Knowledge Graph Completion
TransE vs TransH

Omar Sanchez
Osanchez@umass.edu

1 Problem statement

The goal of this project is to replicate and show the effectiveness of translating on hyperplanes (TransH) over translating embeddings (TransE) for knowledge graph completion as seen in (Lin et al., 2015). Knowledge graph completion involves link prediction tasks which aim at predicting relations between entities under supervision of the existing knowledge graph.

2 Introduction

What is a knowledge graph?
There are many different definitions online explaining what a knowledge graph is. A more generalized definition would describe a knowledge graph as being a graph of semantic relationships. To better understand what a knowledge graph is, we can visualize a graph where each point represents an entity. An entity can represent anything from a person, to a place, or a thing. A knowledge graph can be composed from hundreds to even millions of entities. Not only can an entity represent any of the things listed above, but each entity also contains information about itself. Things such as birthday, location, color, or even a small blurb of text can be contained within an entity. Each entity in a knowledge graph is unique (or so we hope). Alongside entities in a knowledge graph lines that connect these entities to one other. But these entities are not connected randomly, they are connected semantically, or in other words, connected by a common relation that they share. These lines are referred to as relationships.

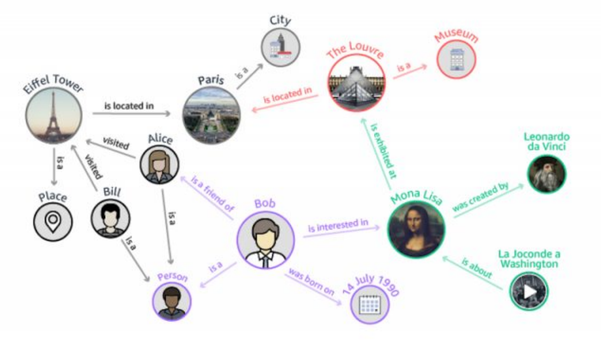
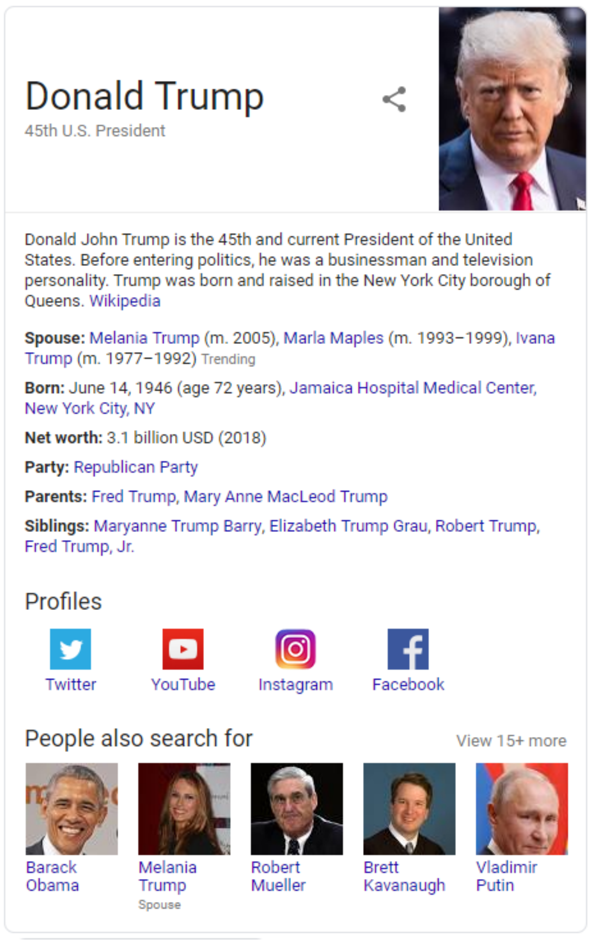


Figure 1: An example knowledge graph showing entities and entity relationships

What is the importance of a knowledge graphs, why is it useful?
You may not realize it, but if you are a user of Googles search engine or many AI assistance such as Google assistant or Siri, then you have some experience using a knowledge graph. Information that is queried using Googles search engine can retrieve results from a knowledge graph. This can be seen when information is presented to us in an info box next to search results. This is information that is being retrieved from Googles personal knowledge graph.

How does a knowledge graph work?
To answer that question, lets first look at


Donald Trump
45th U.S. President

Donald John Trump is the 45th and current President of the United States. Before entering politics, he was a businessman and television personality. Trump was born and raised in the New York City borough of Queens. Wikipedia

Spouse: Melania Trump (m. 2005), Marla Maples (m. 1993–1999), Ivana Trump (m. 1977–1992) Trending

Born: June 14, 1946 (age 72 years), Jamaica Hospital Medical Center, New York City, NY

Net worth: 3.1 billion USD (2018)

Party: Republican Party

Parents: Fred Trump, Mary Anne MacLeod Trump

Siblings: Maryanne Trump Barry, Elizabeth Trump Grau, Robert Trump, Fred Trump, Jr.

Profiles

Twitter YouTube Instagram Facebook

People also search for View 15+ more

Barack Obama
Melania Trump Spouse
Robert Mueller
Brett Kavanaugh
Vladimir Putin

Figure 2: An example of information displayed in an info box when a query is made on Google search engine

where information that is kept in a knowledge graph comes from. Googles first knowledge graph was constructed from a knowledge base known as Freebase. This knowledge base, similar to Wikipedia, is a place where large amounts of structured data is stored. Google collected information from this site, because it not only included many different entities, but was as close to consistent they could get for entities with listed attributes. A knowledge graph works by taking information found in websites like Freebase or Wikipedia, and without going much into detail, converting it into a graph that can be searched to find information on entities. Not only is information on these entities retrieved, but other entities similar to the one queried are also returned. We see these associations in Wikipedia as blue hyperlinks within a page, or as links such as those seen in the bottom of the info box in figure 2.

What is knowledge graph completion? Knowledge graph completion is an intuitive way of doing data cleaning in existing knowledge graphs. Because knowledge bases like Freebase and Wikipedia are publicly sourced and are so large, any errors in these knowledge bases are projected onto knowledge graphs. Knowledge graph completion aims to fix these errors. Some useful examples of data cleaning are removal of duplicate entities and filling in missing values. These are done to a knowledge graph to improve their efficiency, and correctness. This brings us into the topic of this paper, TransE and TransH. These translation models are used to convert a knowledge graph into embeddings in low dimensional space in order to automate the process of knowledge graph completion by looking at entities and the relationships as vectors in order to create new, and correct, relationships between entities.

3 The models

TransE

Manipulating a large knowledge graph is a challenging task. To address scalability and performance issues, a solution that embeds knowledge graph components (entities and relations) in a low dimensional continuous vector space is given. This is said to preserve some of the properties of the original graph. In TransE, relationships are represented as translations in the embedding space, that is, if a relation holds, then the embedding of one entity should be close to the embedding of another entity plus some vector representing the relationship. Let's say we have 2 entities, entity 1 and entity 2. We want the learned embedding of entity 1 + some relation entity 2 for a given fact in the Knowledge graph. We also want entity 1 + some other relation to be far away from entity 2 if the fact is wrong. It is important to note that these entities and relational vectors are within the same embedding space. This is where a major issue in TransE arises. Once the model has learned an embedding vector for each entity and relation, predictions will be performed using the same translation approach in the embedding space.

Embeddings are learned over several iterations, in our approach our model is trained for 500 epochs. The training algorithm tries to find a vector configuration for entities and relationships where the error of translation of a true fact is smaller than the error of translation of a wrong fact by a given value. The error of translation is called dissimilarity measure of a knowledge graph fact. In order to generate wrong facts, the system corrupts either the head or the tail of a relational triplet (h,r,t) (h and t are entities and r is a relation) existing in the knowledge graph with another randomly selected entity. After all entities have gone through a random initialization stage to avoid cases where embedding are equal to 0, the model is trained. Each learning iteration, called an epoch, starts with normalization of the embeddings. the algorithm then iterates over a portion of the knowledge graph facts (training set) and performs an optimization task. The learning process stops based on the model performance on a portion of the knowledge graph facts left apart (validation set).

TransH TransH is similar to TransE, entities and relationships are in fact represented in in vector spaces, however, one major issue with TransE, are it's flaws when dealing with relations with mapping properties of reflexive/one-to-many/many-to-one/many-to-many. In order to preserve these relational properties TransH distributes relationships for each relational property along its own hyperplane. As illustrated in Figure 3, for a relation r, we position the relation-specific translation vector d_r in the relation-specific hyperplane w_r (the normal vector) rather than in the same space of entity embeddings. For a triplet (h, r, t), the embedding h and t are first projected to the hyperplane w_r. The projections are denoted as $h_\perp$ and $t_\perp$, respectively. We expect $h_\perp$ and $t_\perp$ to be connected by a translation vector d_r on the hyperplane with low error if (h, r, t) is a golden triplet. A golden triple is defined as a triple that is correct in terms of wordly facts.

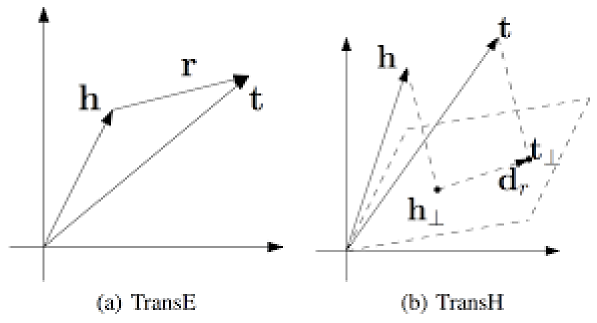


Figure 3: Visualization of TransE and TransH in their corresponding planes

4 What we proposed vs. what we accomplished

The original project proposal aimed to create a knowledge graph for a given context that could be given words or phrases as queries and would return relevant information in a synthesis of question answering and summarizing. When we set off to create a knowledge graph, we had difficulty finding resources to get us started. Many Google searches, research papers, and textbooks later, we came to the conclusion that creating a knowledge graph was not as simple as we thought it would be. A major factor of the change in our proposal was the time frame we had to complete this project. With our current knowledge in NLP, creating a knowledge graph from scratch would require a vast amount of time from each member of the team, which was already smaller than hoped for. After changing our focus from creating a knowledge graph from scratch to learning more about knowledge graphs, we came across knowledge graph completion and proposed models that aimed to find a more efficient solution to knowledge graph completion. In this process we looked at various models including TransE, TransH for knowledge graph completion. We specifically focused on link prediction tasks for each model to determine if the results we received were similar to those proposed in the works of Lin et al. (2015). We were able to successfully train and test a TransE model using their source code. We were also able to Train a TransH model, but code provided only tested the TransE model. Looking for a solution to this issue we stumbled across another git-hub repository created by members of the research team. This repository contained a faster more flexible framework for training each model, and there indeed was a test for the TransH model. However, we were unable to complete the testing using this new framework as it required the ¡sys/mman.h¿ Unix header and it was not available on Windows. A Linux environment was setup for testing, but due to time constraints we were unable to get a working environment due to difficulty in meeting required dependencies.

5 Related work

Other approaches on TransE (Bordes et al., 2013) describe the method as an energy-based model for learning low-dimensional embeddings of entities. Bordes states that the relationships in TransE are represented as translations in the embedding space, saying that if (h, r, 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 r. Their approach relies on a reduced set of parameters as it learns only one low-dimensional vector for each entity and each relationship. Their version of the TransE model is compared to similar models including an unstructured version of TransE (Bordes et al., 2011) which considers data as mono-relational and sets all translations to 0. A Rescal model (Nickel et al., 2011) which is trained by an alternating least-squares method along with other models SE, SME(linear), SME(bilinear) (Bordes et al., 2011) and LFM (Jenatton et al., 2012). These models differ from the version of TransE proposed which uses stochastic gradient descent for training. Performance between models are compared using a series of link prediction tasks. Other researchers (Wang et al., 2014) found that while TransE did provide state of the art predictive performance, it had flaws when dealing with relations with mapping properties of reflexive/one-to-many/many-to-one/many-to-many. Other, more advanced models, try to overcome this flaw by including free parameters to preserve these mapping properties, however, the model complexity and running time is significantly increased. These advanced models also performed worse than TransE (Bordes et al., 2013).

To overcome the issues of TransE a new model is proposed TransH (Wang et al., 2014) which interprets a relation as a translating operation on a hyperplane. In TransH, each relation is characterized by two vectors, the norm vector w_r of the hyperplane, and the translation vector d_r on the hyperplane. For a golden triplet (h, r, t); a triplet that is correct in terms of worldly facts, the projections of h and t on the hyperplane are expected to be connected by the translation vector d_r with low error. This method is said to overcome the flaws of TransE in dealing with different relational properties. The same link prediction tasks from TransE are used to test the effectiveness of TransH vs TransE. In our research we try to replicate the results for each of these models and determine whether or not TransH is a better training candidate than transE.

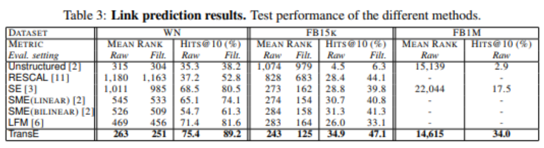


Table 3: **Link prediction results.** Test performance of the different methods.

Dataset	WN				FB15k				FB1M	
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)		Mean Rank	Hits@10 (%)
Eval. setting	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Raw*
Unstructured [2]	315	304	35.3	38.2	1,074	979	4.5	6.3	15,139	2.9
RESCAL [11]	1,180	1,163	37.2	52.8	828	683	28.4	44.1	-	-
SE [3]	1,011	985	68.5	80.5	273	162	28.8	39.8	22,044	17.5
SME(linear) [2]	545	533	65.1	74.1	274	154	30.7	40.8	-	-
SME(bilinear) [2]	526	509	54.7	61.3	284	158	31.3	41.3	-	-
LFM [6]	469	456	71.4	81.6	283	164	26.0	33.1	-	-
TransE	**263**	**251**	**75.4**	**89.2**	**243**	**125**	**34.9**	**47.1**	**14,615**	**34.0**

Figure 4: TransE baseline used in research conducted in (Bordes et al., 2011)

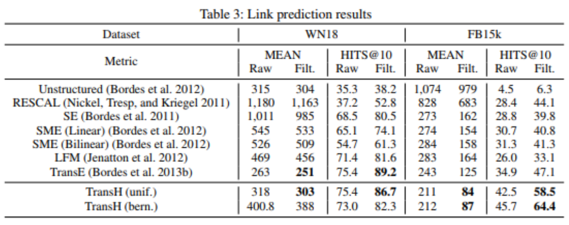


Table 3: Link prediction results

Dataset	WN18				FB15k			
Metric	MEAN		HITS@10		MEAN		HITS@10	
	Raw	Filt.	Raw	Filt.	Raw	Filt.	Raw	Filt.
Unstructured (Bordes et al. 2012)	315	304	35.3	38.2	1,074	979	4.5	6.3
RESCAL (Nickel, Tresp, and Kriegel 2011)	1,180	1,163	37.2	52.8	828	683	28.4	44.1
SE (Bordes et al. 2011)	1,011	985	68.5	80.5	273	162	28.8	39.8
SME (Linear) (Bordes et al. 2012)	545	533	65.1	74.1	274	154	30.7	40.8
SME (Bilinear) (Bordes et al. 2012)	526	509	54.7	61.3	284	158	31.3	41.3
LFM (Jenatton et al. 2012)	469	456	71.4	81.6	283	164	26.0	33.1
TransE (Bordes et al. 2013b)	263	**251**	75.4	**89.2**	243	125	34.9	47.1
TransH (unif.)	318	**303**	75.4	**86.7**	211	**84**	42.5	**58.5**
TransH (bern.)	400.8	388	73.0	82.3	212	**87**	45.7	**64.4**

Figure 5: TransH baseline used in research conducted in (Wang et al., 2014)

6 Dataset

The FB15k dataset was used to train our TransE and TransH models. FB15k is a relatively dense sub-graph of Freebase with approximately 15,000 entities, and 1,345 relationships. Our FB15K dataset was acquired from the following github repository `https://github.com/thunlp/KB2E`.

6.1 Data preprocessing

While much of the data did not require any pre-processing, we did run into an issue where we were unable to interpret some of the results in our testing phase. Freebase represents entities using special IDs. In 2014 Freebase shutdown and its data was migrated to wikidata. APIs that used the freebase API to interpret information on machine IDs had become deprecated. Google later announced a new API that could be used to retrieve this information. The code that our research had been based off was originally based off of this freebase API so a workaround was created using Googles new API. With this we could further analyze our test results to ensure we were getting plausible results.

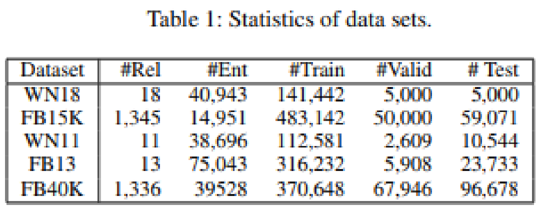


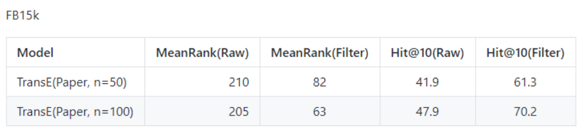
Table 1: Statistics of data sets.

Dataset	#Rel	#Ent	#Train	#Valid	# Test
WN18	18	40,943	141,442	5,000	5,000
FB15K	1,345	14,951	483,142	50,000	59,071
WN11	11	38,696	112,581	2,609	10,544
FB13	13	75,043	316,232	5,908	23,733
FB40K	1,336	39528	370,648	67,946	96,678

Figure 6: dataset information on the FB15K data set compared to other sets

7 Baselines

The best configuration is determined according to the mean rank in validation set according to Wang et al. (2014). Their optimal configurations for TransE and TransH for embedding dimensions k, d = 50, the learning rate = 0.001, the margin = 1.0, B = 960, and dissimilarity metric as L1. The training data in their evaluation is traversed for 500 epochs. We use the same hyperparametrs listed above to try to replicate the results of (Lin et al., 2015), who used these hyperparemeters as well. since we use the same datasets and code as (Lin et al., 2015) to train each model their results are our baseline. The source to their code can be found at `https://github.com/thunlp/KB2E`. According to them, the freebase part of their data was built by themselves, different from the ones used by Wang et al. (2014). Because of this, evaluation results cannot be compared directly with those reported in Wang et al. (2014). Hence, our implementation of TransE and TransH must be compared to those in (Lin et al., 2015).



FB15k

Model	MeanRank(Raw)	MeanRank(Filter)	Hit@10(Raw)	Hit@10(Filter)
TransE(Paper, n=50)	210	82	41.9	61.3
TransE(Paper, n=100)	205	63	47.9	70.2

Figure 7: Baseline that we will compare our results to from (Lin et al., 2015)

8 Approach

Link prediction predicts a missing head or the tail entity given a relation type and a single entity, for example, given a correct hrt triplet (h, r, t). We corrupt the triple by removing either the head or tail entity creating a new triplet in the form (?, r, t) or (h, r, ?), where ? denotes the missing entity. For all test triplets, the head and tail entities are replaced with all possible entities in the knowledge graph and ranked in descending order of similarity scores calculated by score function fr.

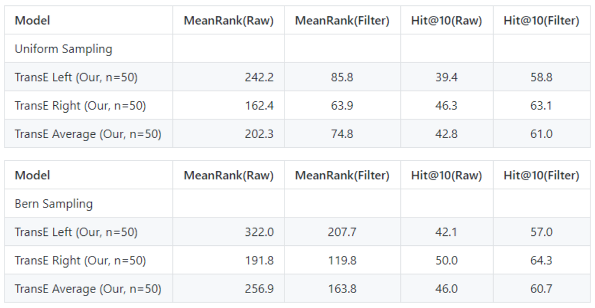
$$f_r(h, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$$

Figure 8: TransE Score Function

$$f_r(h, t) = \|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2.$$

Figure 9: TransH Score Function

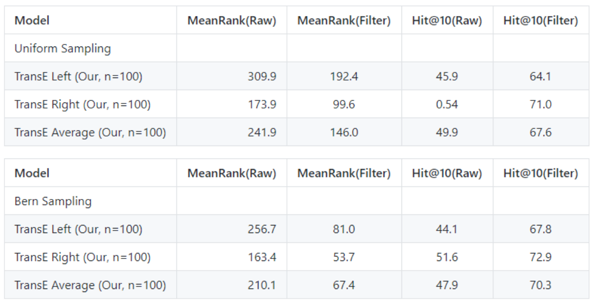
We use this approach to evaluate each model on its ability to predict the missing entity. This score is called the Raw score. Included is also a Filtered score, described in (Bordes et al., 2013), are the scores after filtering out corrupted triples that already existed in the dataset. According to Lin et al. (2015), ranking a corrupted triple appearing in the dataset higher than the original test triple is correct, but is penalized by the Raw score, thus the Filtered setting provides a clearer view on the ranking performance. A good link predictor should achieve lower mean rank or higher Hits@10. figures 9 and 10 show our results attained from link prediction using our replicated TransE model. One thing that was noticed when collecting data is the paper fails to explain that the mean rank and hits @10 calculated are an average of both left and right sampling methods. As previously mentioned, when a triplet is corrupted either its head or tail entity is removed. calculated scores are associated with what side of the triplet was removed, both left and right ranking are given. In (Lin et al., 2015), rankings our outputted using only a single score. To resolve this we simply take the average of both sides to get a combined score. While are results were not exact we were able replicate the link prediction results within a small margin.Differences in scores we believe are a effect of the sampling methods used. These sampling methods are Uniform sampling and Bernoulli sampling. Uniform Sampling makes a random probability replacement of head and tail entities. Bernoulli Sampling when given a golden triplet of the relation r with probability tph/tph + hpt and hpt/tph + hpt, corrupts a triplet by replacing both its head and tail entities. the randomness involved in both of these tasks we believe result in the small differences in our results.



Model	MeanRank(Raw)	MeanRank(Filter)	Hit@10(Raw)	Hit@10(Filter)
Uniform Sampling				
TransE Left (Our, n=50)	242.2	85.8	39.4	58.8
TransE Right (Our, n=50)	162.4	63.9	46.3	63.1
TransE Average (Our, n=50)	202.3	74.8	42.8	61.0

Model	MeanRank(Raw)	MeanRank(Filter)	Hit@10(Raw)	Hit@10(Filter)
Bern Sampling				
TransE Left (Our, n=50)	322.0	207.7	42.1	57.0
TransE Right (Our, n=50)	191.8	119.8	50.0	64.3
TransE Average (Our, n=50)	256.9	163.8	46.0	60.7

Figure 10: TransE link prediction results using uniform and Bernoulli sampling. Hyperparemeter of size set to 50



Model	MeanRank(Raw)	MeanRank(Filter)	Hit@10(Raw)	Hit@10(Filter)
Uniform Sampling				
TransE Left (Our, n=100)	309.9	192.4	45.9	64.1
TransE Right (Our, n=100)	173.9	99.6	0.54	71.0
TransE Average (Our, n=100)	241.9	146.0	49.9	67.6

Model	MeanRank(Raw)	MeanRank(Filter)	Hit@10(Raw)	Hit@10(Filter)
Bern Sampling				
TransE Left (Our, n=100)	256.7	81.0	44.1	67.8
TransE Right (Our, n=100)	163.4	53.7	51.6	72.9
TransE Average (Our, n=100)	210.1	67.4	47.9	70.3

Figure 11: TransE link prediction results using uniform and Bernoulli sampling. Hyperparemeter of size set to 100

9 Error analysis

Our results were compared to findings from the research done by (Lin et al., 2015). In the process of gathering data and analyzing our results, we came across a research paper that stated, The WN18 and FB15k datasets suffer from test set leakage, due to inverse relations from the training set being present in the test set (Dettmers et al., 2017). Because of this we feel that the evaluation results presented in this paper and other papers should be reevaluated. We have noticed in NLP is that choosing a valid dataset is not an easy task.

10 Contributions of group members

- Cyrus Freshman - Responsible for determining initial dataset in project proposal which was later scrapped for the FB15k dataset.
- Vidul Mahendru - Helped with poster ”presentation”.
- Omar Sanchez - Data collection, trained models, tested models, error analysis, wrote project proposal, progress report, created project poster, and wrote final report.

11 Conclusion

Although the extent of the issue for the FB15K dataset has not been quantified, it is apparent that this invalidates our results and I hope to collect new data on the new proposed data sets that have been reduced, FB15k-237 and WN18RR. Our original proposal of creating a knowledge graph from scratch is still something I want to do. NLP is a rapidly evolving subject, one method of approach towards a task always has a better solution and it is up to the developer to find this method. When starting this project I looked at the topic of knowledge graphs as a stand alone topic. something that could be queried into Google search engine and have a result returned immediately. Now I see that this is a naive idea. Knowledge graphs are not a single topic, knowledge graph are created using many different topics in NLP such as translations and embeddings, to even data collection which is not an easy task as we saw in our error analysis. With that said, I hope to continue my journey in learning more about NLP as it seems like that is the era we are in.

References

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Bordes, A., Weston, J., Collobert, R., and Bengio, Y. (2011). Learning structured embeddings of knowledge bases. In *AAAI*.

Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. (2017). Convolutional 2d knowledge graph embeddings. *arXiv preprint arXiv:1707.01476*.

Jenatton, R., Roux, N. L., Bordes, A., and Obozinski, G. (2012). A latent factor model for highly multi-relational data. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'12, pages 3167–3175, USA. Curran Associates Inc.

Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2181–2187. AAAI Press.

Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, USA. Omnipress.

Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119. AAAI Press.