Bridging the Usability–Expressivity Gap in Biomedical Data Discovery

Maulik R. Kamdar
Biomedical Informatics Training Program
Stanford University
maulikrk@stanford.edu

Abstract

During the scientific discovery process, biomedical researchers pose complex questions that require precise answers or relevant datasets for further analysis. Semantic Web technologies provide a solution to structured, expressive, integrated search, but they pose a steep learning curve for domain users. Hence, there is a usability–expressivity gap in biomedical data discovery. In this work, we will look at the translation of Natural Language Queries and Statements to structured triple patterns (*head–relation–tail*), observed in RDF graphs and SPARQL queries. We generate word embeddings from a large corpus of biomedical literature, taking into consideration different named representations of biomedical entities. We also generate triple pattern embeddings by translating the relations in hyperplanes. We use a neural network architecture to map syntactic elements in a constituency-based parse tree to structured triples. We evaluate the generated word embeddings intrinsically by a visualization of biomedical entities, and extrinsically by classifying each abstract into biomedical topics. We also evaluate the performance of the architecture for NL–triple translation for a drug knowledge base.

1 Introduction

During the hypothesize-test-evaluate cycle of the scientific discovery process, biomedical researchers can formulate questions, such as **Q1**:*"Which antineoplastic agents target IDH1 gene in glioma patients?"* or hypothetical statements, such as *"Gene A regulates Gene X and Gene Y in location M"*. The above examples requires precise answers (the corresponding drugs or genes) or relevant *–omics* datasets for further analysis to generate those answers. Such queries cannot easily be answered using traditional search engines, that follow keyword matching paradigm. To obtain answer of such questions currently, a biomedical researcher needs to query multiple heterogeneous databases with varying representations and formats, and aggregate the query results manually.

To address the the data integration challenges, researchers have started using Semantic Web technologies, such as the Resource Description Framework (RDF) and the SPARQL structured query language. Semantic Web technologies help generate a linked and heterogeneous data space that extends over the traditional Web [1]. Multiple data sources can be published as RDF graphs, and similar entities in these graphs are represented using Uniform Resource Identifiers (URIs) or are linked to each other using cross-reference (*x-ref*) attributes. Several projects, such as Bio2RDF [2], EBI RDF Platform [3] and Linked TCGA [4], use Semantic Web technologies to build the Life Sciences Linked Open Data (LSLOD) network from a diverse set of heterogeneously formatted biomedical data sources. The LSLOD network aims to provide a unified, machine-readable data space that aids semantic normalization, web-scale data computation and heterogeneous data integration. Biomedical researchers can query the heterogeneous sources integrated in the LSLOD network through a unified querying interface using the SPARQL query language.

However, for wet laboratory biomedical researchers, both RDF and SPARQL have steep learning requirements. Whereas SPARQL queries are very **expressive**, allowing a user to formulate exhaustive queries across the LSLOD cloud to retrieve precise results (e.g. *"list antineoplastic agents that have molecular weight less than 200mg"*, Natural Language (NL) queries are more **usable**. Hence, semantic search and consequently, integrated biomedical data discovery suffers from this usability–expressivity gap. Hence, a natural language querying method over the LSLOD network that can enable scalable, autonomous discovery of relevant answers and datasets for evaluating hypotheses is required. For this work, we will look into the translation of NL-queries and statements to structured triples *head–relation–tail*, observed in RDF graphs and SPARQL queries.

With the advent of better methods to generate word embeddings [5] and robust, scalable neural network architectures, we will look at a different approach to the NL–SPARQL translation in the biomedical domain. The rest of the paper is organized as follows: In **Section 3** we discuss our approach towards generating word embeddings from biomedical literature, graph embeddings from a data source in the LSLOD cloud, structured in RDF, and our a neural network architecture that combines the salient features of Recursive Neural Networks (RNN) and Long Short-term Memory Networks (LSTM). In **Section 4**, we describe the methods and a real biomedical linked dataset, which we used to evaluate the word embeddings and our architecture. In **Section 5**, we will describe the results of these evaluations. Finally, we will summarize the findings of this project and lay out a future plan to extend it over the entire LSLOD cloud.

2 Related Work

Most current information retrieval methods fall on the intermediate stages on the usability–expressivity spectrum, from structured search, visual query systems, keyword search, to NL-queries [6, 7]. Those methods that do translate NL-queries to structured queries require exhaustive entity indices, domain vocabularies and structured templates, hence are not scalable for querying the entire LSLOD network [8, 9]. NL-querying search engines end up maintaining an inverted index of terms and the documents in the corpus, in which these terms occur. More recently, these search engines are grappling with deep learning methods and word embeddings, but they may fall short to fulfill the needs of biomedical data discovery and question–answering. Newer methods to deal with NL-querying over structured tables include Neural Enquirer [10] and Neural Programmer [11], that encode the user query as well as the entire table using embeddings.

On the other hand, with the advent of the Semantic Web, knowledge graph embeddings have become quite popular recently. A knowledge graph is a multi-relational graph composed of entities as nodes and relations as different types of edges. Each edge can be represented as a triple pattern (i.e. *head entity*$\rightarrow$ *relation* $\rightarrow$ *tail entity*). Knowledge graphs such as Freebase [12], DBPedia [13] and Gene Ontology [14] have become important resources in many applications such as question–answering [9] and semantic search [6]. However, navigating these knowledge graphs and querying is cumbersome and tedious, and most applications often involve numerical computation in continuous spaces. There are several methods recently developed to generate these knowledge graph (or triple pattern) embeddings, where each entity $head$ or $tail$ is represented as a point in a vector space and the $relation$ represents an operation (may or may not be in the same vector space) [15, 16, 17].

3 Approach

3.1 Word embeddings from biomedical publications

We generate the word embeddings using MEDLINE [18], a database of journal citations and abstracts for biomedical literature from around the world. For this work, we only use the title and the abstract of the citation. Biomedical literature is different from conventional text, in the sense that biomedical literature may have the same named entity (e.g. Gene or a chemical) referenced in different publications using different notations. For example, HIV may also be referenced as "Human Immunodeficiency Virus". In the latter case, we want to indicate that the entire term represents the same entity. This might not be possible using simple tokenization. We use the PubTator API [19] to annotate different biomedical entities in the MEDLINE abstracts of the following types — Gene, Chemical, Disease, Species, Single Nucleotide Polymorphisms and Other Mutations. We also sub-

stitute all kinds of numerical values using the NNNNNNN token. Finally, we use the Stanford NLP tokenizer to tokenize the title and abstract. An example of the abstract, before and after preprocessing, is shown in the Listing 1. It might be interesting to employ word embeddings for named entity recognition in biomedical literature, but that is beyond the scope of this project. Moreover, we pretrain word vectors before use in NL–SPARQL translation, as the data repository used for the experiments and evaluation of the architecture in the latter task is small. We use the GloVe software [5] to generate the word embeddings from the tokenized biomedical publications.

Listing 1: MEDLINE title and abstract before and after preprocessing

```
Effect of hemodialysis on methylprednisolone plasma
    levels. The effect of hemodialysis on methylprednisolone levels in
    uremia was investigated. Methylprednisolone 15 mg/kg was given
    intravenously over a period of 20 min ....

effect of hemodialysis on id_chemical_d008775 plasma levels . the effect
    of hemodialysis on id_chemical_d008775 levels in id_disease_d014511 was
     investigated . id_chemical_d008775 NNNNNNN mg/kg was given
    intravenously over a period of NNNNNNN min ....
```

3.2 Triple pattern embeddings from an RDF graph

The data sources in the LSLOD cloud are published and linked together as RDF graphs. Each such RDF graph Δ can be decomposed into triple patterns (i.e. *head entity*$\rightarrow$ *relation* $\rightarrow$ *tail entity*). Each entity h or t can be represented as a point ($\mathbf{h}$ or $\mathbf{t}$ respectively) in the vector space, and the relation r can be an operation (translation, projection, etc.) that can be represented by a vector $\mathbf{r}$. The representations of these entities and relations can be obtained by minimizing a global loss function involving the entities and relations. Simple translation-based methods assume all relations to be in the same vector space as the entities (e.g. TransE [20]) and use a scoring function $f_r(\mathbf{h}, \mathbf{t}) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||$. These methods do not consider reflexive relations (i.e. $(h, r, t) \in \Delta, (t, r, h) \in \Delta \Rightarrow \mathbf{r} = 0, \mathbf{h} = \mathbf{t}$) or one–to–many, many–to–one relations (i.e. $\forall i \in \{0, \ldots, m\}, (h_i, r, t) \in \Delta \Rightarrow \mathbf{h_0} = \ldots = \mathbf{h_m}$). However, in the LSLOD, RDF graphs always have such reflexive biomedical relations (e.g. gene reverse regulation Gene_1 *regulates* Gene_2 and Gene_2 *regulates* Gene_1), as well as one–to–many, many–to–one relations (multiple drugs used to treat a disease).

Hence, we use the method TransH[1] proposed by Wang, et al. [15] to generate triple pattern embeddings by translating the relations in different hyperplanes. We summarize the equations proposed by Wang, et al. below, where $f_r(\mathbf{h}, \mathbf{t})$ denotes the new scoring function, $\{(w_r, d_r)\}_{r=1}^{|R|}$ denote relations hyperplanes and translations respectively, and $(\mathbf{h} - \mathbf{w}_r^T \mathbf{h} \mathbf{w}_r)$ indicates the projection of $\mathbf{h}$ in the hyperplane $\mathbf{w}_r$. $\{e_i\}_{i=1}^{|E|}$ indicate entity embeddings, and Δ' indicates a corrupted graph, generated by incorrect triples. The function $[x]_+$ is a RELU function $max(0, x)$, and $\mathcal{L}$ indicates the loss that needs to be minimized, that includes the constraints (unit length vectors, orthogonal hyperplanes). $\gamma, C, \alpha, \epsilon$ are the hyper-parameters that can be tuned.

$$f_r(\mathbf{h}, \mathbf{t}) = ||(\mathbf{h} - \mathbf{w}_r^T \mathbf{h} \mathbf{w}_r) + \mathbf{d}_r - (\mathbf{t} - \mathbf{w}_r^T \mathbf{t} \mathbf{w}_r)||_2^2$$

$$\mathcal{L} = \sum_{(h,r,t) \in \Delta} \sum_{(h',r',t') \in \Delta'_{(h,r,t)}} [f_r(\mathbf{h}, \mathbf{t}) + \gamma - f'_r(\mathbf{h}', \mathbf{t}')]_+ + C \left\{ \sum_{e \in E} [||\mathbf{e}||_2^2 - 1]_+ + \sum_{r \in R} \left[\frac{(\mathbf{w}_r^T \mathbf{d}_\mathbf{r})^2}{||\mathbf{d}_r||_2^2} - \epsilon^2 \right]_+ \right\}$$

3.3 Neural network architecture

We develop a novel neural network architecture to map syntactic elements in an NL-query or statement to structured triples (cartoon **Figure 1**). This architecture combines the salient features of Recursive Neural Networks (RNN) and Long Short Term Memory (LSTM) Networks.

[1] https://github.com/thunlp/KG2E

For a given RDF graph, we generate entity embeddings $\{e_i\}_{i=1}^{|E|}$, relation hyperplanes and the represented translation vectors $\{(w_r, d_r)\}_{r=1}^{|R|}$, using the TransH method described above. We then encode each triple (h, r, t) in the RDF graph to the following vector $< e_h, w_r, d_r, e_t >$ by concatenating the entity embeddings. The concept is that each syntactic element (represented as a combined embedding from its subtrees $h = [[h_{Left}, h_{Right}]W + b]_+$) will be used as an input to an LSTM activation unit. The order of the triples does not matter in an RDF graph or a SPARQL query, however there may be parent nodes in the syntax tree for which we do not wish any triple to be output. Hence, we also generate a NULL triple representation, that will be ignored when aggregating the triples for the entire NL-statement or query. The RNN determines the best syntax tree, and the LSTM units determine if a node may not be directly converted to the triple pattern vector at a given depth, or is memorized for later use (e.g.**VP** *target IDH1 gene* ...). Some nodes may be ignored (e.g. determiners). To simplify, the current neural architecture only considers NL-statements as NL-queries will require "variable" representations for the SPARQL queries that is beyond the scope of this project.

During training, we consider the root mean square error ($RMSE = \sqrt{\frac{\sum_{t=1}^{N}(\hat{y}-y)^2}{N}}$) on the output of the LSTM unit instead of the cross entropy loss, commonly used during the tasks of classification. The final output of the LSTM unit will be a vector of the same dimensions as the triple pattern vector.

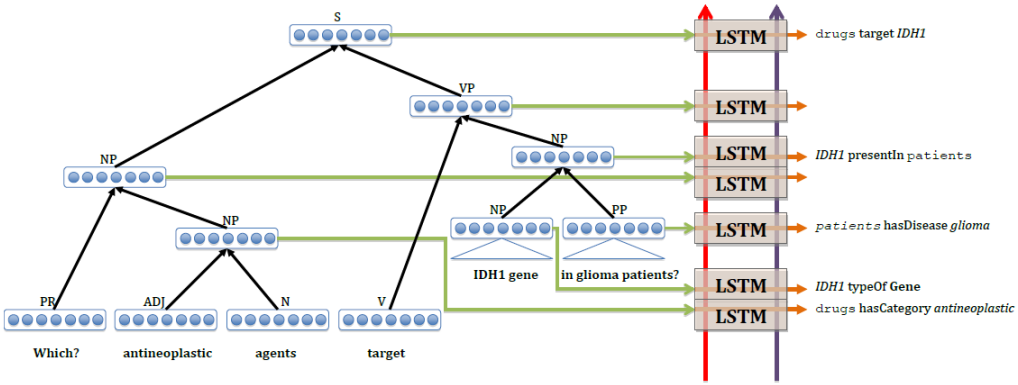

Figure 1: Neural network architecture to map syntactic elements to structured triples. This architecture is inspired from a combination of Recursive Neural Networks and LSTMs.

4 Experiments

4.1 Evaluation of word embeddings

We train GloVe over 20 iterations with an α learning rate of 0.75. We generate 100-dimensional word embeddings. We evaluate the word embeddings using two different methods. We reduce the dimensions of the word embeddings and visualize **only** the biomedical entities in a 2-dimensional space. We color the different entities based on their type (Gene, Chemical, Disease, Species, Single Nucleotide Polymorphisms and Other Mutations), as detected from the PubTator API.

We also perform an extrinsic evaluation. Each publication in MEDLINE is annotated with MESH (Medical Subject Heading) terms (numbers may vary from 5–50), that serve as topics for the publication. These publications are manually annotated, and generating automatic MESH recommendations is an interesting problem in the biomedical domain [21]. We generate a embedding vector for each abstract (by averaging across the word embeddings) and use a 2-layered Neural Network, with 20 hidden nodes in the first layer and 300 hidden nodes in the second layer to generate MESH recommendations. We calculate precision and recall using different thresholds on the resultant probabilities. We use the hyper-parameters learning rate $\alpha = 0.0001$, regulation $\lambda = 0.0001$, 25 epochs and batch size of 100. For the baseline, we use a simplified version of the method mentioned in [21], by first computing a TF-IDF vector for each abstract based on the count frequencies of the words in the abstracts. Using k-nearest neighbors (KNN) algorithm, we determine the MESH term of a given abstract by collecting the MESH terms of its neighbors, and scoring by increasing common counts.

4.2 Evaluation of the architecture

For this work, we use the DrugBank knowledge base [22] in the LSLOD network. DrugBank is an important biomedical repository of drugs, drug targets and drug interactions. An example triple pattern in the DrugBank knowledge base is `drugbank:DB00001 drugbank_vocabulary:target drugbank:BE0000048`.
To create the training, validation and testing dataset, we use a domain–independent sentence generation method [23] to convert RDF triples to an NL-statement (e.g. `Lepirudin` *targets* `Prothrombin`). The NL-statement is annotated using the PubTator API, as we have generated embeddings for annotated biomedical entities (**Section 3**). We use the intermediate tree structure created during this process to annotate desired triple patterns for key syntactic nodes. These triple pattern-annotated syntax trees form our datasets.

We compute the entity and relation embeddings using TransH over the triples in the DrugBank RDF graph. We train TransH using the learning rate $\alpha = 0.05$, constraint hyper-parameter $C = 0.25$, corrupted triple margin $\gamma = 1$ and the orthogonality error hyper-parameter $\epsilon = 0.01$. We train the neural network architecture for two learning rates $\alpha = 0.001$ and $\alpha = 0.005$, regulation $\lambda = 0.001$, 25 epochs and a batch size of 50 NL-statements. We will evaluate the performance of the architecture using precision and recall of the translated triples. These metrics will be calculated by estimating the k-nearest triple pattern vectors for the output of each LSTM unit, given a NL-statement. We will then set different thresholds on k to determine the number of accurate triple pattern vectors in the set of the output triple patterns. There is no current gold standard or an expert curated set of triple pattern-annotated syntax trees for the ground truth.

5 Results

5.1 Biomedical Word Embeddings

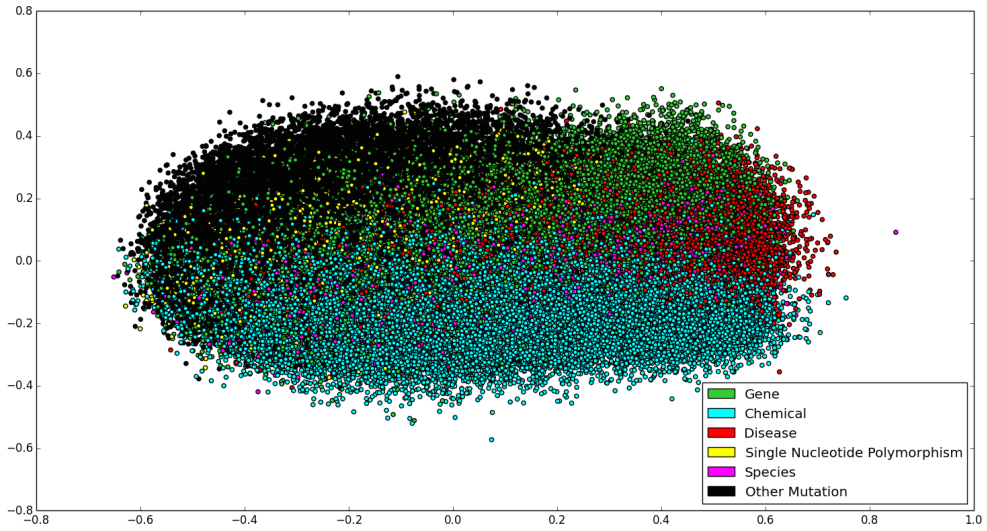

Figure 2: Visualization of the Entity Word Vectors using Principal Component Analysis. It can be seen that the entities of the same type — Gene (Green), Chemical (Blue), Disease (Red) and Mutation (Black), almost cluster together in a 2-dimensional space.

Our corpus of MEDLINE had a total of 1,568,472,762 abstracts with 3,574,811,534 tokens and 10,989,054 unique words. While generating the vocabulary, we removed those words that had a count of less than 5 occurrences. This threshold was determined as many abstracts had spelling errors resulting in different words (e.g. *plasmoduim* instead of *plasmodium*) The final vocabulary size was 2,218,381 words, with 139,491 distinct biomedical entities (The latter statistic does not include words like *id_chemical_d008775*-treated, etc.). We executed dimensionality reduction on

the 100-dimensional word vectors of biomedical entities using the Principal Component Analysis method, and visualized the entities as shown in the scatter plot (**Figure 2**). It can be seen that the entities of different types — Gene (Green), Chemical (Blue), Disease (Red) and Mutation (Black), almost cluster together in a 2-dimensional space. Entities of type Single Nucleotide Polymorphisms (Yellow) and Species (Purple) are scattered around the cluster of Gene and Disease nodes predominantly. A possible explanation may be because SNPs are single base changes generally found in Genes, and diseases may be mentioned or tested in the context of a given organism.

We evaluate our generated biomedical word embeddings also for the task of MESH term recommendations. We extracted these MESH terms from the MEDLINE corpus and only considered those MESH terms for which more than 5 publications are annotated. Hence we have a total of 26,995 MESH term labels. We also consider those publications with an abstract and corresponding MESH terms. We split the MEDLINE corpus into training, validation and test. The training set had around 8.3 million abstracts, validation and testing sets had around 2.7 million abstracts each. We threshold the resultant probability predictions for the MESH topics at different thresholds. The highest precision and recall metrics for the baseline model (KNN, with TF-IDF vectors, $k = 6$) was 0.31 and 0.45 respectively, which is almost close to the original paper discussing the method [21]. The highest precision and recall metrics using our 2-layered neural network model was 0.86 and 0.22 respectively (precision–recall plot not shown). The recall is a bit less when compared to the baseline model, but the overall precision is higher, and some of the desired mesh terms for a given abstract have much larger probabilities to be ranked higher.

5.2 NL–triple translation results

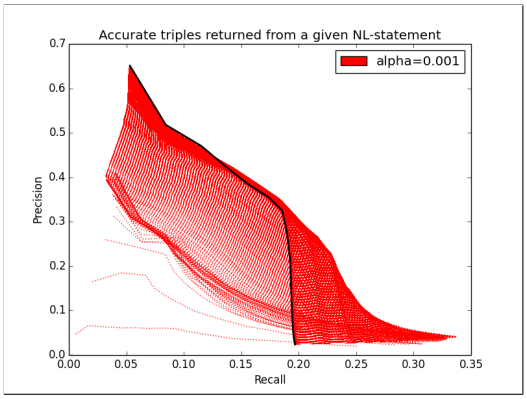

Figure 3: Precision–Recall curve for $\alpha = 0.001$

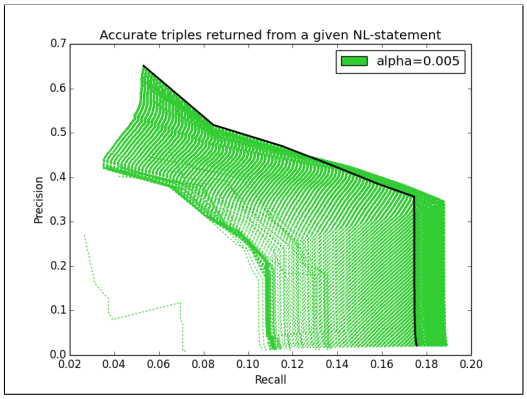

Figure 4: Precision–Recall Curve for $\alpha = 0.005$

We evaluated our neural network architecture using two learning rates, and the other hyper-parameters were fixed. DrugBank knowledge base has more than 6.5 million triples. We generated the verbal representation of 5,000 sets of triples for the training datasets (i.e. 5,000 NL-statements), 1,000 sets of triples for the validation and the testing datasets respectively. Each such set of triple may have a combination of 1-20 triples (including NULL triples). During the prediction stage, each parent node in the predicted syntax tree through the RNN should output one triple pattern vector after LSTM activation. We determine the proximity of the output triple pattern vectors to k-nearest triple pattern vectors in the DrugBank repository, to generate precision–recall curves (e.g. number of accurate triples at $k = 10$ triples output at each parent node in the syntax tree).

The precision–recall curves for the two learning rates are shown in **Figure 5.2**. Each line indicates the precision and recall metrics on training 10 batches of training samples. It can be seen that the precision and recall increases as the number of epoch increases, however as the training increases, the highest recall value actually decreases. However, the recall value at the highest precision value actually increases. The black line indicates the precision and recall curve for the test dataset. At $\alpha = 0.005$ we obtain a slightly higher precision, and the recall value at this higher precision measure ($k = 1$) is also somewhat higher. Moreover, even though the recall is somewhat lower than the first learning rate $\alpha = 0.001$, the rate of "recall decrease" is also lower (lesser width). The maximum

precision+recall for both the plots are around 0.6–0.65 and 0.07 respectively, though area under the curve is higher for $\alpha = 0.005$. We must try to evaluate the method for higher values of α.

6 Conclusion and Future Work

In this work, we have demonstrated a small application that leverages word embeddings, triple pattern embeddings and a neural network for NL–triple translation. Our neural network is inspired by combining Recurrent Neural Networks with Long Short-term memory activation units, and then using a root mean squared error loss function to predict the triple pattern vectors. In this work, we have not evaluated the entity embeddings and the relation hyperplanes and translation vectors of the RDF graphs. There are intrinsic and extrinsic evaluation methods to evaluate these triple pattern embeddings (knowledge graph completion, etc.), that can be explored in the future. Moreover, our method to generate these triple pattern embeddings, assume that the entity and the relation vectors should be in the same vector space. Newer methods, such as TransR [16], generate two separate vector spaces $\mathbb{E}$ and $\mathbb{R}$, and these methods will be explored in the future.

In the future, we will generate word embeddings by running GloVe for different values of the hyper-parameters, especially more number of iterations, and observe whether the clusters of biomedical entities can further be separated. It can be commented that such an architecture may not be scalable across the entire LSLOD cloud, as we will have to generate such triple pattern embedding vectors for every datasource that is currently integrated in the LSLOD cloud. The total number of triples in the LSLOD cloud are more than 1 trillion. Efficient methods to determine vector sub-spaces should be explored. We will also like to evaluate the method for a larger training set with different values of hyper-parameters to see if the precision–recall metrics are better. Moreover, this method should be evaluated for two or more RDF knowledge bases in the LSLOD cloud.

Finally, using word embeddings to generate MESH term recommendations is an interesting and a novel challenge, and we will definitely like to explore this avenue further. In this work, we have used a simple 2-layer recurrent neural network for the extrinsic evaluation, however other kinds of neural networks like Recurrent Neural network can be used for this problem. Instead of simple baselines, the new method should also be compared with polylingual topic models. In the future, we may want to train the entire architecture in an end–to–end fashion, instead of training GloVe, TransH and then the neural network architecture separately. Also, through an empirical analysis of the tokenized MEDLINE corpus, we found that the PubTator API was not able to annotate some chemical and gene names. Given the size of the LSLOD cloud, an end–to–end architecture, that learns the word embeddings and triple pattern embeddings, determines the named entities and performs NL–triple, or NL–SPARQL query translation can be established.

Acknowledgments

I would like to acknowledge CS 224D staff and Dr. Mark Musen.

References

[1] Tim Berners-Lee and James Hendler. Publishing on the semantic web. *Nature*, 410(6832):1023–1024, 2001.

[2] Alison Callahan, José Cruz-Toledo, Peter Ansell, and Michel Dumontier. Bio2rdf release 2: Improved coverage, interoperability and provenance of life science linked data. In *The semantic web: semantics and big data*, pages 200–212. Springer, 2013.

[3] Simon Jupp, James Malone, Jerven Bolleman, Marco Brandizi, Mark Davies, Leyla Garcia, Anna Gaulton, Sebastien Gehant, Camille Laibe, Nicole Redaschi, et al. The ebi rdf platform: linked open data for the life sciences. *Bioinformatics*, 30(9):1338–1339, 2014.

[4] Muhammad Saleem, Maulik R Kamdar, Aftab Iqbal, Shanmukha Sampath, Helena F Deus, and Axel-Cyrille Ngonga Ngomo. Big linked cancer data: Integrating linked tcga and pubmed. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27:34–41, 2014.

[5] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[6] Maulik R Kamdar, Dimitris Zeginis, Ali Hasnain, Stefan Decker, and Helena F Deus. Reveald: A user-driven domain-specific interactive search platform for biomedical research. *Journal of biomedical informatics*, 47:112–130, 2014.

[7] André Freitas, Edward Curry, João Gabriel Oliveira, and Seán O Riain. Querying heterogeneous datasets on the linked data web: Challenges, approaches, and trends. *Internet Computing, IEEE*, 16(1):24–33, 2012.

[8] Cleo Condoravdi, Kyle Richardson, Vishal Sikka, Asuman Suenbuel, and Richard Waldinger. Natural language access to data: It takes common sense! In *2015 AAAI Spring Symposium Series*, 2015.

[9] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Template-based question answering over rdf data. In *Proceedings of the 21st international conference on World Wide Web*, pages 639–648. ACM, 2012.

[10] Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. Neural enquirer: Learning to query tables. *arXiv preprint arXiv:1512.00965*, 2015.

[11] Arvind Neelakantan, Quoc V Le, and Ilya Sutskever. Neural programmer: Inducing latent programs with gradient descent. *arXiv preprint arXiv:1511.04834*, 2015.

[12] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[13] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data*. Springer, 2007.

[14] Michael Ashburner, Catherine A Ball, Judith A Blake, David Botstein, Heather Butler, J Michael Cherry, Allan P Davis, Kara Dolinski, Selina S Dwight, Janan T Eppig, et al. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

[15] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119. Citeseer, 2014.

[16] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.

[17] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934, 2013.

[18] Cornelis Koster, Marc Seutter, and Olaf Seibert. Parsing the medline corpus. *Proceedings RANLP 2007*, pages 325–329, 2007.

[19] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. Pubtator: a web-based text mining tool for assisting biocuration. *Nucleic acids research*, page gkt441, 2013.

[20] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795, 2013.

[21] Minlie Huang, Aurélie Névéol, and Zhiyong Lu. Recommending mesh terms for annotating biomedical articles. *Journal of the American Medical Informatics Association*, 18(5):660–667, 2011.

[22] David S Wishart, Craig Knox, An Chi Guo, Dean Cheng, Savita Shrivastava, Dan Tzur, Bijaya Gautam, and Murtaza Hassanali. Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic acids research*, 36(suppl 1):D901–D906, 2008.

[23] Xiantang Sun and Chris Mellish. Domain independent sentence generation from rdf representations for the semantic web. In *Combined Workshop on Language-Enabled Educational Technology and Development and Evaluation of Robust Spoken Dialogue Systems, European Conference on AI, Riva del Garda, Italy*, 2006.