Visualizing the Uncertainties of Probabilistic Soft Logic With a Graphical Model

Jason Ting
University of California, Santa Cruz
1156 High St
Santa Cruz, CA 95064
jting2@ucsc.edu

***Abstract*— Probabilistic Soft Logic is a machine learning framework used in the Statistical Relational Learning, or SRL, community that generates a probabilistic model to find soft truth. Unlike traditional machine learning with IID methods, Probabilistic Soft Logic breaks the IID assumption by utilizes collective classification to make an inference based on the model. Furthermore, the difference between this probabilistic model and other probabilistic models like Markov Logic Network, Probabilistic Soft Logic uses soft truth value to make it's inference.**

The objective of this paper is to construct a graphical model to visualize the uncertainties and how it impacts the model as a whole. This will be done through an implementation of a multimodal graph with many different nodes and edges that each have different characteristics.

I. Introduction

A. Motivation

The technological advancement starting the beginning of 21st century allowed us to bring machine learning and artificial intelligence to the next level. With higher computational efficiency, we were able to effectively train models to make inference and even train robots to play games like Chess, Go, League of Legends, etc.

Training these models generally requires large amount of data. Most of the time, there exists some excessive data that doesn't help the model in any way. People usually visualize these data through charts, graphs, tables, etc. to identify these unwanted variables. Even though there has been many different methods to visualize data, there hasn't been many methods to visualize different types machine learning algorithms.

The main aspect of Probabilistic Soft Logic is it follows a set of logical rules written to infer with some probability of an event occurring. Below is an example of a logical rule:

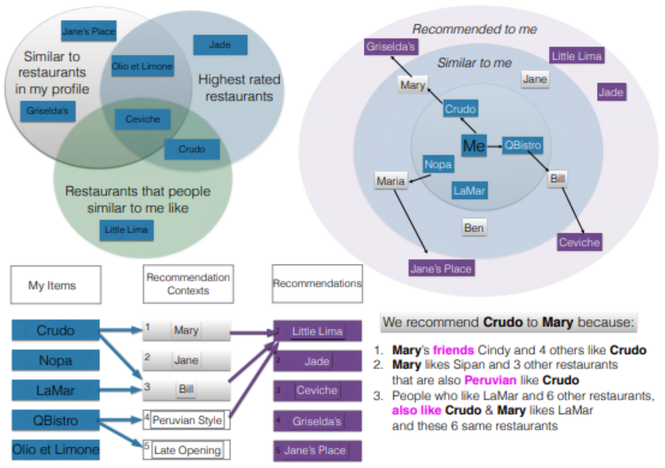

Jane's Place
Similar to restaurants in my profile
Olio et Limone
Jade
Highest rated restaurants
Griselda's
Ceviche
Crudo
Restaurants that people similar to me like
Little Lima
Recommended to me
Griselda's
Similar to me
Little Lima
Mary
Jane
Jade
Crudo
Me
QBistro
Nopa
Maria
LaMar
Bill
Ben
Ceviche
Jane's Place
My Items
Recommendation Contexts
Recommendations
Crudo
Nopa
LaMar
QBistro
Olio et Limone
1 Mary
2 Jane
3 Bill
4 Peruvian Style
5 Late Opening
1 Little Lima
2 Jade
3 Ceviche
4 Griselda's
5 Jane's Place
We recommend Crudo to Mary because:
1. Mary's friends Cindy and 4 others like Crudo
2. Mary likes Sipan and 3 other restaurants that are also Peruvian like Crudo
3. People who like LaMar and 6 other restaurants, also like Crudo & Mary likes LaMar and these 6 same restaurants

Fig. 1. Different types of visualization [1]

1 : Likes (User , ItemA) && Similar (ItemA , ItemB) -> Likes (User , ItemB)

What this specific logical rule we say since a user likes item A with some probability and item A is similar to item B, we can then infer with some probability that the user will like item B. As you can see on the left, this logical rule has an initial weight of 1. Generally, there will be multiple logical rules that come into play when deciding whether a user likes item B.

The motivation behind this paper is to be able to visualize a Probabilistic Soft Logic model in order to visualize how the algorithm works. Not only will it take visualization to the next step, it also helps understand the underlying mechanism behind Probabilistic Soft Logic. Being able to visualize these mechanism can help users see how the uncertainties in each rules in Probabilistic Soft Logic affect the model's decision making. Furthermore, visualizing the impact of the weights can help understand the importance of it corresponding to the logical rules. Another goal of this paper is to visualize the satisfaction of each rules, or how confident each logical rules are with their inference. This lets us know the importance certain logical rules and the role they play in the inference process.

The biggest challenge for this scientific visualization is visualizing the graphical model in an aesthetic way while maintaining the comprehensibility. Visualizing a machine learning model in general has always been fairly difficult as most model deals with matricies of features like neural networks. Having a visual representation of a machine learning model can greatly help enthusiasts understand how the algorithm operates instead of treating it as a black box. Furthermore, this visualization will be able to help users using Probabilistic Soft Logic identify data errors and understand which ground rules will be more beneficial.

B. Data

For this project, we will be using a synthetic dataset by Lise Getoor and her team at University of California, Santa Cruz to create this graphical model. In this dataset, we want to the acquaintance links between different people. We are given information on where each people have lived, what activities they like or dislike, and some known acquaintance link. Our goal is to visualize the probability of the unknown acquaintance links. In this case, we will have some known acquaintence data and we want to infer on the unknown acquaintance data.

In our model, each of our nodes will contain different people and an attribute pertaining to that person. Then we will have an edge what shows the inferred probability of each acquaintance link.

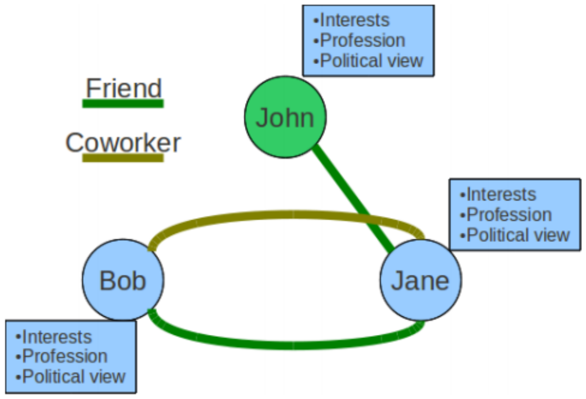
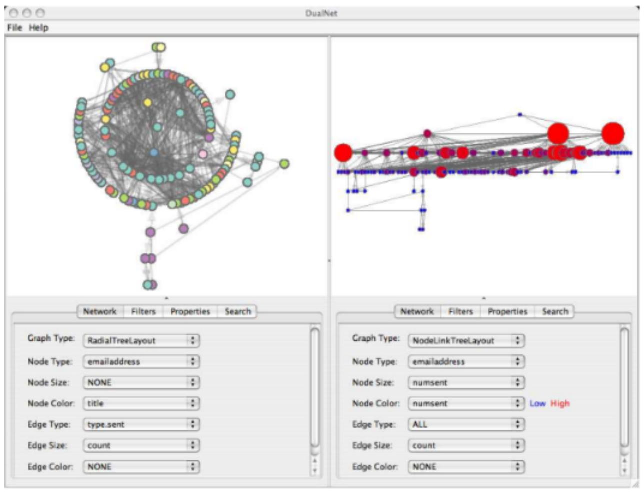


Fig. 2. Multi modal graph [2]



Graph Type: RadialTreeLayout
Node Type: emailaddress
Node Size: NONE
Node Color: title
Edge Type: type.sent
Edge Size: count
Edge Color: NONE

Graph Type: NodeLinkTreeLayout
Node Type: emailaddress
Node Size: numsent
Node Color: numsent
Edge Type: ALL
Edge Size: count
Edge Color: NONE

Fig. 3. Dualnet [3]

II. Related Work

Even though there hasn't been any studies on visualizing a graphical model for Probabilistic Soft Logic, there has been some previous studies on user preferences for explanations on a hybrid recommendation system that utilizes Probabilistic Soft Logic through a variety of text, visual, and graph-based formats [1]. An example would be figure 1 where the author used concentric circles, Venn diagrams, and pathways between columns to describe a logical rule.

In addition, there has also been previous studies on visualizing uncertainty in graphs[2]. In that paper, it talks about different techniques and prototype tools to visualize multimodal graphs. To extend on both of the ideas above, we focus on combining both of those ideas above to create a tool that views the uncertainties of a graphical model for Probabilistic Soft Logic. Just like figure 2, my proposed graphical model will have different nodes and edges except in a way larger scale. Furthermore, it will show more information including the weight of each logical rules, inference value, satisfaction of each rules, and the impact of an inference from each rules.

Namata et al. designed a network visualization tool called Dualnet shown in figure 3 to visualize a multiple coordinated view of a given network. They implemented it using multiple different views that is also interactive as seen above.

III. Background

A. Probabilistic Soft Logic

Probabilistic Soft Logic is an open source machine learning framework that uses logical representation to define large graphical model. Because it is a templating language for Markov Random Field which allows Probabilistic Soft Logic to provide a fast and scalable inference. What makes Probabilistic Soft Logic different from other statistical relational learning models like Markov Logic Network is that it instead of having hard truth values, the ground atoms all have soft truth values between [0,1]. Given a set of weighted logical rules, Probabilistic Soft Logic builds a graphical model defining a probability distribution over the continuous space of values of the random variables in the model. Furthermore, Probabilistic Soft Logic shapes the model into a convex optimization problem wheras Markov Logic Network is an NP-hard problem.

The rules are defined as below:

$$\forall u1, u2, i. \quad w \quad : \quad SimilarUsers(u1, u2) \wedge Likes(u1, i) \longrightarrow \quad Likes(u2, i)$$

$$\forall i1, i2, u1. \quad w \quad : \quad SimilarItems(i1, i2) \wedge Likes(u1, i1) \longrightarrow Likes(u1, i2)$$

In this notation, w will represent the weights, $u1, u2, i, i1, i2$ will all represent random variables. The two rules above are defined as logical rules and the goal is to use some available data to create many different ground rules based on logical rules. A ground rule is a logical rule where the random variables are replaced by constants. For example, SimilarUsers('Annie', 'Bob') would be a a ground rule where 'Annie' and 'Bob' are both constants.

As mentioned above, all ground atoms have soft truth values between [0,1] and in order to compute the soft truth value for each logical rule, one must first transform the logical rules into boolean logic.

Because the logical rules are in the form of A implies B $\big(A \longrightarrow B\big)$, the boolean logic would be $\neg A \vee B$.

The truth table is shown below:

A	B	$\neg A \vee B$
F	F	T
F	T	T
T	F	F
T	T	T

'A' would be the lefthand side of the rule and 'B' would be the righthand side, but once you convert it to boolean logic, the left and righthand side of the rule will disappear.

Now that you have transformed the ground rules logic boolean logic, you can check whether the rules are satisfied or find the distance to satisfaction d of each individual ground rules by using the Lukasiewicz relaxation formula below:

$$r_1 \wedge r_2 = max\{r_1 + r_2 - 1, 0\}$$

$$r_1 \vee r_2 = min\{r_1 + r_2, 1\}$$

$$\neg r = 1 - r$$

where r represents the value of each ground atoms.

Now we can transform this into a MAX SAT problem with this Lukasiewicz relaxation in the formula below:

$$argmax_{r \in [0,1]^n} \sum_{C_j \in C} w_j min(\sum_{i \in I_j^+} r_i + \sum_{i \in I_j^-} (1 - r_i, 1))$$

where C represents the logical knowledge or rules that gets interpreted by using Lukasiewciz logic, w_j represents the weight of each logical knowledge. Let I_j^+ in $i \in I_j^+$ within the summation correspond to the non-negated atoms in the ground rules, likewise, let I_j^- in $i \in I_j^-$ within the summation correspond to the negated atoms in the ground rules.

Because of the formula above, this allows Probabilistic Soft Logic to be a convex optimization problem which is fast and scalable. Probabilistic Soft Logic will jointly infer the best satisfaction value for all the target atoms to maximize the equation above.

Just like many other machine learning algorithm, Probabilistic Soft Logic also includes a weight learning component. The nice thing about the weight learning component is there are many different types of weight learning method users can invoke including different types of voted perceptron algorithm including MaxLikelihood MPE, Expectation Maximization, MaxPseudoLikelihood, etc. In addition to voted perceptron algorithms, Probabilistic Soft Logica can also invoke different types of grid search algorithm including Random Grid Search, Rank Search, Continous Random Grid Search, etc.

IV. Method

A. Forced Directed Graph

Forced directed graph is an algorithm that assist in drawing graphs in an aesthetically pleasing way. Instead of x and y coordinates, forced directed graph utilizes information from the data to create its graph. To do this, the algorithm simulates the the graph for a certain number of iterations before actually plotting the graph. In addition, the algorithm uses a spring-like attractive force based on Hooke's law to pull adjacent nodes closer to each other while utilizing repulsive force based on Coulomb's law to push all irrelevant nodes away.

Forced directed graph algorithm has many advantages including simplicity, flexibility, and most importantly interactivity. This helps with the visualization of a Probabilistic Soft Logic model because of the amount of nodes and edges it contains. Interactivity can help users choose certain nodes to view and examine some details.

To implement this algorithm, I used a javascript library called d3. D3 is a scientific visualization library that implements forced directed graph as well as various other implementation.

B. Data Preprocessing

In order to visualize any sort of graphical model, we first need to obtain the data. The data used to visualize the graphical model in this paper is the simple acquaintance data. The data contains information on where people lived, who they know, and what they like. The goal of this model is to infer who everyone knows. In order to obtain the ground rules and its satisfaction, after evaluation, I output those data onto a txt file.

After obtaining the data from the Probabilistic Soft Logic algorithm, there was a lot of data preprocessing involved. Generally, the form that the data needs to be in for a forced directed graph is shown below:

'nodes' : [
{ "groundAtom": "LIKES('Steve', 'Hiking')", "group": 2, "type": "closed" },
{ "groundAtom": "LIKES('Dhanya', 'Hiking')", "group": 2, "type": "closed" },
{ "groundAtom": "LIKES('Ben', 'Hiking')", "group": 2, "type": "closed" }
]

'links' : [
{ "source": "KNOWS('Ben', 'Alex')", "target": "KNOWS('Elena', 'Alex')", "rule": "¬(KNOWS('Elena', 'Alex')) ∨ ¬(KNOWS('Ben', 'Elena')) ∨ KNOWS('Ben', 'Alex')", "satisfaction": 0.985 },
{ "source": "KNOWS('Ben', 'Steve')", "target": "KNOWS('Ben', 'Elena')", "rule": "¬(KNOWS('Elena', 'Steve')) ∨ ¬(KNOWS('Ben', 'Elena')) ∨ KNOWS('Ben', 'Steve')", "satisfaction": 0.996 },
{ "source": "KNOWS('Ben', 'Alex')", "target": "KNOWS('Ben', 'Dhanya')", "rule": "¬(KNOWS('Dhanya', 'Alex')) ∨ ¬(KNOWS('Ben', 'Dhanya')) ∨ KNOWS('Ben', 'Alex')", "satisfaction": 1 }
]

The nodes data is used to represent the nodes in the graphical model. Each node will have a unique id, or groundAtom, which will be the node's identifier. The group of the nodes will determine which predicate it belongs to. The type will allow the user to know whether it is open or closed predicate.

As for the link data, it is used to show correlation between the nodes. It basically inform the users which nodes are affecting each other. The link data will be fairly similar to node data except it will contain the source, target, rule, and satisfaction. The source and target of each link correspond to the source and destination node respectively. The rule and satisfaction parameter will correspond to the rule of the source and destination nodes and the satisfaction of that entire rule.

V. Results

Figure 4 shows the overall graphical representation of a Probabilistic Soft Logic model. The nodes being displayed above represents individual ground atoms of the model. The color of the nodes in the Probabilistic Soft Logic each represents different predicate. However, only open predicate(s) will have the name of the ground atoms displayed. There are two reasons for this. First off, it is to prevent over crowding of the graphical model. Secondly, since we are making inference on the open predicate(s), the user will be able to identify nodes that are open predicate(s).

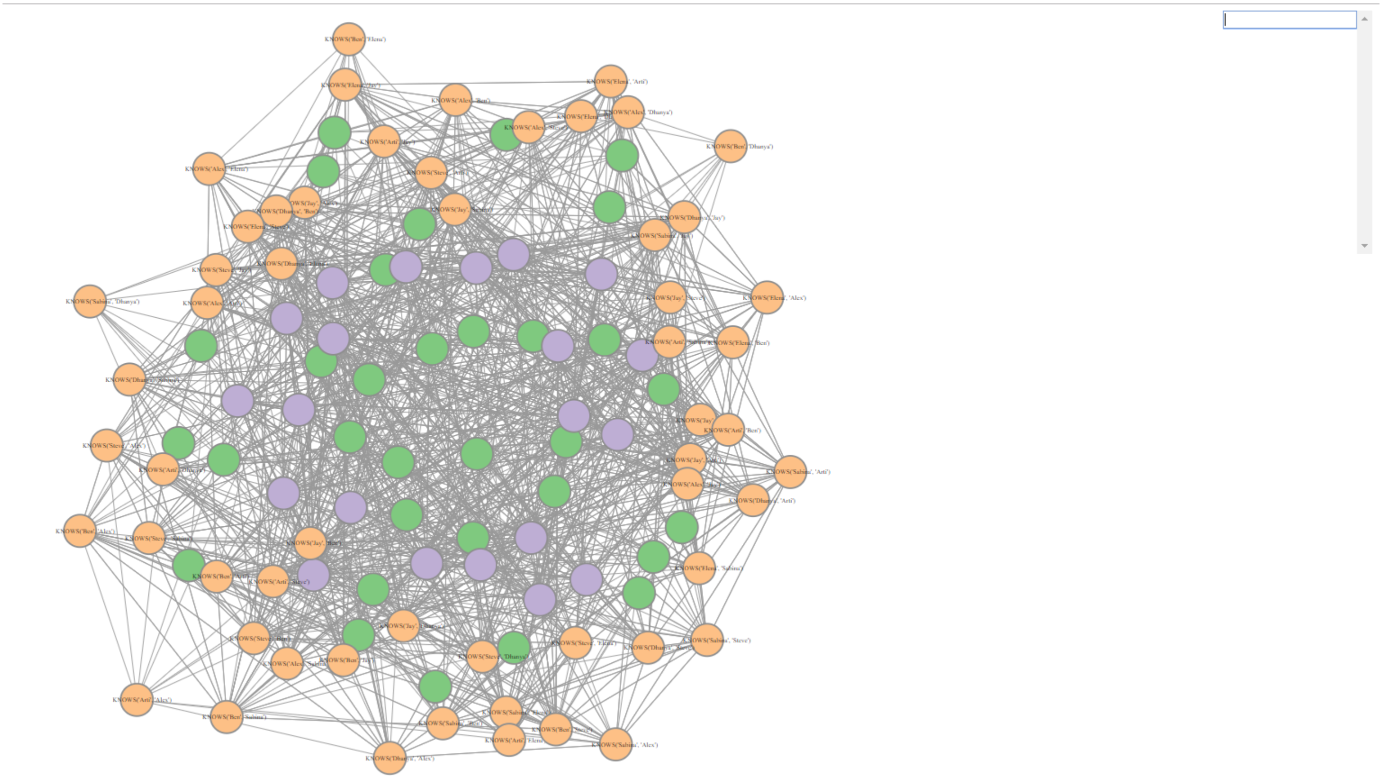


Fig. 4. Simple Acquaintance Graphical Model With Search Bar

The edges of each ground atoms above can represent two different possibilities:

- Link two unobserved atoms within each corresponding logical rules together
- Link an unobserved atom with an observed atom of the corresponding logical rules

Technically speaking, Probabilistic Soft Logic is an undirected graphical model where all the nodes will create many cliques, however for the sake of visualization, a link between two observed atoms is not really useful because the value of the unobserved atoms that you are inferring is the important part.

Another feature of this visualization is a user can select a node and all the unrelated links with the selected node will disappear and the opacity of unrelated nodes will also decrease as shown in figure 5. Furthermore, the names of all related ground atoms to the selected node will appear. This shows you a better visualization of which ground atoms are affecting the selected node instead of visualizing one giant graph.

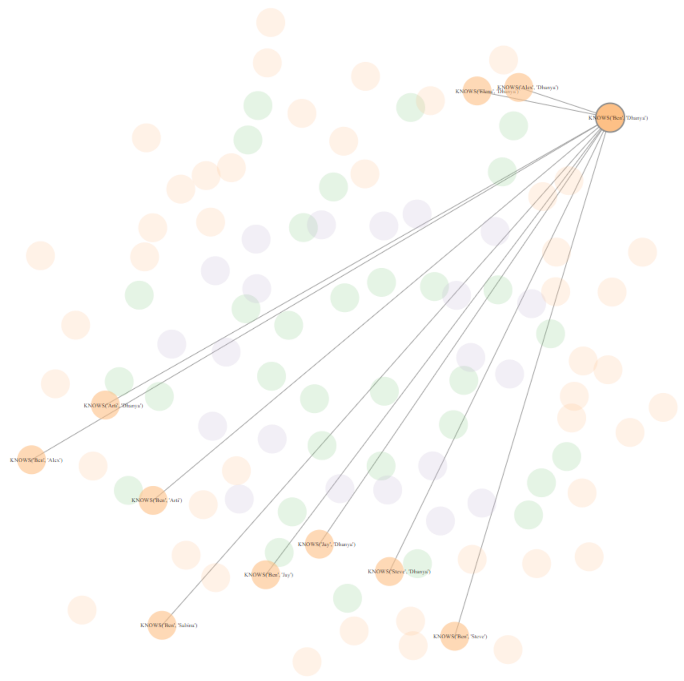
In addition, selecting the nodes will also trigger a side bar menu as shown in figure 6. The side bar will allow the user to view all the rules corresponding to the selected node and the satisfaction of each rule.

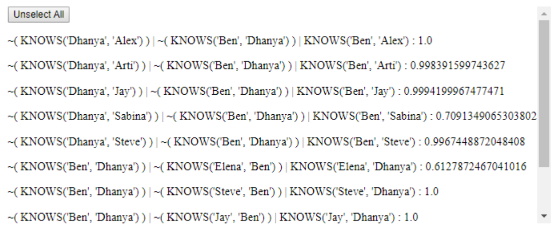
Being able to view all the rules corresponding to the selected node can help users spot errors in the data. For example, if a user was expecting certain ground rule(s), he or she can examine the side bar to ensure that the ground rule(s) is actually there. Furthermore, users can also catch bizarre rules that shouldn't exist. Besides the ground rules, being able to see the satisfaction of each ground rules will show the users which ground rule isn't as satisfied as others.

The color of the nodes in the Probabilistic Soft Logic each represents different predicate. This color scheme also help users visually cluster predicates together. In addition, color scheming help users further identify the open predicate. If the colors weren't different, it will be difficult to identify open predicate nodes by having it display the ground atoms.

Another important feature from this scientific visualization is it has the capability of displaying the rules each link creates. Although it is not as relevant currently, however once the link is color schemed with red being the connection is negatively affecting the satisfaction of the rule and green being the connection is positively affecting the satisfaction of the rule, users will be able to identify the rule of the connection and how that specific connection affects the satisfaction.

In certain cases, there will exist much more nodes





```
~( KNOWS('Dhanya', 'Alex') ) | ~( KNOWS('Ben', 'Dhanya') ) | KNOWS('Ben', 'Alex') : 1.0
~( KNOWS('Dhanya', 'Arti') ) | ~( KNOWS('Ben', 'Dhanya') ) | KNOWS('Ben', 'Arti') : 0.998391599743627
~( KNOWS('Dhanya', 'Jay') ) | ~( KNOWS('Ben', 'Dhanya') ) | KNOWS('Ben', 'Jay') : 0.9994199967477471
~( KNOWS('Dhanya', 'Sabina') ) | ~( KNOWS('Ben', 'Dhanya') ) | KNOWS('Ben', 'Sabina') : 0.7091349065303802
~( KNOWS('Dhanya', 'Steve') ) | ~( KNOWS('Ben', 'Dhanya') ) | KNOWS('Ben', 'Steve') : 0.9967448872048408
~( KNOWS('Ben', 'Dhanya') ) | ~( KNOWS('Elena', 'Ben') ) | KNOWS('Elena', 'Dhanya') : 0.6127872467041016
~( KNOWS('Ben', 'Dhanya') ) | ~( KNOWS('Steve', 'Ben') ) | KNOWS('Steve', 'Dhanya') : 1.0
~( KNOWS('Ben', 'Dhanya') ) | ~( KNOWS('Jay', 'Ben') ) | KNOWS('Jay', 'Dhanya') : 1.0
```

Fig. 5. Node Selection

than the current example. It will be a difficult task for a user to find a specific ground atom. A way to tackle this problem is implementing the search bar on the top right as shown in figure 4. You can select which ground atom you want to vie by simply typing in the ground atom in the search box. This will alleviate the problem of looking at every node manually to find the specific ground atom.

VI. Proposed Research Directions

The current graphical model design is just a prototype. Given the time and opportunity, I want to be able to compare two different Probabilistic Soft Logic models and visualize the performance difference between the two. This can help us compare the pros and cons of both model hopefully be able to learn from both models. In addition, I want to be able to color code the links of each node to represent whether the that link positively or negatively affected the open node's value. With this, we can improve on the two models to hopefully create a better model. In addition, visualization is very subjective and therefore this project will always have room for improvement or enhancement.

The current search bar does not give suggestions as you type in the ground atom. Adding this feature will allow users to find the ground atoms he or she is searching for much faster than manually looking at the graph or typing everything in perfectly.

Another important implementation would be to create a general purpose code that can preprocess all the data when you run the training script for Probabilistic Soft Logic. Afterward, just open up the html file and the visualization would be there.

VII. Conclusion

Overall, visualizing machine learning algorithm in general can be quite tricky. Being able to visualize the graphical model for Probabilistic Soft Logic can be beneficial for the user utilizing the machine learning model. It helps the user obtain a better understanding visually of what the model actually looks like. In addition, it allows the user to double check for data error and see the satisfaction of each rule.

References

[1] User Preferences for Hybrid Explanations. (2019). [online] Available at: https://linqs.soe.ucsc.edu/sites/default/files/papers/kouki-recsys17.pdf [Accessed 1 Feb. 2019].

[2] Visualizing Uncertainty in Graphs. (2019). [online] Available at: https://users.soe.ucsc.edu/ pang/261/s10/projects/projects/ncesario/proj/paper.pdf [Accessed 1 Feb. 2019].

[3] A Dual-View Approach to Interactive Network Visualization. (2019). [online] Available at: https://www.cs.umd.edu/ namatag/dualnet/cikm2007-Full.pdf [Accessed 1 Feb. 2019].

[4] Hinge-Loss Markov Random Fields and Probabilistic Soft Logic. (2016). [online] Available at: https://linqs.soe.ucsc.edu/sites/default/files/papers/bach-jmlr17.pdf [Accessed 1 Feb. 2019].

[5] Simple Acquaintence data. (2016). [online] Available at: https://github.com/linqs/psl-examples/tree/master/simple-acquaintances/data [Accessed 1 Feb. 2019].