feature in either being categoric, numeric or is excluded. The typing can be user-defined or automatically determined on conditions. In the latter case, features are set to numerical if values are floating type or have more than a minimum number of unique elements (e.g., if the number of unique elements >20% of the total non-missing). Features are set to categoric if values are boolean, integer or string. The second step (Figure 1B) is encoding the categoric values into a one-hot dense array. The one-hot dense array is subsequently used to create combinatory features using *k* combinations over *n* features (without replacement, Figure 1C). The default *k* is set to 1, meaning that the input matrix (*Xcategoric*) is the same as combinatory matrix (*Xcombination*). When k>1, *n* boolean features are combined by multiplication for the *k* unique combinations (eq.1). The new combinatoric feature (Xc) is then added to the dense array. To avoid high computational costs, mutual exclusive features are excluded, and features are excluded in case *Xi* contains less then ymin positive samples (default ymin=10, eq.2).

(1)

(2)

The dense array (*Xcombination*) is used to assess significance with the categoric features (*Xcategoric*) (Figure 1D). Significance is tested by means of the hypergeometric distribution, where we test for over-representation of successes in sample *Xi*. The hypergeometric P-value between feature X*i* and feature Xc*j* (*Pc(i,j)*), is calculated as the probability of randomly drawing *x* or more successes from the population in *n* total draws with population size *N* containing *K* successes. For any *Xi* and *Xcj*, *Pc(i,j)* is computed as depicted in eq.3.

(3)

To assess significance across the numeric features (*Xnumeric*) in relation to the dense array (*Xcombination*) we utilized the Mann-Whitney-U test. Each numeric vector *Xni*, is split on categoric feature *Xci* versus ~*Xci*, and tested whether randomly selected value from *Xci* will be less than or greater than a randomly selected value from ~*Xci*.

All tested edge probabilities between pairs of vertices, either categoric-categoric or categoric-numeric, are stored in an adjacency matrix (*Padj*), and are corrected for multiple testing. The default Multiple Test Method (MTM) is set to Holm20 (Figure 1, equation 4) but optional are various other False Discovery Rate (FDR)18 or Familywise error rate (FWER)19 methods.

(4)

The last step in HNet (Figure 1F) is declaring significance for node-links. An edge is called significant with alpha being set at 0.05 by default. The edge-weight is computed as depicted in equation 5.

(5)

The final output of HNet is an adjacency matrix containing edge weights that indicate whether pairs of vertices are significantly associated or not in the graph. The adjacency matrix can then be examined as a network representation using d3graph.

**d3graph** is a dynamic graph representation to deeper examine the detected associations. Just like static graphs, the dynamic graph consists out of nodes and edges for which sizes and colours are adjusted accordingly. However, with d3graph we create an interactive network that is stand-alone html file. The network is created with collision and charge parameters to ensure that nodes do not overlap. Each node contains a text-label, and the links of associated nodes can be highlighted when double clicked on the node of interest. For deeper examination of the network, it is also possible to break links based on the edge weights using a slider. Each node involves a tooltip that can easily be adapted to display any of the underlying data. We developed d3graph as a python function which outputs custom java script file based on user defined parameters. The custom java script file on its turn uses functionalities from the d3 java script library (version 3). In its simplest form, the input for d3graph is an adjacency matrix for which the elements indicate pairs of vertices are adjacent or not in the graph.

**Bayesian structure learning.** We utilized the Bayesian structure learning algorithm to learn the optimal DAG using the data sets based on a score-based approach, and under the assumption that the data is complete (no missing values). The score-based model selection approach consists out of two main parts, the scoring function, and the search strategy. The scoring function maps models to a numerical score based on how well the model fits to the given data set, whereas the search strategy enables selection of a model with optimal score across the search space of all possible models. We used Bayesian information criterion (BIC) as the scoring functions to measure the fit between model and data, and hill-climbing as search strategy. We ran Bayesian structure learning on data set containing the varying set of samples and selected the best scoring model. Just like d3graph, we also developed a separate library for Bayesian structure learning, build on library for working with Probabilistic Graphical Models (pgmpy), that allows functionalities for structure learning on a given data set or DAG.

**Software Architecture.** The HNet library is built using Python and contains three main components. The library set\_*dtypes* handles the typing of the input variables that are available in the data frame. *df2onehot* converts and controls the input data to creates a one-hot array based on user-defined parameters. *d3graph* creates a dynamic and interactive network in d3. Code for HNet is available at <https://github.com/XXX>, whereas for d3graph is available at <https://github.com/XXX>. In addition, we also created a library for Bayesian structure learning at <https://github.com/XXX>. Usage examples for each library can be found on github.

# **Data**

**Data**. Directed Acyclic Graphs (DAG) of Sprinkler and Alarm16 are used to generate a data set by means Bayesian inference and forward sampling. Generated data sets vary in sample sizes; N=[100,1000,5000,10000]. The number of nodes for the Sprinkler model contains 4 nodes, 4 arcs, and 8 parameters. The Alarm model contains 37 nodes, 46 arcs and 509 parameters. The titanic data set contains 891 samples with 12 feature columns with mixed data types.

# **Results**

In this work, we evaluate our model in the detection of edge probabilities between pairs of vertices for both directed and undirected node-links. We perform experiments on two synthetic data sets with varying number of parameters, and with known ground truth. Finally, we perform an experiment on the titanic data set which is a well-known unstructured data set with many similarities of a real-world applications as it contains in its raw form continues, discrete and categorical variables.

**Sprinkler data set.** A natural way to study the relation between nodes in a network is to analyse the presence or absence of node-links. The sprinkler data set contains four nodes and therefore ideal to demonstrate the working of HNet in inferring a network. Links between two nodes of a network can either be undirected or directed (directed edges are indicated with arrows). Notably, a directed edge does imply directionality between the two nodes whereas undirected does not. We generated data using Bayesian forward sampling using the Conditional Probability Distributions (CPDs) for the Sprinkler system as shown in Figure 2A. Each node consists of two states, either being True or False for which we sampled with N=100, 1000, 5000, and 10000 samples. The true state is commonly used as response variable whereas the false state is seen as background. In this case the false state also describes the condition of the variable and is therefore also used as response variable. This means that in this model, the four nodes are split into eight nodes, each describing a state. The results using HNet for 1000, 5000 and 10000 samples showed consistent similar detection of significant node-links (*Pholm*<0.05, Figure 2B and C). The inferred network contains both directed and undirected edges and is a good representation of the initial CPD. As an example, an edge is seen between Wet Grass and Sprinkler is True. When the Sprinkler is on, there is also an association with no Rain (false), and no Clouds (false). On the other hand, when the sprinkler is off (False), an associated is seen with Cloudy and Rain. When we lower the sample size to N=100 samples, we see the absence of node-links (coloured orange in Figure 2C) compared to the data set with >1000 samples. To examine the minimum number of samples that shows the same node-links as seen with >1000 samples, we gradually increased the number of samples from 100 towards 1000 in steps of 10. We demonstrate convergence of number of edges after approximately 400 samples, depending on the multiple test correction (Figure 2D). In addition, we compared the exact node-link to the network with >1000 samples. We show that detected edges converge by an increasing number of samples to a network build on a large sample size.

**Alarm data set.** The Alarm data set is a medium to large network containing 37 nodes with a total of 509 parameters. The data set is used to compare the performance of HNet with Bayesian structure learning, random results and the golden truth. To generate a data set with a ground truth, we used Bayesian forward sampling with Conditional Probability Distributions (CPDs) of the Alarm system. We sampled with N=1000, 5000 and 10.000 samples. Because some nodes consist more than two states, we only considered the true states as response variable to avoid analysing mixed background groups. Because the golden truth of node-links and edge directionality is known for the network, it can be used to examine the performance of HNet compared to Bayesian structure learning. We did setup three experiments: 1. HNet versus golden truth, 2. Bayesian structure learning versus golden truth, and 3. random adjacency matrix versus the golden truth. Each experiment is performed for the detection of directed and undirected edges (Figure 3A). The performance is scored using Matthews correlation coefficient (MCC). Note that MCC is a measure to quantify the quality of binary classifications, in this case the detected node-links and its directionality. Coefficient values range between -1 and +1 with coefficient of +1 represents a perfect prediction, 0 an average random prediction and -1 an inverse prediction. The results for N=1000, towards 10.000 samples, including edge directionality, showed an average MCC score of 0.23 + 0.0001 (*P*<1x10-4) for HNet. Bayesian structure learning showed an average MCC score of 0.34 + 0.009 (*P*=1x10-10), and the average MCC score when using random edges is 0.004 + 0.0003 (*P*=0.4). We also analysed the specificity of the various models in the detection of undirected node-links. To make the results comparable across the various models we symmetrized the elements on the adjacency matrix with respect to the diagonal. The average MCC score for the detection of undirected edges in HNet is 0.33 + 0.0002 (*P*<1x10-6), for Bayesian structure learning the average MCC score is 0.52 + 0.006 (P<1x10-11), and the average MCC score when using random node-links is 0.004 + 0.0003 (P=0.49). Finally, we also compared the undirected adjacency matrix of the golden truth towards the directed model which results in an average MCC score of 0.69 (*P*<1x10-13).

**Titanic data set.** The titanic data set contains a data structure that is often seen in real use cases (i.e., the presence of categorical, boolean, and continues variables per sample) which is therefore ideal to demonstrate the steps of HNet, and to show the interpretability. The first step is the typing of the 12 input features, followed by one-hot encoding (Figure 4A). This resulted in a total of 2634 one hot encoded features for which only 18 features had the minimum required of 10 samples; Survived [1,0], Pclass [1,2,3], Sex [female,male], Sibsp [0,1,3,4], Parch [0,1,2], Cabin, Embarked [C,Q,S]. The total number of features for the model is 20, which includes the two numeric features; Fare and Age. The next step in HNet is to determine the node-links for which in total 60 unique edges across 47 nodes is detected (alpha=0.05 and multiple testing correction is Holm, Figure 2A). Note that the detected node-links can be indicative for directionality, as an example no survival (survived=0) is significantly associated with males, but not the other way around. Therefore, an directionality can be seen from males to no-survival. Although the ground truth of this data set is unknown, the strongest association is in line with intuitive expectations, i.e., first class passengers are significantly associated with High Fare (fare>60.3, P<2.87-79), whereas third class passengers are significantly associated with low fare (fare<8.1, P<4.99-73). The next best association is between passengers that are female and survived (P<4.79-57), followed by male passengers that did not survived (*P*=4.79-57). The network graph is consistently expanded across the survived yes/no clusters. The male-no survival cluster is expanded with low fare, having no siblings, embarking position is S or Q whereas the females-survival cluster is expanded with high fare, having 1 sibling, and embarking position is C. Interestingly, directionality for passengers that did not survived is outwards whereas those that survived is mainly inwards. This may suggest that survived passengers are on coordinated actions, whereas it is not for passengers that did not survive.

# **Discussion**

Taken together, we demonstrate the detection of statistically significant associations in (un)structured data sets using HNet. The detected edges between nodes can be either directed or undirected. However, it should be noted that the directionality indicates that a feature is statistically overrepresented but does not necessarily imply causality. Furthermore, HNet provides deterministic results which can be deeper examined using the interactive network.

# **Author Contributions**

ET designed the study, analysed the data, drafted the manuscript, and developed the python code.

# **Competing interests**

The author has declared that no competing interests exist.

# **References**

1. Epasto, A. & Perozzi, B. Is a single embedding enough? Learning node representations that capture multiple social contexts. in *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019* (2019). doi:10.1145/3308558.3313660.

2. Qiu, J. *et al.* Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec. in *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining* (2018). doi:10.1145/3159652.3159706.

3. Grover, A. & Leskovec, J. Node2vec: Scalable feature learning for networks. in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016). doi:10.1145/2939672.2939754.

4. Tang, J. *et al.* LINE: Large-scale information network embedding. in *WWW 2015 - Proceedings of the 24th International Conference on World Wide Web* (2015). doi:10.1145/2736277.2741093.

5. Friedman, N. & Koller, D. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Mach. Learn.* (2003) doi:10.1023/A:1020249912095.

6. Braun, D. A., Mehring, C. & Wolpert, D. M. Structure learning in action. *Behavioural Brain Research* (2010) doi:10.1016/j.bbr.2009.08.031.

7. Binder, J., Koller, D., Russell, S. & Kanazawa, K. Adaptive Probabilistic Networks with Hidden Variables. *Mach. Learn.* (1997) doi:10.1023/A:1007421730016.

8. Lauritzen, S. L. & Spiegelhalter, D. J. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *J. R. Stat. Soc. Ser. B* (1988) doi:10.1111/j.2517-6161.1988.tb01721.x.

9. Sachs, K., Perez, O., Pe’er, D., Lauffenburger, D. A. & Nolan, G. P. Causal protein-signaling networks derived from multiparameter single-cell data. *Science (80-. ).* (2005) doi:10.1126/science.1105809.

10. Tsamardinos, I., Brown, L. E. & Aliferis, C. F. The max-min hill-climbing Bayesian network structure learning algorithm. *Mach. Learn.* (2006) doi:10.1007/s10994-006-6889-7.

11. Chickering, D. M. Learning Bayesian Networks is NP-Complete. in (1996). doi:10.1007/978-1-4612-2404-4\_12.

12. O’Gorman, B., Babbush, R., Perdomo-Ortiz, A., Aspuru-Guzik, A. & Smelyanskiy, V. Bayesian network structure learning using quantum annealing. *European Physical Journal: Special Topics* (2015) doi:10.1140/epjst/e2015-02349-9.

13. Agrawal, R. & Srikant, R. Fast Algorithms for Mining Association Rules in Large Databases. in *Proc. of the 20th International Conference on Very Large Data Bases (VLDB’94)* (1994).

14. Zaki, M. J. Scalable algorithms for association mining. *IEEE Trans. Knowl. Data Eng.* (2000) doi:10.1109/69.846291.

15. Martellini, M. & Malizia, A. *Cyber and Chemical, Biological, Radiological, Nuclear, Explosives Challenges: Threats and Counter Efforts*. (2017).

16. Beinlich, I. A., Suermondt, H. J., Chavez, R. M. & Cooper, G. F. The ALARM Monitoring System: A Case Study with two Probabilistic Inference Techniques for Belief Networks. in (1989). doi:10.1007/978-3-642-93437-7\_28.

17. Singh, A., Saraswat, S. & Faujdar, N. Analyzing Titanic disaster using machine learning algorithms. in *Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2017* (2017). doi:10.1109/CCAA.2017.8229835.

18. Benjamini, Y., Drai, D., Elmer, G., Kafkafi, N. & Golani, I. Controlling the false discovery rate in behavior genetics research. in *Behavioural Brain Research* (2001). doi:10.1016/S0166-4328(01)00297-2.

19. Hochberg, Y. A sharper bonferroni procedure for multiple tests of significance. *Biometrika* (1988) doi:10.1093/biomet/75.4.800.

20. Holm, S. A simple sequentially rejective multiple test procedure. *Scand. J. Stat.* (1979).

# **Figures and Tables**

**Figure 1. Method overview HNet.** (A) XXX (B) XXXX (C) XXX (D) XXX.

**Figure 2. Results on the Sprinkler model using HNet**. (A) Conditional Probability Distributions (CPD) that is used to generate data using Bayesian forward sampling. (B) Adjacency matrix determined by HNet with an input of N=1000 samples of the Sprinkler CPD. Elements (node-links) are coloured based on the -log10(*P*holm). The diagonal is colored red for visualization purposes. (C) Graph network view of the adjacency matrix. Node size is based on the percentage of available labels. Node color is based on the unique feature names. Edge width is set by the -log10(*P*holm) value. The edges with color grey are only seen up to sample size is 100. For larger sample sizes >1000, both orange and grey edges are consistently detected. (D) Number of detected edges for a varying number of samples when using multiple test correction; Holm, Bonferonni or Benjamini/Hochberg. (E) Comparison of node-links using N=1000 samples versus a varying number of samples. The results are based on three multiple test correction methods; Holm, Bonferonni or Benjamini/Hochberg.

**Figure 3. Results on Asia model using HNet.** For the Asia CPD we sampled up to N=10.000 samples and compared the performance between HNet and Bayesian structure learning, random results and the golden truth. (A) MCC score between models for directed is shown with dashed line, and straight line for directed. Models are depicted with different colores; HNet=red, Bayesian structure learning=blue, Random=black, golden truth=yellow. (B) Costs in runtime for HNet and Bayesian structure learning over various samples.

**Figure 4. Results on Titanic data set using HNet.** (A) Table depicts the input features in the model, the original typing, the typing by HNet, the number of unique labels per feature, and the remaining labels that agree with the minimum of 10 samples. (B) Adjacency matrix determined by HNet for which the elements in the matrix are the node-links that are coloured on the -log10(*P*holm). (C) Graph network view of the adjacency matrix. Node size is based on the percentage of available labels. Node colour is based on the unique feature names. Edge width between nodes is set by the -log10(*P*holm) value.