RelaViz: Graph Visualization of Learned Relations Between Entities

Joel Ferstay*

University of British Columbia

ABSTRACT

RelaViz is a tool for graphically visualizing the behaviour of relational learning algorithms. Its purpose is to help algorithm designers view and probe the sometimes hidden activity of an algorithm, to allow them to assess its performance. The class of relational learning algorithms considered form relations between entities. This relational data is in the form of triplets, (e^l, r, e^r), where e^l represents the left entity, r represents the relation, and e^r represents the right entity. Given a list of entities, *E*, relations, *R*, and training relational triplets, *X*, the relational learning algorithm learns new relational triplets. Associated with each new triplet is a degree of confidence of the algorithm that the new relation is correct. The RelaViz tool allows designers to view the identity of new relations formed between entities, as well as the algorithm's associated measure of confidence that these new asserted relations are correct. Webs of relations and entities can be large, however, and this necessitates targeted viewing of the relational data space. RelaViz is specifically designed to leverage properties that the algorithm designer is interested in, to target viewing of relations the algorithm has learned, while still having enough flexibility to explore the entire space of entities and relations if they choose. Furthermore, RelaViz allows the designer to view the progress of their algorithm in learning new relations – this is important because the class of relational learning algorithms considered are executed for a designer-specified, fixed number of iterations. RelaViz could help the designer determine when the algorithm is done the learning process.

1 INTRODUCTION

Can computers be taught common sense? Relational learning algorithms are machine learning devices that attempt to capture common sense by learning relations between entities in the world [10]. Machine learning algorithms need data, however, and the specific data required are a list of entities, relations, and entity-relation-entity data.

Entity-relation-entity data usually come in the form of a triplet, (e^l, r, e^r), where e^l stands for left entity, r stands for relation, and e^r stands for right entity. A particular instantiation of this general triplet could be (grass, grows_in, dirt). Some researchers in artificial intelligence have suggested that we might simulate human common sense by storing enough relational facts about the world, and generalizing from them appropriately to unobserved propositions [10]. To illustrate this process, suppose we are given propositions such as (mug, used_for, drinking), (mug, can_contain, coffee), (mug, can_contain, juice), (mug, can_contain, water), (cup, can_contain, juice), (cup, can_contain, water), (cup, can_contain, milk). From these propositions, one might predict the propositions (cup, used_for, drinking), (cup, can_contain, coffee), and (mug, can_contain, milk) [10]. Existing entity-relation-entity data needs to come from somewhere, however, and to produce high quality proposition predictions, machine learning algorithms typically require a lot of data. This is where accessible relational data on the web comes into play, along with Knowledge Bases (KB)s, and the need for structured embeddings of these KBs to make their data accessible to machine-learning algorithms.

In order to gather, organize, and make deliberate use of massive amounts of information generated daily, special kinds of web-based relational database specifically designed for knowledge management, collection, and retrieval called Knowledge Bases have been built [3]. A large amount of data regarding general and specific knowledge are now available online: OpenCyc, WordNet, Freebase, DBpedia, etc. [3]. The problem for web-scale machine learning is that these KBs are highly structured and organized. Although this is a benefit for the original purpose a KB is intended for, it does mean that their knowledge is "locked-up" in a symbolic framework that is not flexible enough to be exported for use in machine-learning algorithms [3]. To solve this problem, Bordes et al describe and implement a statistical machine learning approach that learns to represent elements of any KB in a low (e.g. 50) dimensional vector space [3]. These embeddings are important for the production of relational learning data because they can then be used as input to a learning algorithm for learning relations of the type (e^l, r, e^r) [3].

Why might visualization be necessary for newly predicted entity-relation-entity data? One reason is there is currently no way of validating the new, predicted relations produced from learning algorithms automatically, other than the traditional method of having a set of completely known data, and splitting it into train and test sets. Visualization can be used as a kind of verification tool for whether or not the relations between entities make sense. To illustrate why good visualization is needed, let us look at a current example for displaying new, learned relations. In the study by Bordes et al, new, learned relational data are simply displayed in a table format for the reader to assess their plausibility, see Figure 1 [3]. This approach will not scale up to multiple relations between entities and viewing many entities and relations at a time.

Table 7: **Knowledge extraction**. Examples of lists of e^r predicted with the embeddings learnt out of raw text for e^l ="*people*". Lists are displayed by decreasing triplet probability density order.

e^l	people				
r	build	destroy	won	suffer	control
e^r	*livelihoods*	*icons*	*emmy*	*sores*	*rocket*
	homes	*virtue*	*award*	*agitation*	*stores*
	altars	*donkeys*	*everything*	*treatise*	*emotions*
	houses	*cowboy*	*standings*	*eczema*	*spending*
	ramps	*chimpanzees*	*pounds*	*copd*	*fertility*

Figure 1. The display used in Bordes et al for conveying their relational learning algorithm's effectiveness. Presenting the data in table format will not scale up to the visual inspection of many entity-relation-entity data and many relations between entities. *Note: Table 7 is taken from Bordes et al* [3].

With no truly expressive tools for assessing a relational learning algorithm's performance, and the inability to automate validation of new, learned relations, it is clear there is a need for an effective visualization tool. To this end, I present RelaViz. RelaViz is specifically designed to leverage properties that the algorithm designer is interested in to target viewing of relations their algorithm has learned, while still having enough flexibility to allow them to explore the entire space of relational data if they choose. Furthermore, RelaViz can allow the designer to view the progress of their algorithm in learning new relations – this is important since the class of relational learning algorithms considered are executed for a designer-specified, fixed number of iterations. RelaViz can help the designer determine when the algorithm is done the learning process. This paper is organized as follows: In Section 2, I will discuss related work. Section 3 will present an overview of RelaViz and its features. Section 4 will discuss the strengths and limitations of this solution, and Section 5 will discuss future directions for this work. Section 6 concludes the paper.

2 Related Work

At present, I could not find any visualization tools that deal explicitly with assessing a relational learning algorithm's performance. There are, however, other visualization tools that deal with the problem of debugging and assessing the performance an algorithm, by visualizing its output, in other domains. As well, many tools deal with the task of visualizing large graphs. I will draw inspiration for my design from these works.

Constellation is one such tool that can be considered as an algorithm performance assessment tool [8]. Constellation uses a graph layout algorithm to allow users to examine a large semantic network [8]. Another visualization tool used to assess algorithm performance is MizBee [7]. MizBee is a visualization tool for browsing synteny in comparative genomics [7]. This tool can help a designer assess their algorithm's performance by visualizing its output, and even led one designer to discover that his synteny algorithm was flawed, and to redesign it [7].

The relational graph used in this project could be quite large – the relational learning algorithm of Bordes et al considers 81, 061 nodes [3]. There are many studies that could help guide the presentation of this relational data in graph form. One concern to be addressed by RelaViz is viewing the identity of relations between entities even though they are attached to links that will appear very small if the entire graph were shown. Some approaches to very large graph visualization take the approach of "giving up" on simply displaying the entire graph and having every piece of it at your finger tips for inspection at all times. One such approach is demonstrated by the ASK-GraphView system, which displays simplified, selectable overview representations of a massive underlying graph, and shows the user a detailed node-link graph representation of a portion of the underlying graph based on selections on these representations [1]. RelaViz is similar in the sense that it uses an adjacency matrix, whose behaviour is analogous to that of the simplified selectable representation used in ASK-GraphView, to map to an underlying graph [1]. Inspiration for RelaViz is also drawn from the work, "Search, Show Context, Expand on Demand," which uses a measure of degree of interest to determine what the user observes – this inspiration is evident in the use of the derived attribute for expressing the degree of certainty in relations between entities to guide user navigation of a large underlying graph [11]. Finally, although this is the first application of a linked graph and adjacency matrix view to assessment of a relational learning algorithm that I am aware of, analogues of this linked-view approach are present in the literature. One such application of this linked approach is to social network data performed by the MatrixExplorer system. This system allows users to filter, cluster, and lay out data in different ways to find relationships in it [5]. Initial inspiration for representing relational data as a graph was taken from the survey paper on graph visualization by Herman et al [6].

3 RelaViz

To determine the design of RelaViz, activities a relational learning algorithm designer might want to perform are considered. These may include:

- Examining the identity of the relation between two entities: for instance, given that there exists a relation between entities cat and milk, what is it?
- Examining the directionality of the relation between two entities: Given cat likes milk, it is not necessarily the case that milk likes cat.
- Examining the probability that a relation is true: relational learning algorithms may produce relations with an associated probabilistic quantity of how likely the relation is.
- Determining the learning progress of the algorithm: given that this is an algorithm requiring a user-specified, fixed number of iterations, how do we help the user determine this parameter?

This list of tasks defines design criteria for the RelaViz system.

3.1 RelaViz Display

Upon starting the system, users will see the screen shown in Figure 2. Relational data is already loaded by default. The node-link graph view is on the right of the display, and is absent until a regional selection is made on the adjacency matrix view. The names to the top and left of the adjacency matrix display the names of the entities in the graph, and the cells display information pertaining to the number of links present between two entities, or a measure of the degree of certainty the algorithm has that the relation(s) between two entities is likely correct: the adjacency matrix view can display two types of information: a derived attribute expressing the sheer number of relations between two entities, and a derived attribute expressing the degree of confidence the algorithm has in the relational links formed between entities. The attribute expressing the number of relations between two entities is simply derived using a sum of the number of relations present between two entities, and is encoded using colour – a white square indicates no relation, and darker colours represent many relations. See Figure 4. The attribute expressing degree of certainty was derived using the sum of the certainties in each relation present between two entities, divided by the number of relations present between the entities. If the number of relations is zero, the square is left white. See Figure 4. Justification for the use of these

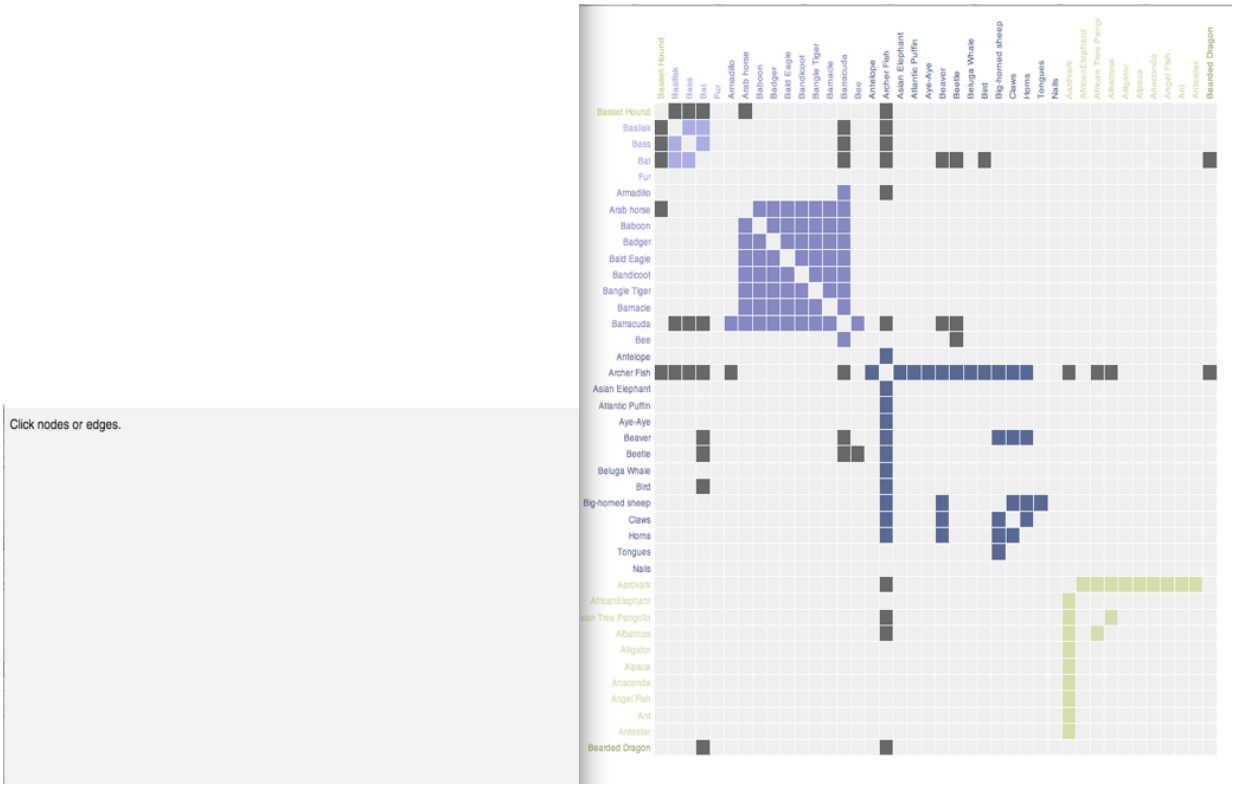

Figure 2. The RelaViz system at start-up. Only the adjacency matrix representation of the relational data is present initially. A relational data dataset is loaded by default – this will be changed in future iterations of the project. Colour in the adjacency matrix encodes information regarding two derived attributes: one encoding a measure of the number of relations between entities, and one aspiring to capture the confidence of the algorithm that the relations present here are correct. The two modes are toggled between by clicking the whitespace to the right of the matrix.

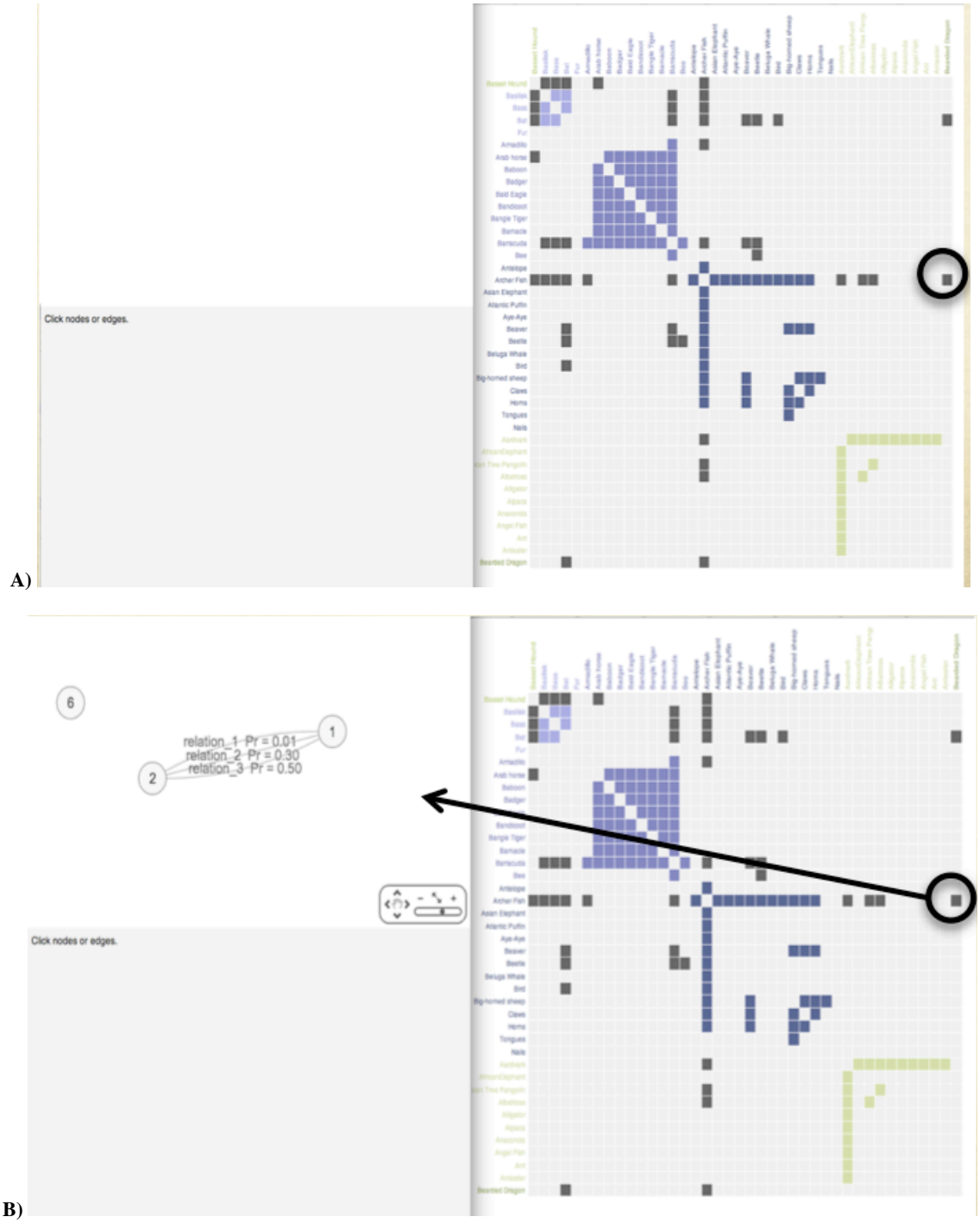

Figure 3. A) Selecting a region on the adjacency matrix will display the node-link graph representation corresponding to this patch. B) Nodes corresponding to this point on the adjacency matrix are displayed as a node-link graph on the left panel of the display.

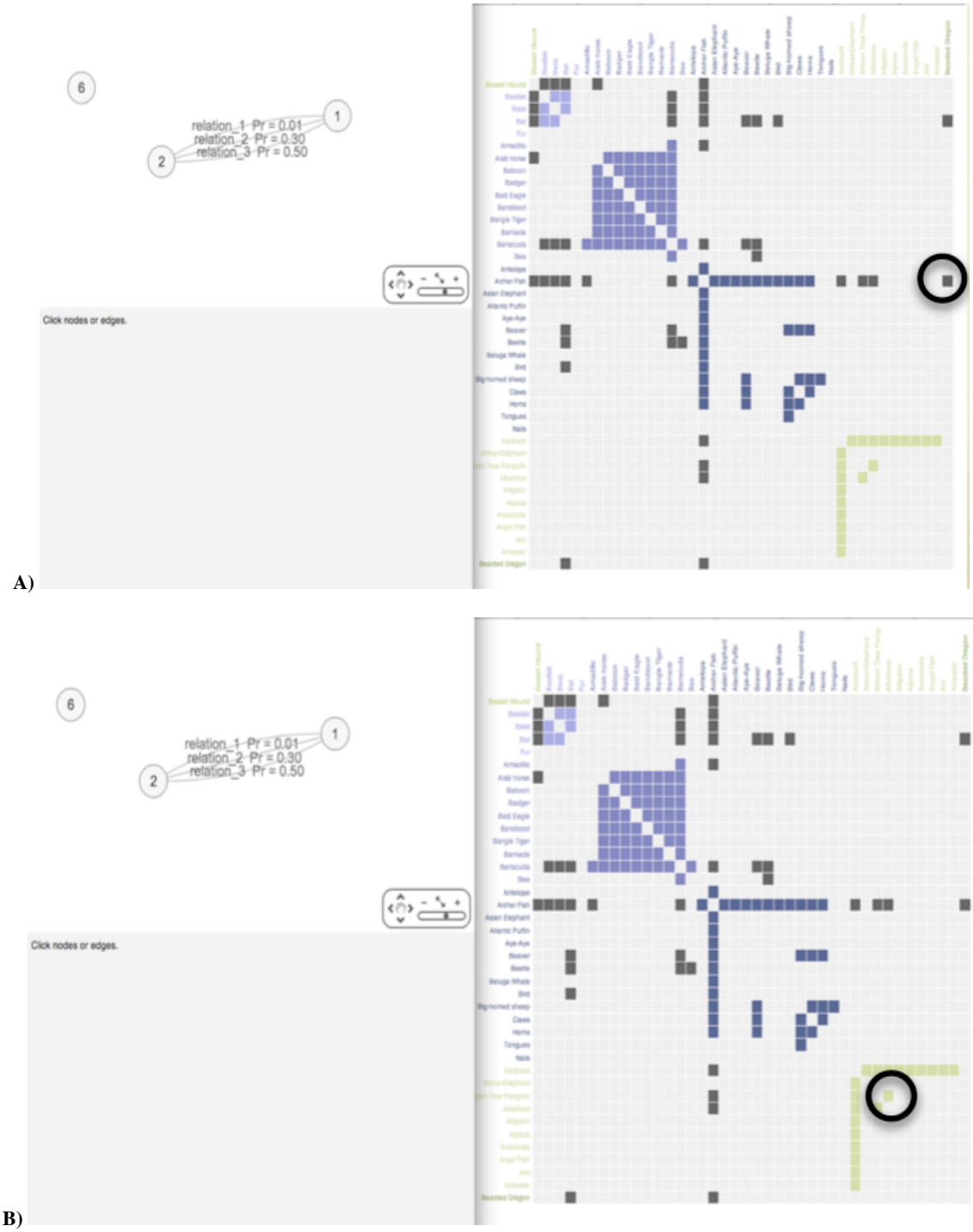

Figure 4. A) Darker colours encode greater quantities – here, this circled dark marking encodes either a lot of relations between these two entities, or a high degree of certainty in the relations formed between these two entities. B) Lighter colours encode smaller quantities – here, this circled light marking encodes either only a few relations between these two entities, or a low degree of certainty in the relations formed between these two entities.

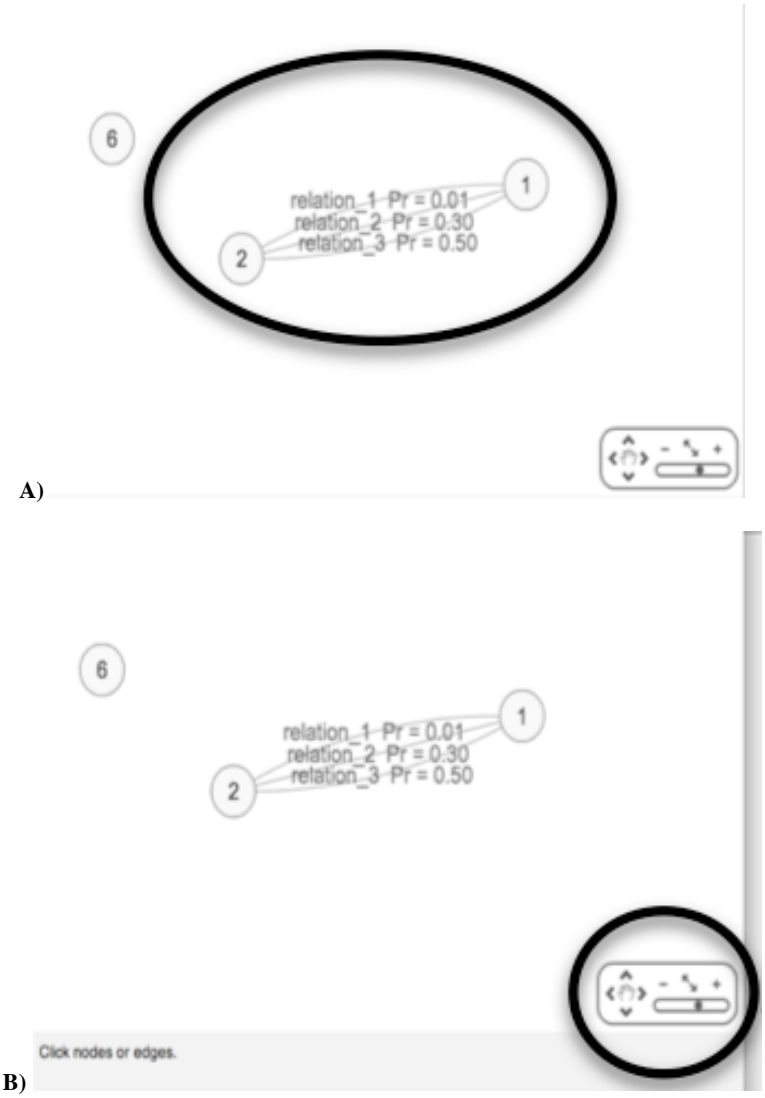

Figure 5. The node-link graph view shows the entities and relations at a finer grain of resolution than does the adjacency matrix view. A) The identity of the relations are present as labels on the individual links, and the degree of confidence the algorithm has that the relations are true is given by a quantitative estimate represented by a floating point value. Relations shown here are bidirectional. B) The panel on the bottom right can be used to pan and zoom into the graph.

two derived measures comes from trying to find some way or piece of information that the algorithm designer is interested in discovering, to guide their search of the solution space for the relational data, whose abstraction could be a massive graph. The user can toggle between views of the two derived attributes by selecting the whitespace to the right of the adjacency matrix representation. This interaction procedure will be improved in future iterations of the system. The adjacency matrix representation of the graph serves as a kind of "enriched overview" – at once compactly displaying the contents of the entire graph, and providing information relevant to the nature and number of relations to guide search of the graph to regions of potential interest. Upon finding a region of interest, the user can select this patch in the adjacency matrix view to display the corresponding node-link graph representation of this region of the graph: See Figure 3. The node-link graph representation of the data contains finer-grained details of the relationships between two entities, including the exact number of relations, the identity of the relations, the directedness of the relations, and a quantitative measure of the algorithm's certainty that the relation is correct given as a floating-point number. There are also controls for panning and zooming. See Figure 5.

The left panel of this tool is implemented using the Cytoscape Web tool [9]. The adjacency matrix panel of this tool is implemented using Protovis [4].

4 Strengths and Limitations of the Work

RelaViz represents a first attempt at creating a visualization tool for assessing relational learning algorithms. Its design is tailored to helping relational learning algorithm designers assess whether or not the relations learned between entities "make" sense, by allowing the designer to visually inspect those entities,

and assess progression of the learning process by monitoring the number of relations represented in the adjacency matrix.

Unfortunately, the system has only be used to present synthetic relational learning algorithm data, whose structure is based on the algorithm output of the system presented by Bordes et al [3]. It is therefore difficult to validate how well this tool can truly help assess relational learning algorithm performance. For instance, the assumption that the derived, degree of certainty attribute representation used in the adjacency matrix view to guide graph navigation will lead to the algorithm designer discovering regions of interest in assessing whether the algorithm is discovering relations that "make sense," is difficult to validate; the claim still remains hypothetical. I would argue, however, that marking regions of the graph based on certainty can help the designer determine whether the relations the algorithm is certain of are true, and the relations the algorithm has low certainty of are probably not true, by visually inspecting them in a targeted way that saves the user from sifting through an entire massive graph; this is better than no guidance at all.

Finally, given that real relational data was not used in this project, it will be interesting to see how the level of sparsity of the graph could impact this design. It could turn out that it is robust to the problems associated with viewing dense graphs, since we are only observing localized patches of the graph at a time. Depending on the size of the graph, however, it will also be interesting to see how well the interactive adjacency matrix view scales with the number of entities – specific regions may be difficult to select if they are small, and this may necessitate some form of sorting based on the derived attribute type by low to high, or some form of aggregation if possible.

Unfortunately at the onset of the project, much work was invested in implementing animated link-splitting within the Gephi framework in an attempt to reduce link information clutter, and show link information on demand [2]. This approach was inadequate, however, since it amounted to simply allowing for unguided navigation of a potentially massive graph. The current solution is more thoughtful in its approach to directing a designer's attention around a potentially massive dataset representation. It also has the added benefit of allowing designers view algorithm learning progression by the degree to which the adjacency matrix is filled in, when it is set to the number of links mode.

5 Future Work

There is much room for future work in improving the existing state of the RelaViz system. First, the colour encoding of quantity used by the two derived attributes representing algorithm certainty and relation quantity could be improved: namely, saturation level of a single hue could be used to express the level of certainty or quantity of relations, depending on what is being viewed. Second, a legend could be placed to the right of the adjacency matrix to make explicit that less saturation encodes low, and more saturation encodes high. Third, a visible toggle control should be placed above the adjacency matrix making it clear there are two ways to view it. Fourth, the node-link representation needs to be updated to encode certainty that a relation is correct by band thickness of the links: thick bands for a high degree of certainty, and thin bands for a low degree of certainty. Fifth, it could be there is some structural information present in the graph that might be interesting to explore. It may be that this structure is not captured by the zoomed-in view of the node-link graph currently present in the user interface. Although there are controls within this view to zoom out and pan, including a third view which is a much more zoomed-out version of the graph around this neighborhood may ensure that no interesting structural phenomena, such as hierarchical relationships present in the relational data, go unnoticed. Most importantly, real relational learning algorithm output data needs to be acquired to validate whether or not this linked adjacency matrix and node-link graph view can effectively help designers assess the performance of their algorithm. It may also be interesting, however, to see if this tool can be applied to biological data, where networks of interactions exist between components that are often encoded as relations [9]. As well, multiple relations often exist between biological components, and this may be effectively visualized by observing the node-link view with split links [9]. In order to make this application, I anticipate that the derived attribute measure that was tailored for the relational learning algorithm domain will need to be tailored to a particular biological dataset, to guide navigation of a large network of interactions.

6 Conclusion

This work represents a first attempt at creating a tool for the assessment of relational learning algorithm behaviour and performance through a linked graph and adjacency matrix visualization. The adjacency matrix view attempts to allow for targeted navigation of the relational learning data graph based on two measures the algorithm designer is concerned about: the number of relations or presence or absence of relations between entities, and the degree of certainty the algorithm has that relations are correct. The tool attempts to make up for the fact that the adjacency matrix representation, by its nature, is limited in its capability for displaying several relations between entities and their identities by including the node-link graph view. The node-link graph view allows for viewing the exact data encoded by the relational data triplets: identities of the relations between entities, the directed-ness of these relations, and the actual degree of certainty the algorithm has that a specific relation is correct. The idea was to leverage the strengths of these two representations for the task at hand, while allowing each to cover up for the other's weakness in expressiveness. Although there is still much work to be done in validating this tool, its approach to targeted navigation of relational learning data is an exciting first attempt in a domain where automated validation of algorithm behaviour on new data is currently not possible, and guidance in determining parameters governing learning termination conditions is absent.

References

[1] J. Abello, F. van Ham and N. Krishnan. ASK-GraphView: a largescale graph visualization system, *IEEE Transactions on Visualization and Computer Graphics 12,* pp. 669-676, 2006.

[2] M. Bastian, S. Heymann and M. Jacomy. Gephi: an open source software for exploring and manipulating networks. *International AAAI Conference on Weblogs and Social Media*. 2009.

[3] A. Bordes, J. Weston, R. Collobert and Y. Bengio. Learning Structured Embeddings of Knowledge Bases. In *Proceedings of the 25th Conference on Artificial Intelligence (AAAI-11),* San Francisco, USA, 2011.

[4] M. Bostock and J. Heer. Protovis: a graphical toolkit for visualization. In *IEEE Transactions on Visualization and Computer Graphics 15,* pp. 1121-1128, 2009.

[5] N. Henry, J.D. Fekete. MatrixExplorer: a dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics 12*, pp. 677-684, 2006.

[6] I. Herman, M. Melancon and S. Marshall. Graph visualization in information visualization: a survey. In *IEEE Transactions on Visualization and Computer Graphics* 6, pp. 24-44, 2000.

[7] M. Meyer, T. Munzner and H. Pfister. MizBee: a multiscale synteny browser. In *IEEE Transactions on Visualization and Computer Graphics 15,* pp. 897-904, 2009.

[8] T. Munzner, F. Guimbretiere and G. Robertson. Constellation: a visualization tool for linguistic queries from MindNet. In *Proceedings of the Information Visualizaiton 1999 IEEE Symposium on Information Visualization*. Pp. 132-135, 1999.

[9] M. Smoot, K. Ono, J. Ruscheinski, P. Wang and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics,* 1; 27(3): pp 431-432, 2010.

[10] I. Sutskever, R. Salakhutdinov, and B. Tenenbaum. Modelling relational data using Bayesian clustered tensor factorization. In *Advances in Neural Information Processing Sytems (NIPS)*, 22:1821–1828, 2009.

[11] F. van Ham and A. Perer. Search, show context, expand on demand": supporting large graph exploration with degree-of-interest. In *IEEE Transactions in Visualization and Computer Graphics15*, pp. 953-960, 2009.