Worksheet #4
May 7th, 2020

Paper: Relational Neural Networks

- General questions

1. What is this paper about? Could you summarise its contribution in a paragraph?

   This paper proposes a method based on random walks to build a neural network structure from relational data.

2. How does this work differ from others mentioned in the paper?

   As clearly stated in the abstract: "We distinguish ourselves from current approaches that rely on expert hand-coded rules by learning relational random-walk-based features to capture local structural interactions and the resulting network architecture"

3. Do the authors present experiments? What is the methodology used? Does it sound correct? Why?

   Yes. They do present experiments. They applied their method to well-known relational datasets found in the liteature (unfortunately, very few available :() and compared with other relational methods, including Markov Logic Networks (MLNs). One result in Table 4 (perhaps, the most relevant of all, because this would show the actual benefit of the method) looks weird: for UW-CSE, AUC-PR is much higher than AUC-ROC, which contradicts all other results of all tables when comparing these numbers, and which contradicts some facts in the literature that state that ROCs are optimistic when compared with Precision-Recall curves, for imbalanced data – pls, see excellent paper that compares ROC and PR curves: `https://dl.acm.org/doi/10.1145/1143844.1143874`.

4. What are the main results/findings/conclusions? Are the results useful/relevant? Why?

   Despite having produced the result mentioned above, the method is quite interesting and promising as a way of capturing relevant relations from data. Moreover, the other results look very good.

- Technical questions

  1. Regarding the sentence: "While expressive, these models (note: structured) do not incorporate or discover latent relationships between features as effectively as deep networks.", and having played with the Aleph system, do you think that there is any limitation in learning latent relationships (for example, in order to learn the relation `grandparent` the system may need to learn the relation `parent`) in systems such as Aleph?

     My personal opinion: I believe that current systems that learn from relational data (ILP-based, graph-based or neural networks-based) still lack a component to learn "latent relationships" with the exception of works by Cropper and Muggleton, which do not yet solve the problem, but investigate "predicate invention" and "abduction" (an example: `http://andrewcropper.com/pubs/scai15-incomplete.pdf`) as a way of filling missing knowledge in data.

     Take as an example the task of learning "grandparent" based in "father" and "mother". The ideal thing would be that the system generates – internally (no need to show these to the user):

```
gp(X,Y) :- mother(X,Y), mother(X,Y).
gp(X,Y) :- mother(X,Y), father(X,Y).
gp(X,Y) :- father(X,Y), father(X,Y).
gp(X,Y) :- father(X,Y), mother(X,Y).
```

     and, internally, executes an algorithm that factorizes that basic knowledge in a hierarchy:

```
p(X,Y) :- mother(X,Y).
p(X,Y) :- father(X,Y).
```

     In other words, mother appears twice with the same pattern, so let's create a new predicate for that. Likewise, father appears twice with the same pattern. When we do that, a new knowledge emerges that captures the latent relationship p(X,Y) and a hierarchy that is hidden in the data:

```
gp(X,Y) :- p(X,Z), p(Z,Y).
```

```
p(X,Y) :- mother(X,Y).
p(X,Y) :- father(X,Y).
```

Aleph can do that, but we need to give examples for parent **and** grandparent. The ideal thing to do would be to give examples only to the concept we want to learn: grandparent, and the system be able to create the hierarchy automatically, which is far from trivial :(. I am waiting for ideas ;)

2. How does *Relational Random Walks* (RRW) work and how does it compare with Inductive Logic Programming (ILP)? Does RRW actually learn latent relationships?

   RRW uses an algorithm that is similar to ILP systems, but follows random paths instead of constructing a lattice based on the saturated clause. I don't believe RRW learns latent relationships due to reason above.

3. How does the proposed method allow for a reduction in the number of network weight parameters? And why is this reduction important?

   The proposed method ensures parameter tying of multiple ground instances of the rule. This helps reducing the number of network weight parameters, which will accelerate the neural network learning phase.

4. What are Tensor Based Models?

   Roughly speaking, a tensor is a matrix with more than 2 dimensions. This representation is quite useful to model relationships among multiple multidimensional variables.

5. The examples reported in this paper are all about binary relations. Could you think of a way of extending this to handle n-ary relations?

   It is possible, but at the cost of a very high computational cost. Once again, I am waiting for solutions ;)

6. In the evaluation, the authors use AUC-ROC and AUC-PR to evaluate the models. Why are they using these evaluation metrics? (you may need to complement your reading with this).

   Comment about the results, discussed above, give a hint on the answer to this question. ROC curves plot sensitivity (Recall or True

Positive Rate) against False Positive Rate (FPR). If we have a skew in the class variable (taking a binary variable, we would have a much higher proportion of one value against the other), an error in the smaller class value should weigh more than an error in the majority class value. However, ROCs will hide this imbalance. On the other hand, PR curves plot Precision (TP/(TP+FP)) against Recall, which gives a more realistic idea (not so optimistic as the ROC) of the performance of the model. More details about the differences are in the proposed paper above.