Von: **Luis Galárraga** shamantobi@gmail.com
Betreff: Fwd: VLDB: Your manuscript entitled Fast Rule Mining in Ontological Knowledge Bases with AMIE+
Datum: 17. Februar 2015 09:57
An: Fabian Suchanek f.m.suchanek@gmail.com, Katja Hose khose@cs.aau.dk, Christina Teflioudi chteflio@yahoo.gr

LG

---------- Forwarded message ----------
From: **Sihem Amer-Yahia** <em@editorialmanager.com>
Date: 2015-02-17 9:45 GMT+01:00
Subject: VLDB: Your manuscript entitled Fast Rule Mining in Ontological Knowledge Bases with AMIE+
To: Luis Galárraga <galarrag@enst.fr>

Ref.: Ms. No. VLDB-D-14-00104
Fast Rule Mining in Ontological Knowledge Bases with AMIE+
The VLDB Journal

Dear Mr. Luis Galárraga,

Reviewers have now commented on your paper. You will see that they are advising that you revise your manuscript. If you are prepared to undertake the work required, I would be pleased to reconsider the revised manuscript.

The reviewers' comments can be found at the end of this email or can be accessed by following the provided link.

This is your login information:
Your username is: lgalarraga
Your password is: galárraga3378

When revising your work, please submit a list of changes or a rebuttal against each point which is being raised when you submit the revised manuscript.

Your revision is due by 18 Apr 2015.

Please make sure to submit your editable source files (i. e. Word, TeX).

To submit a revision, go to http://vldb.edmgr.com/ and log in as an Author. You will see a menu item called 'Submissions Needing Revision'. You will find your submission record there.

Yours sincerely

Sihem Amer-Yahia
Editor-in-Chief
The VLDB Journal

Reviewers' comments:

The paper proposes several optimizations to AMIE. All reviewers agree that the extensions do not meet the bar for VLDBJ publication. The reviewers are recommending very extensive major revisions including the following.
Better justification of (and formal description of) the optimizations.
Thorough evaluation of the benefits/limitations of each optimization.
Scalability results for each optimization.
Separate training and testing data in AMIE+ and in other methods, including more recent learning methods than used in the WWW paper (scalability and quality).
Clarification of the usefulness of mined rules given their low precision (particularly Rev 1 and Rev 3's comments).
Better presentation of the theory and implementation (so that the algorithms are clear and the results reproducible).
Additional important revision suggestions are included in the detailed reviews.

Reviewer #1: This manuscript presents an extended version of a previously published paper at WWW conference. In that paper authors presented a novel method for mining rules from knowledge bases called AMIE. The proposed extension basically optimizes AMIE with a series of heuristics to improve its performance. From my point of view, this extension does not contribute enough to justify another publication at a high-impact journal. Proposed heuristics seem mainly derived from the expertise of authors in the evaluated datasets. They do not rely on a well-grounded basis that justifies their introduction. For example, the main proposed heuristic (which contributes most to the speed-up) is an ad-hoc approximation of the PCA confidence for a specific type of 3-atom rules, being not trivial to generalize it to any kind of rule. (Indeed, experiments are limited to 3-atom rules).

However, my main concern with this paper is the very poor precision of the predicted statements produced by the mined rules (35%-45%). I presume that these precision scores are quite far from those that could be obtained with other data mining techniques. For example, statistical classifiers have been successfully applied to RDF data to infer new statements with high precision scores. So, the question is: what are these mined rules useful for? Authors propose to use them for generating wrong statements in watermarking schemes! (Is this a confirmation of the poor value of the mined knowledge with this method?) They also propose to generate probabilistic

KBs, but there are more straightforward and well-grounded methods to build much more effective statistical predictive models. Finally, authors compare their approach to two ILP methods which were proposed at earlier 2000. More robust and grounded method for mining relational data have been proposed since then. A fair
comparison to these methods needs to be carried out in order to demonstrate the difficulty of the problem and how far the precision scores are from these methods.

Reviewer #2: The article describes AMIE, a Horn rule mining approach for RDF knowledge base, and presents an improved AMIE+ with new techniques to scale the performance. AMIE mines rules in two separate steps. It first uses support as the pruning metric to iteratively find frequent Horn clauses of longer length; and then applies a confidence constraint under the Partial Completeness Assumption to filter out generated rules. AMIE+ mainly improves the performance of the second step. Since computing the exact value of a PCA confidence can be very expensive, AMIE+ uses the upper bounds and also approximations for rules of particular forms. Additionally, AMIE+ speeds up the rule generation process by avoiding redundant computations for non-closed rules or reusing previous results. The experiments show the AMIE+ can scale to larger knowledge base than AMIE, WARMR and Aleph. They also demonstrate that the PCA confidence is a good quality indicator for evaluating rules on knowledge
base, over which it is hard to compute the exact precision.

The first contribution of the article is a detailed framework for mining logical rules in RDF knowledge base (KB) using ideas inspired from the traditional association rule mining approaches. A key operator for AMIE is computing the support for candidate rules. However, the queries presented throughout the paper (mainly in Section 5.3 and 5.4) to compute the support appears to be incorrect. For each candidate rule, they are actually computing the total number of instantiations of the rule in the KB rather than the number of instantiations of the head atom that appears in some instantiation of the rule. For example, the example query in Section 5.4 on Page 11, shown as below, returns the number of instantiations of the rule "r(z, y), isMarriedTo(x, z) => livesIn(x, y)" for each relation r with the aggregate value larger than k.

```
SELECT R.x_r, SUM(R.N) FROM (
   SELECT B2.x_r AS x_r, COUNT(*) AS N
   FROM K AS H, K AS B1, K AS B2
   WHERE H.x1 = B1.x1 AND H.x2 = B2.x2
       AND B2.x1 = B1.x2 AND H.x_r = "livesIn"
       AND B1.x_r = "isMarriedTo"
   GROUP BY B2.xr, H.x1, H.x_r, H.x2) AS R
GROUP BY R.x_r
HAVING SUM(R.N) >= k
```

This may become clearer if it is simplified to the following equivalent un-nested query.

```
SELECT R.x_r, COUNT(*) AS pseudo_support
FROM K AS H, K AS B1, K AS B2
WHERE H.x1 = B1.x1 AND H.x2 = B2.x2
     AND B2.x1 = B1.x2 AND H.x_r = "livesIn"
     AND B1.x_r = "isMarriedTo"
GROUP BY R.x_r
HAVING pseudo_support >= k
```

The correct query should contain a semi-join as following:

```
SELECT rel_names.x_r, COUNT(*) AS support
FROM K AS H, (SELECT DISTINCT x_r FROM K) AS rel_names
WHERE H.x_r = "livesIn" AND
   EXISTS
      (SELECT *
        FROM K AS B1, K AS B2,
        WHERE H.x1 = B1.x1 AND H.x2 = B2.x2
            AND B2.x1 = B1.x2 AND B1.x_r = "isMarriedTo"
            AND B2.x_r = rel_names.x_r)
GROUP BY rel_names.x_r
HAVING support >= k
```

The article should carefully re-examine and re-write those SQL and SPARQL queries. Similarly, the Algorithm 2 is not very accurate. It would be clearer if "distinct" is added on line 15: "for all distinct x \in SELECT ?x FROM K WHERE q do." (In contrast, the DISTINCT in the query on the top of Page 11 is redundant).

The second contribution of this article is four lossless optimizations. The first three optimizations improve the performance of generating rules, while the last one optimizes the post-pruning process.
1) The first optimization avoids evaluating non-closed rules in the last iteration. The article lacks experimental results to show the effect of this technique.
2) The second optimization stops expanding a rule if it has a 100% PCA confidence. I found this a bit confusing. Since AMIE+ does not compute the PCA confidence during expanding rules, then the question is how AMIE+ identifies whether a rule has a 100% PCA confidence or not and whether the PCA confidence is actually computed. The article needs to clarify how and when this optimization can be applied.
3) Using the third optimization technique to expand a rule, AMIE+ chooses the same frequent relations for new atoms as in the last iteration when the rule is built if the rule is equivalent to its parent rule. The article uses an example to describe this technique, but lacks some formal definitions and discussions. The article should have more details to give answers to several critical questions: a) How to determine whether a new query can be rewritten to an old one? b) How the query rewriting is done? What rules are used in the query

rewriting? c) Can this technique be applied to general rules? Or does AMIE+ only use this this technique for those rules that have the exact form as the given example?
4) The fourth optimization technique uses a confidence threshold to filter out generated closed rules that have a small confidence upper bound to avoid expensive computations for its actual confidence. This technique can only be used for rules with exactly two body atoms of the same predicate sharing one variable. This restriction limits the benefits gained from applying this technique. As shown in Table 7 in the experiments section, we can see that there is not a big improvement in execution time and only a small amount of rules may be removed.

To be able to prune more general rules efficently, the article further presents an approximation method to estimate the PCA confidence of a rule. However, the rules that this technique can be applied to are still a bit limited. AMIE+ uses this only for a rule that has two body atoms, contains no constants and each variable occurs exactly two times. There are two additional issues with Section 6.2.
a) In the last paragraph of Section 6.2.1, the authors write that "where ov can denote ov_{ss}, ov_{so} or ov_{oo} depending on the joining variables of the argument". However, the query form for the expression is fixed and given. For the query form, ov should be specifically ov_{os}, since r_0 and r_1 shares x_1. The authors may intend to describe the notation for general expressions. Indeed, a general expression for general forms (i.e. all forms that satisfy the previously stated condition) should be given.
b) In Section 6.2.1 on Page 14, when the article introduces the notation of |rng(directed)|, it says it "is the number of distinct directors in the range of directed". I strongly suspect there is a typo here. |rng(directed)| should denote the number of distinct movie, instead of directors, in the range of directed. Each director directs on average 1/fun(directed) movies. Of those movie, each one has a probability of ov_{os}(directed, hasActor)/|rng(directed)| to appear in the join result. Therefore, the product of the two gives the average number of movies for each director that appears in the join between directed and hasActor. Using |rng(directed)| for the number of distinct directors does not make any sense.
c) There are two expressions in 6.2.2: one computes #x1 per x0 and the other compute #x2 per x1. Given the symmetry between r_0 and r_1 (directed and hasActor in the example), we can switch the two expressions. That is, we can use the expression with ov and |rng| to compute #x2 per x1 and use the other expression for #x1 per x0. Is there any preference of one way over another? How does AMIE+ choose?

The experiments show the good scalability of AMIE+ on large KB. To better understand the experimental results, the article should give a breakdown of the runtime. In particular, how long do the first and second step take respectively? The extensions in AMIE+ focus on the performance of the second step and they achieve impressive performance gains. This indicate the second step to post-prune generated rules dominates the execution time. This is quite different from other ILP methods where the majority time is spent on expanding rules. Also, the experiments show the improvement by the confidence threshold approaches (upper bound and approximation), but do not give any result for the other optimization techniques, especially the query rewriting and caching.

The article gives experimental results on the quality of rules to compare the PCA confidence with the standard confidence in Section 7.4.2. It runs AMIE on the YAGO2 dataset, and validates the predictions that are not in the dataset. If I understand this correctly, the predictions are made on the training set YAGO2, and those that do not exist in the training set are validated in a newer YAGO2s version or manually. To evaluate the prediction capability of a method, it is better to separate the training and the testing set. The article, however, generates the rules and uses the rules to generate predictions on the same dataset. Therefore, a possibly better design is to run AMIE on a training subset of YAGO2, apply those rules to a separate testing subset of YAGO2 to generate new predictions and validate each prediction on YAGO2 or in other ways. Similarly, when the article compares AMIE+ with other methods, the experiments should contain a type of experiments that separate
the training and the testing data and use recalls as well as precisions as metrics.

Other minor comments.
1) The standard ILP methods can be used to mine rules on knowledge base under the Partial Completeness Assumption. In fact, the Partial Completeness Assumption essentially presents a way to generate negative examples, which can be fed to an ILP method. It would be great if the article contains experiments where Aleph is provided with such negative examples.
2) In Section 4.4.1, it is not quite accurate to say that "Thanks to the FUN-property, the assumption is also true for inverse-functional relations." Strictly speaking, the assumption is true for inverse-functional relations only when we know the complete domain of the y variable in r(x, y). If there is an unknown value of y, we clearly cannot know whether r(x, y) is true for r, a given value x, and the unknown value of y. The article needs to make this implicit assumption explicit that the domain for each variable is complete.
3) The three operators used in the rule mining can actually be reduced to a single and simpler operator: add a new atom with a shared variable with the current rule.
4) The rdf:type relationship in a KB can be used to derive the type for each argument of a predicate. The information should be used, if it was not used, in the mode setting of Aleph to facilitate the learning.

In summary, the article presents interesting extensions to AMIE, and uses concrete and clear examples to describe the optimization techniques. However, it needs to address some issues. The important ones include:
a) The queries to compute the support need to be re-examined.
b) Some elaboration on 6.1.2 and 6.1.3 is needed with more formal discussions.
c) For completeness, a general form of the approximation expression of the denominator of the PCA confidence should be added, or briefly described. For example, what is the expression for a rule r_0(x1, x0), r_1(x2, x1) => r_h(x_0, x_2)?
d) The experiments should give a breakdown of the execution time of AMIE+ on the two rule mining steps and show the performance improvement on the respective step for each optimization technique.

Reviewer #3:
I have serious reservations on this paper, but I think that authors could improve the paper - with a hge amount of efforts and a complete rewrite of its most critical parts - hence I recommend a very major revision.

Let me start with the main criticism: why AMIE+? If one looks at the 4-atom rules of Table 2, none of them seems very good. E.g., in rule 1, why is the rule (living in X) -> (citizen of X) stronger because a citizen imports and exports the same goods? (note: being "same good" is an artifact of syntax limitation that you call "language bias", your language cannot say that either one import or one export are sufficient, which seems a more obvious rule - but still I don't see the improvement in adding imports or exports instead of the 2-atoms rule). Likewise, in rule 2, why a KB should store Capital(Rome, [.89,.78]) and not Capital(Rome, Italy) so as to need a 4-atom rule of inference? In rules 3, we infer the currency of a country from the currency of a city, isn't it the normally the other way around? And in rule 4, is LeaderOf (Person, Country) as in Rule 3 or the way around? (a Country cannot die). If these are the only 4 examples of good new rules

LeaderOf (Person, Country) as in Rule 3 or the way around? (a Country cannot die). If these are the only 4 examples of good new rules produced by AMIE+, I
am really not impressed.

My second concern has to do with the technical content of this paper, especially in Sections 5 and 6. It alternates theory with implementation, but it does not make a good service to either of them. What's most critical, it is weak in bridging theory to implementation. If is almost impossible to link definitions to algorithms and it would be impossible to replicate algorithms, which are delivered in a mixed formal - informal way that is not satisfactory. Here are some of my criticisms, which apply to Section 5 on AMIE - your previous work:

1. Algorithm 1 is too high level and as such it leaves too much unexpressed (e.g. "execute in parallel", "is not pruned for output").
2. SQL and SPARQL are presented at page 11 only to say that they are discarded for a custom implementation.
3. The "vanilla in-memory database" implementation is too informally described (e.g.: each "index is a map from the first item to a map from the second item to a set of the third item": a more formal description and a figure, please!

Then we move to Section 6 on AMIE+. I don't agree on how AMIE+ is presented. Authors start with an informal discussion of changes (6.1.1 and 6.1.2, but they describe "perfect rules" without really saying that AMIE+ "stops adding perfect rules" and without saying why such an obvious thing was not done in AMIE), then they go at low-level (6.1.3 for discussing a caching approach by using a pseudo-code SPARQL-ish that is not properly defined), then the go high-level to define confidence upper-bonds (with no link from mathematics to implementation). This is the "new part", but it misses a global introduction, a description of the rationale of additions bringing from AMIE to AMIE+, and a description of AMIE's implementation, with its internal data structures and a link from theory to implementation that shows how theoretical approximations are turned into clearly described, repeatable, working algorithms.

The wide set of experiments is good, but it recaps in 7.1 and 7.2 "obsolete" results of WWW where AMIE was shown superior to other systems - in spite of the fact that now AMIE+ is present and it is superior to AMIE; the really interesting new section is 7.3 comparing AMIE and AMIE+, and I must say that I learned (by intuition) more things about AMIE+ from Section 7.3 than from Section 6; but this is not how a paper should be written, as experiments should not reveal the semantics very late, when you are reading page 19; learning that the performance goes down from days to hours or even minutes is impressive but one would first be ensured that what you can find with such improved times is worth the effort.

Let me go back to the beginning of the paper. I really liked the intro and related work, they are illuminating about an interesting field of research, and raised a lot of positive expectations. Problems started at section 3. The FUN-property of a KB is problematic, it assumes something about the major KBs are written which may be factually true (e.g. in YAGO or DBPEDIA) but should really be verified on case-by-case (e.g., what about biological KBs?) so I think it must be contextualized to the KBs under investigations and be presented as a factual observation which holds in that limited setting. Rules are introduced with no limitations on their syntax but then, under the strange section heading "language biases", we discover that rules should be connected, closed, nonreflexive, and can be recursive (no example provided). There exists a better way of defining rule syntax & constraints! Incidentally, using a closed rule as example of generic rule does not help the reader's
intuition. The discussion on support and coverage is rather clear (but an example would not hurt).

For what concerns the central notion of convergence, discussed in Section 4, I liked and understood section 4.1 and 4.2, but I would like to understand sooner that 4.3 is a discarded and useless digression and I would appreciate a deeper discussion of Section 4.4, possibly with examples (note that 4.4 and 4.4.2 have the same title, which is not elegant). Then we come to Section 5, that is, where I started my review.

I am sorry that I didn't give very specific line-by-line suggestions, but I think the current format does not allow for that. I am ready to make an effort over a thoroughly rewritten, well organized version of this paper.

—