SUPPLEMENTARY MATERIAL
OWL-NETS: Transforming OWL Representations for Improved Network Inference

Callahan, Baumgartner, Bada, Stefanski, Tripodi, White, and Hunter

1. Acronyms and Definitions

1.1. *Acronyms*

- **AUC:** Area under the receiver operating characteristic curve
- **CCDF:** Complementary Cumulative Distribution Function
- **OBO:** Open Biomedical Ontologies
- **SPARQL:** SPARQL Protocol RDF Query Language
- **OWL:** Web Ontology Language

1.2. *Definitions*

- **Clique:** A complete subgraph within a network.
- **Clustering Coefficient:** The measure of how much the nodes of a network tend to cluster together.
- **Connected Component:** A component is said to be connected if any two vertices can be connected by by a path.
- **Degree:** The degree of node is determined by counting the number of edges that are connected to it.
- **Degree Heterogeneity:** A measure of how much the degree distribution of a network deviates from a ”regular network”.[1]
- **Diameter:** The Shortest distance between the two most distant nodes in the network.
- **Disassortativity:** A network is said to have a dissassortative structure if high degree nodes tend to be connected to lower degree nodes.

2. Link Prediction Algorithms

The eight local similarity algorithms, defined consistent with the literature,[2,3] are:

(1) Degree Product: Given two nodes i and j, this measure is calculated as the product of the degree (i.e., the number of connected nodes) of nodes i and j, where k is the node degree:

$$score(i,j) = k_i k_j \tag{1}$$

(2) Common Neighbors: Given two nodes i and j, this measure is calculated as the number of neighbors that are common to both nodes i and j, where $\Gamma(j)$ is the set of nodes connected to node j:

$$score(i,j) = \mid \Gamma(i) \cap \Gamma(j) \mid \tag{2}$$

(3) Jaccard Coefficient:[4] Given two nodes i and j, this measure is calculated as the number of neighbors that are common to both nodes i and j normalized by the number of nodes adjacent to either node i or node j:

$$score(i,j) = \frac{\mid \Gamma(i) \cap \Gamma(j) \mid}{\mid \Gamma(i) \cup \Gamma(j) \mid} \tag{3}$$

(4) Srenson Similarity:[5] Given nodes i and j, this measure is calculated as the number of neighbors that are common to both nodes i and j normalized by the sum of the degrees of node i and node j:

$$score(i,j) = \frac{\mid \Gamma(i) \cap \Gamma(j) \mid}{k_i + k_j} \tag{4}$$

(5) Leicht-Holme-Newman:[6] Given two nodes i and j, this measure is calculated as the number of neighbors that are common to both nodes i and j normalized by the product of the degrees of node i and node j:

$$score(i,j) = \frac{\mid \Gamma(i) \cap \Gamma(j) \mid}{k_i * k_j} \tag{5}$$

(6) Shortest Paths: Given two nodes i and j, this measure is calculated as the reciprocal of the length of the shortest path from node i to node $j(\sigma(i,j))$:

$$score(i,j) = \frac{1}{\sigma(i,j)} \tag{6}$$

A score of zero is given for all node pairs not connected by a path.

(7) Resource Allocation:[7] Given two nodes i and j, this measure is calculated as the sum of the reciprocal of the degrees of nodes adjacent to both nodes i and j:

$$score(i,j) = \sum_{z \in \Gamma(i) \cap \Gamma(j)} \frac{1}{k_z} \tag{7}$$

(8) Adamic-Advar:[8] Given two nodes i and j, this measure is calculated as the sum of the reciprocal of the log of the degrees of the nodes adjacent to both nodes i and j:

$$score(i,j) = \sum_{z \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log(k_z)} \tag{8}$$

The two global similarity algorithms, defined consistent with the literature,[9] are:

(1) Katz:[10] Given an unweighted adjacency matrix A, $A_{i,j}$ is one if there is a link between nodes i and j and zero if there is not. Each element of A_{ij}, A^k has value equal to the number of walks with length k between nodes i and j:

$$score(i,j) = \sum_{k=1}^{\infty} \beta^k A_{ij}^k \tag{9}$$

where β, must be lower than the largest eigenvector of matrix A, that is used to give shorter paths more weight. Consistent with the literature,[11] a value of $\beta = 0.001$ was used.

(2) Rooted PageRank: A random walker starts from node i and randomly moves to a neighbor of node i. The walker then has a probability of $1 - \alpha$ for teleporting back to node i. Consistent with the literature,[11] a value of $\alpha = 0.15$ was used.

Algorithm 1: Construction of Query Graph and Identification of NETS Nodes

```
Input  : query: a text file containing a SPARQL query
Output: G: a directed graph where edges represent triples from the
        input SPARQL query, NETSnodes: a list of NETS nodes
$G \leftarrow$ empty directed network;
for line in query do
    if line starts with ? then
        split line on ? into three parts;
        $subject \leftarrow part1$;
        $predicate \leftarrow part2$;
        $object \leftarrow part3$;
        $edge \leftarrow (subject, object)$;
        $edgeLabel \leftarrow predicate$;
        if 'identifier' in predicate then
            $NETSnodes \leftarrow object$;
        end
    end
    $G \leftarrow (edge, edgeLabel)$ ;
end
return $G$, $NETSnodes$;
```

Algorithm 2: Identification of NETS Edges

```
Input  : G: a directed graph where edges represent triples from the
         input SPARQL query, NETSnodes: a list of NETS nodes
Output: NETSedges: a hash table, where keys are NETS edges and
        values are the shortest paths through G connecting the nodes
        in the edge
$NETSedges \leftarrow$ empty hash table;
for source.node in NETSnodes do
    for target.node in NETSnodes do
        if length of $G.shortestPath(source.node, target.node) > 0$ then
            $paths \leftarrow G.$shortestPath$(source.node, target.node)$;
        end
        for path in paths do
            if $owl : Restriction$ in path then
                $source \leftarrow source.node$;
                $target \leftarrow target.node$;
            end
            $NETSpath \leftarrow path$;
            $edge \leftarrow (source, target)$;
            BREAK;
        end
    end
    $NETSedges \leftarrow key(edge), value(NETSpath)$
end
return $NETSedges$;
```

Algorithm 3: Creation of Network Node and Edge Metadata

```
Input  : $G$: a directed graph where edges represent triples from the
         input SPARQL query, $NETSnodes$: a list of NETS nodes,
         $NETSedges$: a hash table, where keys are NETS edges and
         values are the shortest paths through $G$ connecting the nodes
         in the edge
Output: updatedG: a directed graph where edges represent triples from
         input SPARQL query in addition to NETS node and edge
         metadata, $NETSNodeMetadata$ a hash table, where the keys
         are $NETSnodes$ and the values are a list of tuples of node
         labels and identifiers, $NETSEdgeMetadata$ a hash table,
         where the keys are $NETSedges$ and the values are a tuple,
         where the first item is the shortest path and the second item is
         a list of tuples of edge labels and identifiers
$NETSNodeMetadata \leftarrow$ empty hash table;
$NETSEdgeMetadata \leftarrow$ empty hash table;
for $node$ in $NETSnodes$ do
    updatedG ← SPARQL syntax as triple to retrieve node labels and
      identifiers;
    $NETSNodeMetadata \leftarrow$
      $key(NETSnode), value([(label, identifier)])$;
end
return updatedG;
for $edge$ in $NETSedges$ do
    updatedG ← SPARQL syntax as triple to retrieve edge labels and
      identifiers;
    $NETSEdgeMetadata \leftarrow$
      $key(NETSedge), value(NETSpath, [(label, identifier])$;
end
return updatedG, $NETSNodeMetadataz$, $NETSEdgeMetadata$;
```

Algorithm 4: Construction of OWL-NETS Abstraction Network

```
Input  : $queryresults$: a json file containing the results from running a
         SPARQL query against a knowledge base, $NETSedges$: a hash
         table, where keys are NETS edges and values are the shortest
         paths through $G$ connecting the nodes in the edge,
         $NETSNodeMetadata$ a hash table, where the keys are
         $NETSnodes$ and the values are a list of tuples of node labels
         and identifiers, $NETSEdgeMetadata$ a hash table, where the
         keys are $NETSedges$ and the values are a tuple, where the first
         item is the shortest path and the second item is a list of tuples
         of edge labels and identifiers
Output: $OWL-NETS$: a directed OWL-NETS graph of an input
         SPARQL query, $Metadata$: a json file containing metadata for
         the $OWL-NETS$ graph
$OWL-NETS \leftarrow$ empty directed network;
$Metadata \leftarrow$ empty dictionary;
for $result$ in $queryResults$ do
    for $edges$ in $NETSedges$ do
        if $edges[0]$ in $result$ then
            $source \leftarrow result[edges[0]]$;
            if $NETSNodeMetadata[edges[0]]$ in $result$ then
                $label \leftarrow result[NETSNodeMetadata[edges[0]]]['label']$;
                $identifier \leftarrow result[NETSNodeMetadata[edges[0]]]['id']$;
                $sourceMeta \leftarrow (label, identifier)$;
            end
        end
        if $edges[1]$ in $result$ then
            $target \leftarrow result[edges[1]]$;
            if $NETSNodeMetadata[edges[1]]$ in $result$ then
                $label \leftarrow result[NETSNodeMetadata[edges[1]]]['label']$;
                $identifier \leftarrow result[NETSNodeMetadata[edges[1]]]['id']$;
                $targetMeta \leftarrow (label, identifier)$;
            end
        end
        if $NETSEdgeMetadata[edges]$ in $result$ then
            $label \leftarrow result[NETSEdgeMetadata[edges]]['label']$;
            $identifier \leftarrow result[NETSEdgeMetadata[edges]]['id']$;
            $edgeMeta \leftarrow (label, identifier)$;
        end
        $OWL-NETS \leftarrow (source, target)$;
        $Metadata \leftarrow (sourceMeta, targetMeta, edgeMeta)$;
    end
end
return $OWL-NETS$, $Metadata$;
```

Table S1. Query 1 Descriptive Network Properties by Network Representation

Property	OWL	OWL-NETS	p-value
Nodes	1578.400 (155.850)	247.950 (22.741)	< 0.0001
Edges	4110.930 (445.103)	1130.100 (153.514)	< 0.0001
Average Degree	5.204 (0.091)	9.083 (0.473)	< 0.0001
Density	0.003 (0.000)	0.037 (0.002)	0.002
Diameter	10.000 (0.000)	5.880 (0.325)	< 0.0001
Clustering Coefficient	0.067 (0.005)	0.338 (0.013)	0.013
Degree Assortativity	-0.223 (0.001)	-0.122 (0.019)	0.019
Degree Heterogeneity	13.684 (1.542)	2.354 (0.185)	< 0.0001
Number of Shortest Paths	5694.170 (1054.645)	683.680 (116.755)	< 0.0001
Average Shortest Path Length	3.760 (0.039)	2.934 (0.048)	< 0.0001
Number of Cliques	3532.050 (376.901)	703.110 (95.554)	< 0.0001

*All descriptives were run on the undirected versions of the networks.

Table S2. Descriptive Network Properties by Network Representation and Query

Property	Q2:OWL	Q2:OWL-NETS	Q3:OWL	Q3:OWL-NETS
Nodes	840	59	22,679	1783
Edges	1426	59	33,848	3940
Average Degree	3.419	2.000	2.980	4.420
Density	0.0040	0.0344	0.0001	0.0020
Diameter	13	4	9	18[a]
Degree Assortativity	-0.193	-0.630	-0.124	-0.308
Degree Heterogeneity	8.789	4.533	558.463	6.673
Number of Shortest Paths	2,683	202	91,483	8,986[a]
Average Shortest Path Length	4.06	3.19	4.13	6.54[a]

*Query (Q). All descriptives were run on the undirected versions of the networks.

[a] Query 3:OWL-NETS statistics were derived on the largest connected component.

Table S3. Query 2 Link Prediction Algorithm Run-Time

Algorithm	OWL	OWL-NETS
Degree Product	00:00:07	00:37:42
Shortest Path	00:00:10	01:49:08
Common Neighbors	00:00:07	00:31:47
Jaccard Coefficient	00:00:07	00:32:04
Sorenson Similarity	00:00:07	00:32:09
Leicht-Holme-Newman	00:00:07	00:32:09
Adamic Advar	00:00:08	00:32:51
Resource Allocation	00:00:07	00:32:36
Katz	00:00:30	08:12:51
Rooted PageRank	00:01:00	12:16:21

*Run-time calculated as the total time to run 100 iterations of each link prediction algorithm.

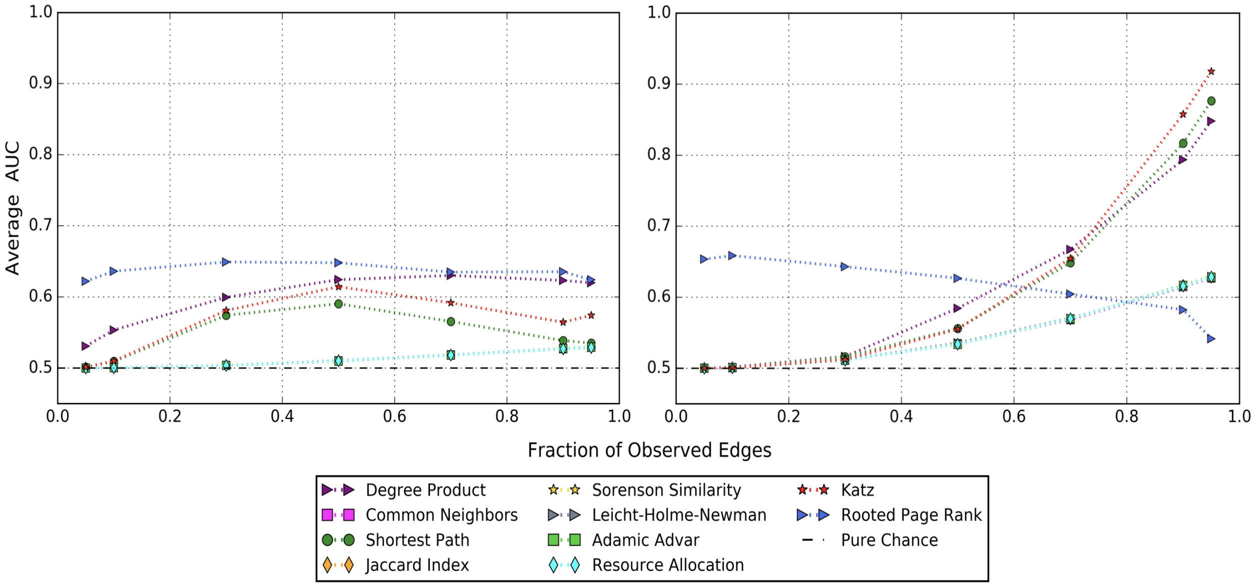

Fig. S1. Comparison of Link Prediction Methods by Network. (left) The original OWL representation network and (right) the OWL-NETS abstraction networks created from Query 2 (Table1).

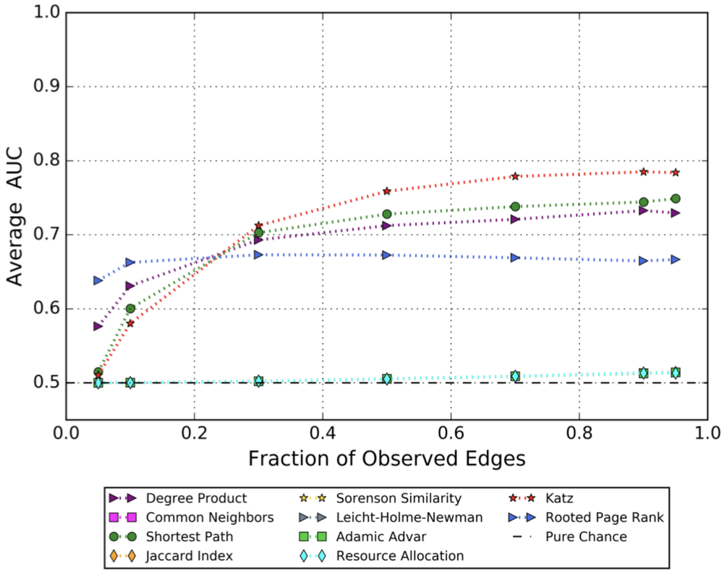

Fig. S2. Link Prediction Methods. The OWL-NETS abstraction networks created from Query 3 (Table1).

Table S4. Top Scoring Edges from the Query 3 OWL-NETS Abstraction Networks (n=6 edges)

Node 1	Node 2	Description
AG-1067[a]	MMP2[b]	AGI-1067 is derived from probucol, which has been shown to decrease MMP-2 expression and activity in Apolipoprotein E-deficient mice.[12]
DB03683[a]	APAF1[b]	DB03683a targets MMP9 through an unknown mechanism. Downregulation of MMP9 induces APAF1 expression.[13]
celiprolol[a]	CYCS[b]	Celiprolol is an investigational drug used to treat hypertension. Cytochrome c has been shown to mediate hypertension in rats and in humans.[14,15]
1454838[c]	TF[b]	Transferrin (TF) TF binds to and transports iron. Iron is required for the proliferation of multiple myeloma cells. CD147 (1454838) is overexpressed in multiple myeloma cells and is positively correlated with cell proliferation.[16,17]
DB04513[a]	RAF1[b]	DB04513 targets Calmodulin 1, which can regulate the threshold for activation of the Ras/Raf/MEK/ERK signaling pathway.[18]
CXCL12[b]	DB07691[a]	DB07691 is an n-phenylbenzamide, which can inhibit the Mitochondrial Permeability Transition Pore whose continual opening is associated with mitochondrial dysfunction. CXCL12 regulates mitochondria association around the MTOC (microtubule organizing center).[19,20]

[a] DrugBank entity (DrugBank ID used for experimental compounds); [b] Uniprot entity (gene symbol shown); [c] Reactome entity (database identifier).

References

1. E. Estrada (ed.), *The structure of complex networks: theory and applications* (Oxford University Press, 2012).
2. A. Clauset, C. Moore and M. E. J. Newman, *Nature* **453**, 98 (2008).
3. S. Soundarajan and J. Hopcroft, Using community information to improve the precision of link prediction methods, in *Proceedings of the 21st International Conference on World Wide Web, ACM*, 2012.
4. P. Jaccard, *Bull Soc Vaudoise Sci Nat* **37**, 547 (1901).
5. J. A. Hanley and B. J. McNeil, *Biol Skr* **5**, 1 (1948).
6. E. A. Leicht, P. Holme and M. E. Newman, *Physical Review E* **73**, p. 026120 (2006).
7. T. Zhou, L. Lu and Y.-C. Zhang, *Eur Phys J B* **71**, 623 (2009).
8. L. A. Adamic and E. Adar, *Soc Networks* **25**, 211 (2003).
9. R. Guns, *J Data Inform Sci* **1**, 59 (2016).
10. L. Katz, *Psychometrika* **18**, 39 (1953).
11. D. Liben-Nowell and J. Kleinberg, *J. Am. Soc. Inf. Sci.* **58**, 1019 (2007).
12. B. J. Wu, N. D. Girolamo, K. Beck, C. G. Hanratty, K. Choy, J. Y. Hou, M. R. Ward and R. Stocker, *J Pharmacol Exp Ther* **321**, 477 (2007).
13. C. S. Gondi, N. Kandhukuri, D. H. Dinh, W. C. Olivero, M. Gujrati and J. S. Rao, *Int J Oncol* **33**, 783 (2008).
14. C. P. Venditti, M. C. Harris, D. Huff, I. Peterside, D. Munson, H. S. Weber, J. Rome, E. M. Kaye, S. Shanske and S. Sacconi, *J Inherit Metab Dis* **27**, 735 (2004).
15. W.-Z. Ying and P. W. Sanders, *Kidney Int* **59**, 662 (2001).
16. B. K. Arendt, D. K. Walters, X. Wu, R. C. Tschumper, P. M. Huddleston, K. J. Henderson, A. Dispenzieri and D. F. Jelinek, *Leukemia* **26**, 2286 (2012).
17. K. VanderWall, T. R. Daniels-Wells, M. Penichet and A. Lichtenstein, *J Inherit Metab Dis* **18**, 449 (2013).
18. N. Agell, O. Bachs, N. Rocamora and P. Villalonga, *Cell Signal* **14**, 649 (2002).
19. G. Morlino, O. Barreiro, F. Baixauli, J. Robles-Valero, J. Gonzalez-Granado, R. Villa-Bellosta, J. Cuenca, C. O. Sanchez-Sorzano, E. Veiga, N. B. Martn-Cfreces and F. Snchez-Madrid, *Mol Cell Biol* **34**, 1412 (2014).
20. S. Roy, J. ileikyt, B. Neuenswander, M. P. Hedrick, T. D. Chung, J. Aub, F. J. Schoenen, M. A. Forte and P. Bernardi, *ChemMedChem* **11**, 283 (2016).