Last Updated : 3rd April 2017 (IST)

GSoC- 2017 Proposal

GSoC: Knowledge base embeddings for DBpedia

Contact information

- Your Name: *Nausheen Fatma*
- Email Address:
  fatma.nausheen@gmail.com; nausheen.fatma@research.iiit.ac.in
- Instant messenger name and protocol (if any):
- Any professional (e.g. github, linkedin) or personal (e.g. blog) URLs you would like to list:
  Github : https://github.com/nausheenfatma
  LinkedIn : https://www.linkedin.com/in/nausheenfatma/

Project

- Project Name: One of our suggestions?, or your own idea
  Your suggestion
  Name : Knowledge base embeddings for DBpedia

- Project Description: Your own description of the project and why it's interesting

Word embeddings has been found to be very useful in the research community in the recent years by bringing semantically similar word closer in the vector space. Word embeddings is being actively used in many applications such as sentiment analysis, recommendation systems, question answering, etc. Knowledge graphs have been popularly used for storing data in the form of graph for using entities and relationships. The objective of this project is to find embeddings for knowledge graphs entities and relationships. If we want to find word embedding for a movie like "Beauty and the Beast", simple aggregation/averaging of word embeddings for individual word tokens may or may not make great sense , as these words may be scattered far away in the text space. It would be very useful if we can get embeddings of the complete phrase as 1 atomic unit. Since knowledge graphs already stores the data in entities and relationship form, it would be very useful to get embeddings representation for the same.

- If you would be willing and able to work on another of our suggested project ideas instead, which ones?
  Working on your suggested project idea.

- Please describe why you are interested in this specific project:
  The project aligns well with my current research work, so I thought it would be a great idea to explore more in this direction.

- Please describe a tentative project architecture or an approach to it:
- Please detail an expected project plan and timeline with milestones:
- Please include in your plan how will you evaluate the performance of your contribution (in terms of time, or accuracy, or both), as well as which data sets you will use for that evaluation.

Please refer to the below detailed description for the previous 3 questions :

Objectives of the project:

i) Run the existing techniques for KB embeddings
ii) Evaluate across domains.
iii) Compare and Analyse various approaches.

(Before the actual coding begins, warm up tasks) (till 29th May)

1) **Literature Survey :** Read the research papers, and find the current state-of-the-art methods and other popular existing techniques. Narrow down the list of methods which are absolute necessary to implement.

We will look at those papers which relates to **link prediction** (predicting subject, predicate or object). One of the ways to find this could be to read the paper abstracts and find those papers who are using link prediction for their evaluation metrics to report results.

Right now, from the research papers I could find :

Translation models : **TransE (2013)** , TransH (2014), **TransR (2015)**

Composition models : RESCAL (2011) **, HOLE (2016)**

Current state of the art is : SSP (2017) (Open Source code not available)

New ideas around : **wiki2vec and fasttext**

I have shortlisted the following approaches with the reasons stated:

i) **TransE** has been used as a baseline for many of the research papers. It indicates that this algorithm is important benchmark. **TransE** could also act as our baseline. Since TransH is just an improvement around the same idea of TransE, I plan to drop it for my project, to work on a new approach/idea.

ii) **TransR** is also a translation model like TransE, but the semantic space for TransE and TransR is different. So, it would be good to explore this direction.

iii) **HOLE** is an entirely different approach. It is a compositional vector space model. And it is one of the recent papers with state-of-the-art-results.

iv) **wiki2vec approach :** And I would also like to run and work around **wiki2vec approach** (https://github.com/idio/wiki2vec), which would make use of Wikipedia text and evaluate this approach. It would help us to understand how well we can map entities in the text space. There doesn't seem to be any publication for this work.

v) **SSP** is the current state-of-the-art (AAAI 2017) which learns from symbolic KB data and textual descriptions. However, it has no open source code available. I would study this approach and compare its performance with iv), since both include textual semantic space. Implementing SSP from scratch for which open source code is not available is low priority and not the focus of this project. We can do the comparison using the results reported in the paper.

2) Understanding the Evaluation (dataset and metrics) :

We want to build a system which can do the following predictions :

a) entity predictions (Link prediction) :

i) predict head h, when the input is relationship and tail (_, r, t)
For example :

<___, dbo:capital, dbo:New Delhi>

The system should predict a ranked list of countries with “dbr:India” competing for the first place .

ii) predict tail t, when the input is relationship and head (h, r, _)

For example : <dbr:La_La_Land_(film), dbo:director,___>

The system predict a rank of candidate entities (probably directors like Damien Chazelle, Andrew Jarecki, Adam McKay , etc) for the tail t with Damien Chazelle competing for the first.

b) relationship predictions :

iii) predict relation r, when the input is head and tail (h,_,t) .

For example : <dbr:Barack Obama,____, dbr:Michelle Obama>

The system should predict the relationship here. The “spouse of” should compete for the 1st position.

However , there can be cases in which 2 entities have no relationship at all. For example: entity : “India” might have no relation with an entity, say , “Michael Jackson”.

We can come up with some threshold to ranking. Score below which the system should give an empty list. It would be an interesting thing to explore.

The triples can be of the following types:

1:1 case :
Example :

<India , capital,___> which has only 1 relevant **subject or object** prediction i.e New Delhi in this case.

1:N case :
Example :

<Angelina Jolie, type ,___> which can have multiple relevant **object** predictions i.e Thing, Person, Agent, Natural Person, Entertainer, etc.

N:1 case :
Example :

<_____, starring, Angelina_Jolie > which can have multiple relevant **subject** predictions i.e Shark_Tale, Lara_Croft_Tomb_Raider:_The_Cradle_of_Life, Foxfire_(1996_film), Gone_in_60_Seconds_(2000_film),True_Women, Pushing_Tin, etc.

I will evaluate 1:1 and N:N case using the same evaluation metrics. For 1:1 case, the relevant prediction should compete to occupy the 1st rank.
And for N:N cases, if there are n relevant predictions, they should compete for the first n places.

If the predictions (either belonging to 1:1 case or N:N case) does not occupy the highest ranks, the evaluation metrics would give low scores for later ranks, and hence we can easily compare the systems for both the cases using the same metrics.

Evaluation Metrics : These predictions would be a ranked list instead of single value predictions, and. The ranked lists would be evaluated using metrics **Mean Rank, Hits@10, and MAP.**

Consider the following examples :

Q No.	Input (s & p)	Ranked predictions for object o
1	Lil_Wayne, birthplace, ____	Atlanta, **New Orleans**, Austin, St. Louis, Toronto, New York City, Wellington, Dallas, Puerto Rico
2	J._K._Rowling, genre, ______	Animations, Computer Animation, **Comedy**, Adventure film, **Science Fiction**, **Fantasy**, Stop motion, Satire, **Drama,** Horror
3	Ryan_Gosling, producer, _____	**Lost River,** La La Land, **ReGeneration**, Half_Nelson, The_Ides_of_March, The_Believer_(film), Blue_Valentine_(film), Crazy,_Stupid,_Love, Breaker_High, The_Big_Short_(film),

		The_Place_Beyond_the_Pines, Murder_by_Numbers, Weightless_(film)

1) How to find MAP:

MAP = Mean of Average Precision
Average Precision (Q)= average of precisions at changing recall points

Average Precision for Q-2) :

Rank	Predicted Entity	Relevance	Precision at kth rank	Recall at kth rank
1	Animations	-	0/1	0/10
2	Computer Animation	-	0/2	0/10
3	**Comedy**	+	1/3	1/10
4	Adventure film	-	1/4	1/10
5	**Science**	+	2/5	2/10
6	**Fiction**	+	3/6	3/10
7	Stop motion	-	3/7	3/10
8	**Fantasy**	+	4/8	4/10
9	Satire	-	4/9	4/10
10	**Drama**	+	5/10	5/10

Recall Changes at Orange highlighted points above.

Average Precision (Q2) = (1/3+2/5+3/6+ 4/8+ 5/10)/10=0.223

Similarly:
Average Precision (Q1) = (½)/10=0.05
Average Precision (Q3) = (1/1+ 2/3)/10=0.167

MAP = (Average Precision (Q1) +Average Precision (Q2) +Average Precision (Q3)) / 3
MAP = (0.223+0.05+0.167)/3 = 0.147

2) How to find Hits @ 10 (or Precision @ 10):

Number of relevant predictions in the first 10 ranked predictions :

Hits @ 10 for Q1= 1/10
Hits @ 10 for Q2= 5/10
Hits @ 10 for Q3= 2/10

Average Hits @ 10 =(0.1+0.5+0.2)/3=0.267

3) Mean Rank
Mean Rank for Q1 =2/10=0.2
Mean Rank for Q2 = (3+5+6+8+10)/10=3.2
Mean Rank for Q3 =(1+3)/10=0.4

Average Mean Rank=(0.2+3.2+0.4)/3=1.27

A good ranked linked will have higher MAP, Hits @ 10 and lower Mean Rank.

Challenge :
There can be "special cases" like rdf:type and dct:subject. rdf:type predicate can have many predictions (much more than 10) for a given subject entity. The system may end up giving the most common predictions for all cases i.e: Agent,Thing,Person,Entity,Object, etc. So it can be difficult to assess the performance. Since these predictions are true but the important or more specific ones might be missing. So in evaluating such cases, the more specific object "type" such as "Actor" are more relevant and should be given better score than the a tag which belongs to a more general class such as "Thing", "Person" or "Agent", etc.

Idea to resolve the above challenge :

In order to deal with the "special" properties, instead of having just binary relevance, we can have floating point relevance.

We give 1 full point to the most specific class but smaller weights to super classes, and 0 points to the most generic class i.e owl:Thing

In order to decide the weights we can consider the total number of hierarchy levels an object type fall into.

For example, consider the triple :

P._V._Sindhu, rdf:type, ______

There can be many prediction which can be true for this case :
owl:Thing, dbo:Agent , dbo:Person, dbo:Athlete, dbo:BadmintonPlayer, yago:Female109619168 , etc.

Now the most specific type is dbo:BadmintonPlayer.

dbo:BadmintonPlayer has the following super classes :
dbo:BadmintonPlayer -> dbo:Athlete->dbo:Person->dbo:Agent -> owl:Thing

Lets ignore the owl:Thing for now as owl:Thing is the root class for every entity in DBpedia, so no points for guessing owl:Thing.

dbo:BadmintonPlayer -> dbo:Athlete->dbo:Person->dbo:Agent

We see the most specific class is at level (height) 4.

We give (1/level) points for getting more and more specific class.

For example, if we guess dbo:Agent, we give 1/4 points.
if we guess dbo:Person,we give 2/4 points.
if we can guess dbo:Athlete ,we give 3/4 points.
and if we can guess dbo:BadmintonPlayer, we give 4/4 points.

Lets, try the example with Average Precision for 5 ranks:

Q No.	Input (s & p)	Ranked predictions for object o
1	P._V._Sindhu, rdf:type, ________	Thing, Work, BadmintonPlayer, Person, Actor,

Average Precision for Q-2) :

Rank	Predicted Entity	Relevance	Precision at kth rank	Recall at kth rank
1	Thing	0	1/1	1/5
2	BadmintonPlayer	+1	2/2	2/5
3	Work	-1	2/3	2/5
4	Actor	-1	2/4	2/5
5	Person	+2/4	3/4	3/5

Unweighted Average Precision at changing recall points :

(1/1+2/2+3/4)/3
=(1+1+0.75)/3
=0.916

Weighted Average Precision at changing recall points :

=((1/1*0) +(2/2 *1) + (¾ * 2/4))/3
=(0+1+0.375)/3=0.458

which is lower than the unweighted average precision. But if we exchange the position of Person with Thing, it would make no difference in the score. Therefore, we consider the weighted average technique.

Now, if we exchange the position of Person and Thing, the weighted average would be :

=((1/1*2/4) +(2/2 *1) + (¾ * 0))/3
=(0.5+1+0)/3
=0.5

which is higher than 0.458.

In cases where we have to predict single object entity such as

La La Land ,director, ____

Damien Chazelle

or multiple object entities, such as :

La La Land ,starring,

Emma Stone, Ryan Gowsling

we give just +1 relevance in such cases.

Thus the proposed method would apply to all use cases :

i) single object entity prediction.
ii) multiple object entity prediction.
iii) multiple class type prediction.

[29th May- 15th June]

3) **Replicate results on existing datasets** (simply run open source on their data) :

Standard datasets are found to be : WN18 & FB15

Dataset Sizes :

WN18 :
Test - 5000 samples
Train - 141442 samples
Valid-5000 samples

FB15:
Test - 59071 samples
Train - 483142 samples
Valid-50000 samples

Run the open source codes which can be found here :
https://gist.github.com/mommi84/07f7c044fa18aaaa7b5133230207d8d4

This may require some time. Estimated time from the literature survey :

In AAAI15 paper (TransR),
http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/viewFile/9571/9523, it is written "the training time of TransE and TransH are about 5 and 30 minutes, respectively. The computation complexity of TransR is higher than both TransE and TransH, which takes about 3 hours for training." This is the training time for the WN18, and FB15K datasets.

[15th June- 30th June]

We already have whatever we require in place. And we have run the codes on existing datasets. We want to test this on DBpedia now.

4) **Select and run code on a subset from DBpedia for evaluations :**

We can use all the relationships from DBpedia of entities (which are linked to freebase and are present in FB15K) to build a new dataset.

DBpedia entities are linked to their Freebase equivalent with predicate relation “owl:sameAs” . For example, consider the following triple from DBpedia which shows how DBpedia entities are connected with Freebase:

<dbr:Emma Watson, owl:sameAs , freebase:Emma Watson>

i)We can find all the relationships for dbr:Emma Watson for a freebase entity freebase:Emma Watson found in FB15K.

ii) Select only those relationships which contain DBpedia URIs only and not string objects.

I am planning to consider :
1) object properties (entity values such as dbr:New_Delhi)
2) "special" properties such as rdf:type and dct:subject
Right now I am not considering the datatype properties which have literal values (strings, numbers, etc.)

We can use this dataset for our training and evaluations.

I have already made a code for building the DBpedia equivalent FB15K , which can be found on : https://github.com/nausheenfatma/Summer2017

Size of the dataset :
Training time estimate :

<u>5)</u> **<u>Train and test on a subset of DBpedia for a domain</u>** (e.g. Movies/Actors or Music/Artists)

Select a subset say X, from DBpedia dataset from 4) for evaluation, and run the existing codes on X quickly.

The evaluation might give little insight for the small dataset X, if this X is chosen randomly from DBpedia. Therefore, we would chose the X from some popular domain, say “Movies/Actors ” or “Music/Artists”. It would be a lot easier to understand and check if the method is “making some sense” on this small domain or not.

We should also plot the data for 2D visualization (using PCA/t-SNE) for a few examples, which would help us to understand the semantic space like whether:

i) all the entities of type “Movie” ,”Country” close by **(entity type)** ?

ii) Or are entities with similar predicates **(entity type+ relationship)** like “president” or “starring”, etc. close by. Example : India is an entity of type:Country, and India has predicates like “capital”, “president”. Does that make “India” closer to another entity “America” which also has some common predicate like “capital”, “president”,etc.

iii) Or are the movies who share the same director/actor close by **(subject+relationship+object type)**. Example: Ryan Gosling has “starred” in mutiple entities of type “Movies” like “The Notebook” , “La La Land”, etc. Are those movies closer by than the others?

We should investigate such questions through examples and visualizations.

[1st July - 28th August]

6) Train on all of DBpedia (or a large subset) and evaluate on 5.

7) Train on all of DBpedia and evaluate on all of DBpedia (by holding out data ahead of training time). For this case, we will evaluate not just on a specific domain, but randomly taken held out data from DBpedia. This can be ensured by taking triples where the subject and objects for the triples belong from 100(or some considerable number) different rdf:types /classes for example “Movies”, “Country”, “Person”, etc. to ensure variety of objects in subjects/objects.

The statistics can be found at
http://wiki.dbpedia.org/dbpedia-2016-04-statistics

If we go to section **3.Entity Type (or Class) Statistics** in the above link we see statistics of various entity types. We randomly take 100 triples from each entity type for building our final validation set for whole DBpedia.

8) Analyse the results : After running all the codes across all the datasets, we will analyse and document the following :

i) ***Compare the techniques :*** Check which of the techniques are working best and why. Use 2D visualisation (using PCA/t-SNE) with a few common examples for all the approaches to understand the comparison.

ii) ***Look for consistency*** : Find whether the results for various approaches for the DBpedia dataset are consistent with other datasets -WN18K and FB15K.

For example : Is TransH better than TransE for both DBpedia as well as WN11K and FB15K / Are the rankings of all the techniques consistent?

iii) ***Report any challenges*** we come across with the DBpedia dataset, and think of some optimal ways to handle those challenges which require minimal manual efforts.

DBpedia dataset has not been used previously in any of the research papers. This project will help us to :

a) create a new standard dataset from DBpedia for research purpose.
b) verify the research already done on this DBpedia dataset.
c) identify problems/limitations in this DBpedia dataset.

Related Work

Short notes/pointers from the papers :

1) TransE (NIPS 2013)

Broad idea : a method which models relationships by interpreting them as translations operating on the low-dimensional embeddings of the entities.

Semantic space : Relationships are represented as translations in the embedding space:

if (h, l , t) holds, then the embedding of the tail entity t should be close to the embedding of the head entity h plus some vector that depends on the relationship .

Motivation :

The main motivation behind our translation-based parameterization is that hierarchical relationships are extremely common in KBs and translations are the natural transformations for representing them.

Another, secondary, motivation comes from the recent work of [8], in which the authors learn word embeddings from free text, and some 1-to-1 relationships between entities of different types, such "capital of" between countries and cities, are (coincidentally rather than willingly) represented by the model as translations in the embedding space. This suggests that there may exist embedding spaces in which 1-to-1 relationships between entities of different types may, as well, be represented by translations. The intention of our model is to enforce such a structure of the embedding space.

How to evaluate on test set :

For each test triplet, the head is removed and replaced by each of the entities of the dictionary in turn.

Dissimilarities (or energies) of those corrupted triplets are first computed by the models and then sorted by ascending order; the rank of the correct entity is finally stored. This whole procedure is repeated while removing the tail instead of the head. We report the mean of those predicted ranks and the hits@10, i.e. the proportion of correct entities ranked in the top 10.

$h + l \approx t$ when (h, l, t) holds (t should be a nearest neighbor of $h + l$), while $h + l$ should be far away from t otherwise

Shortcomings:

i)Good mostly for 1:1 relationships.

ii) No text information apart from KB triples is used.

Open Source Code at:
https://github.com/thunlp/KB2E/tree/master/TransE

2) TransH (AAAI 2014) :

* Same concept as TransE- ie. same semantic space, translation model.
*overcomes the flaws of TransE in dealing with

reflexive/one-to-many/many-to-one/many-to-many relations while keeping the model complexity almost the same as that of TransE.
The basic idea behind TransE is that, the relationship between two entities corresponds to a translation between the embeddings of entities, that is, $h + r \approx t$ when (h, r, t) holds. Since TransE has issues when modeling 1-to-N, N-to-1 and N-to-N relations, TransH is proposed to enable an entity having different representations when involved in various relations.

3) TransR (AAAI 2015):

Broad Idea : Models such as TransE and TransH build entity and relation embeddings by regarding a relation as translation from head entity to tail entity. We note that these models simply put both entities and relations within the same semantic space.

Semantic space :
In this paper, we propose TransR to build entity and relation embeddings in separate entity space and relation spaces. Afterwards, we learn embeddings by first projecting entities from entity space to corresponding relation space and then building translations between projected entities.

Motivation :
relations and entities are completely different objects, it may be not capable to represent them in a common
semantic space.

4) HOLE- AAAI-16

Broad Idea : In this work, we propose holographic embeddings (HOLE) to learn compositional vector space
representations of entire knowledge graphs.
Open Source Code at:
https://github.com/mnick/holographic-embeddings

5) SSP AAAI 2017

Broad Idea : this paper proposes the semantic space projection (SSP) model which jointly learns from the symbolic triples and textual descriptions. Our model builds interaction between the two information sources, and employs textual descriptions to discover semantic relevance and offer

Open source code : https://github.com/BookmanHan/Embedding

(Have to email the authors to confirm if this the SSP implementation)

Translation models : TransE/TransH/TransR/CtransR
Compositional models : RESCAL/HOLE

Technical skills

- Please describe in a few lines your programming knowledge or experience (if any):
  Python, SPARQL, Scikit-Learn, Keras

- Please describe any other project related experience (if any):
  I have been working on mining Trivia/Interesting facts from DBpedia dataset for the past 1.5 years. In fact recently, I got a research paper also on that in AAAI-17.

- Please provide one or more URLs to code samples that you have written in the past or to DBpedia Spotlight's SVN/issue tracker for which you have provided a fix in the form of a pull request. (Optional but highly appreciated if provided. Proof of code ownership is also required.):

None

Open Source

- Please describe any previous Open Source development experience:
  None

- Why are you interested in Open Source development ?
  Contributing to the community, networking with awesome people

Background & education

- What school do you attend?
  International Institute of Information Technology, Hyderabad

- What is your specialty/major at the school?
  MS by Research in Computer Science and Engineering

- How many years have you attended there?
  2 years 8 months

Research

- What is your current research experience? Please point us to the best paper you have read (preferably in the context of your project proposal), and the best paper you have written, if any.

2.5 years of experience as a Masters Research Student in the NLP-MT Lab of Language Technologies Research Center, IIIT-Hyderabad.

Best paper read for this project :

SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions (https://arxiv.org/pdf/1604.04835.pdf)

Best paper I have written :
The Unusual Suspects: Deep Learning Based Mining of Interesting Entity

Trivia from Knowledge Graphs
(http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14585)

- Would you be interested in co-authoring a conference paper with your mentors about your work in GSoC 2017?
  *A big **Yes** !!*

Summer plans

- What city/country will you be spending this summer in?
  Hyderabad and New Delhi, India

- How much time do you expect to have for this project (est. per day and per week)?
  30 hours a week

- Please list jobs, summer classes, and/or vacations that you'll need to work around:

Writing my Masters thesis, finding industry opportunities would mainly consume my summer time. My sister is expected to deliver a baby in mid-July, I am planning a 1 week break during that time :)

GSoC Experience

- Did you participate in a previous Summer of Code project? If so, please describe your project and experience:
  No. This is the first time I have applied in GSoC.

- Have you applied or do you plan to apply for any other 2017 Summer of Code projects? If so, which ones?
  No. This is the only project I applied.

- Why did you decide to apply for the Google Summer of Code ?

My institute IIIT-H is the second highest school for GSoC participation (source : https://en.wikipedia.org/wiki/Google_Summer_of_Code). I have seen many students here participating in GSoC every year and flaunting *their GSoC T-Shirts which I find really cool. I would be graduating before next GSoC, and don't know if I would ever get this chance to participate in GSoC. GSoC is a great platform for expanding your global network by connecting and learning with such great mentors, and it gets your work and profile more visible to the world. What more a student can ask :) ?*

- Why did you decide to apply for a DBpedia Spotlight project ?

And finally...

- ... in 2 sentences, **why should we take YOU** ?

I have worked on DBpedia data for the past 1 year for my Masters thesis which had a closely related sub task. Given the time constraints of GSoC, I believe this is an extra advantage I have to understand and implement things quickly.