Project Report: Emerging Relation Detection from News in Heterogeneous Information Networks(HIN)

Eric Dang
erickdang@ucla.edu

Noor Nakhaei
noornk@ucla.edu

Hui Wang
logicvay2010@ucla.edu

Wenhao Zhang
wenhaoz@cs.ucla.edu

1 ABSTRACT

Real world knowledge is dynamically growing with new people and events are rapidly appearing in every-day news. These new names cause the appearance of new relations, as a result, the global knowledge graph (KG) needs to be updated to reflect these emerging relations. In this project we focus on finding the new relations between people, locations, and organizations, all of which we refer to as entities. In order to implement the emerging relation detection algorithm, we need to overcome a few challenges: 1) limited initial information makes it hard to use traditional sentence-based models, 2) no negative relations exist in the KG, and 3) the rate of appearance of the new entities and relation is high and the KG needs to be updated rapidly. Our plan to construct the KG and detect emerging relations begins with a heterogeneous textual graph, built from joint occurrences of words in the news, to which we apply a heterogeneous graph embedding framework. This framework embeds words into a vector representation based on their co-occurrences and learns a classifier on these embedding to predict the probability of two entities being an emerging relation.

2 INTRODUCTION

An emerging relation is a new connection in the KG between two entities that aren't already connected in the KG. Entities are defined as persons, locations, and organizations, so the new relation could be a family connection of two people or a person owning a organization. Extracting relations of the entities from news, or more generally from text, is a hot topic in natural language processing and information retrieval. Different approaches have been proposed during the years to extract relations, from locally mining the sentences to find the relations [2, 4, 5], to performing link prediction in graphs, and random walk inference techniques [1, 3]. In this paper we use heterogeneous graphs and word embedding to show the entities and relations between them. For extracting the emerging relations between entities from the news, we have to deal with a large volume of data that's being updated constantly. Predicting emerging relations is applicable to news related tasks, such as information retrieval, news ranking, and event detection[HEER paper]. To extract the emerging relation, we need to overcome three main challenges:

- Only using sentence-based models will cause problems due to the limited initial information.
- Due to the lack of negative relations, we need a way to extract negative relations from the KG. These extracted negative relations could be false negatives [4] in reality, which introduces noise to our framework.
- New relations are emerging rapidly and it's important to keep the KG up to date.

Thus, a novel Heterogeneous graph Embedding framework for Emerging Relation detection (HEER) will be used to over come the challenges. Figure 2 explains the intuition of how the HEER algorithm works in detecting emerging relations. We first construct the Heterogeneous Textual Graph from the news, where the nodes are entities and contextual words, with the edges being the co-occurrence of the nodes in a sentence. All the words extracted from the news are categorized into entities or contextual words. Entities are the words recognized as either a person, location, or organization by the tokenizer, and the rest being contextual words. We use both the KG and the Heterogeneous Textual Graph for predicting the emerging relations, by applying a classifier on the jointly learned word embedding from the news. Our approach could be summarized as following:

- Construct the Heterogeneous Textual Graph and KG.
- Extract the joint embedding of entities and words using a gradient descent based method.
- Learn a classifier on the word embedding to predict the emerging relations and adding them to the KG.

In summary, our ultimate goal is detecting the emerging relations precisely. The rest of the report is organized as follows: Section 3 formulates the problems, Section 4 describe the data and preprocessing methods, Section 5 introduces the details of the proposed HEER framework, Sections 6 is the experimental design and results, Section 7 is a brief review of related word, and Section 8 is the conclusion.

3 PRELIMINARY

Before we dig into the details of each method, we give the related concepts and notations.

Definition 1 Entity and relation: In our project, an entity, denoted as e_i, can be one of the following: a person, an organization, or a location. And we use the label $y \in \{0, 1\}$ to denote whether a link (relation) exists between the entity-entity pair (e_i, e_j) in actuality.

Definition 2 Knowledge Graph (KG): The KG is denoted as $G_{kg} = (E_{kg}, \epsilon_{kg})$ where E_{kg} is the set of entities, and ϵ_{kg} is the set of relations. We have the label $z \in 0, 1$ to denote whether a relation pair exists in G_{kg}, i.e. $z = 1$ if $(e_i, e_j) \in \epsilon_{kg}$.

As mentioned, current knowledge databases are not up to date, so there may be an entity with label $z = 0$ while it's y label is 1. Thus, relations that are not present in the KG does not imply that the relation does not exist in actuality. As a result, we cannot say

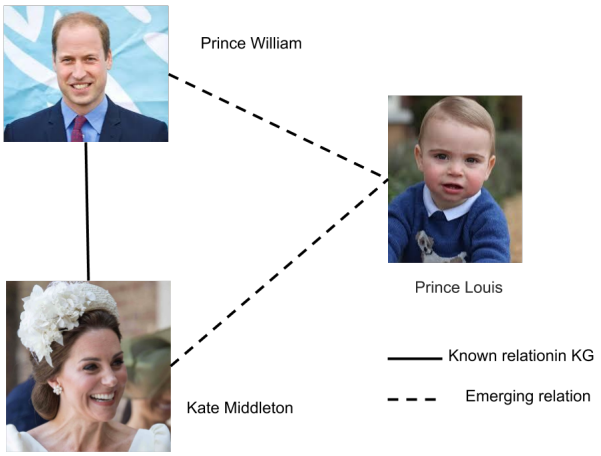

Figure 1: An example of emerging relation

that entity-entity pairs with $z = 0$ are negative relations. Rather we refer to them as unlabeled relations.

Definition 3 Emerging Relation: A relation, with label $y = 1$ is a relation that exists in reality, which is the news in our case. This type of relation may or may not exist in the KG. If it does not exist in the KG, then we hope that our algorithm detects this as an emerging relation.

Definition 4 Heterogeneous Textual Graph (HTG): A heterogeneous textual graph is represented as an undirected graph $G_{news} = (V_{news}, \epsilon_{news})$, where V_{news} is the set of nodes including entities E_{news} and contextual words C_{news}. $\epsilon_{news} \subseteq V_{news} X \epsilon_{news}$. The weight of the edges are determined by the frequencies of co-occurrences.

Heterogeneous Textual Graph encodes the information from news into a graph, from where we will learn the emerging relations.

4 METHODS

In this section, we first discuss the construction of the Heterogeneous Textual Graphs (e.g. KG, entity-entity graph, word-word graph, entity-word graph) from the BBC news corpus. Then, we learn the joint embedding of entity-entity, entity-word, word-word from these HTGs. And lastly, we detect emerging relations.

4.1 Constructing Heterogeneous Textual Graphs from News

In this project, a subset of news articles from the political category in the BBC news corpus were sampled considering the fact that we have limited time to finish the experiment. Prior to constructing these HTGs, we tokenized the news articles and removed the stop words. Then we annotated the remaining words using the Stanford Named Entity Recognizer(NER) tool. Words labeled with "PERSON", "LOCATION", and "ORGANIZATION" are entities. The remaining uni-gram words are contextual words. When constructing the Heterogeneous Textual Graphs, we take entities or contextual words as nodes. Edges are the co-occurrences of any two words within every 5-word sliding window in news sentences. The frequency of a co-occurrence link was deemed as its weights.

In order to model the relations between entity-entity, entity-word, word-word, we further divide the Heterogeneous Textual Graph into several sub-graphs: 1) Knowledge Graph, 2) Entity-Entity Graph, 3) Word-Word Graph, 4) Entity-Word Graph. Knowledge Graph only contains entities that are known to DBpedia. Entities that failed to match DBpedia are considered as new entities. For simplicity, we randomly sampled some entities from the knowledge graph and treat them as new entities in this project. Entity-Entity graph are comprised of the known entities and new entities. Word-Word graph are comprised of all the contextual words. Entity-Word graph is a bipartite graph which represents the relations between entities and their surrounding contextual words.

4.2 Joint Embedding of the News and the KG

Up until this point, we have the KG, and the HTG, which is comprised of the three following sub-graphs entity-entity, word-word, and entity-word.

Having these graphs are enough for training our learning model, but it takes a significant amount of time since there are a large number of words in the news. We use both the KG and the HTG to make the embedding. We use a hyper parameter θ to determine which graph, the KG or HTG, we are relying on more to embed each word, which will be discussed in detail later on. To learn the joint embedding, we define a conditional probability for each co-occurrence of two words, which can either be both entities, contextual words, or a combination of the two. The conditional probability of the co-occurrence of an entity (e_i), having a word (c_j) is:

$$p(e_i|c_j) = \frac{e^{s_i^T t_j}}{\Sigma_{k=1}^{|E_{news}|} e^{s_k^T t_j}} \tag{1}$$

where S_i represents the joint embedding of the entity i, and T_j represents the joint embedding of the word j. This probability is calculated for all the entities in the news. Thus, for two entities, e_1 e_2, with the same neighboring words would have similar embedding. But because of the big number of contextual words and entities in the news, we try to calculate this probability for a small number of neighbors for each node(entity or contextual word), by defining negative edges. So for each node, we pick a positive weighted edge, and k negative edges, proportional to the edge's weight. For those $k + 1$ selected edges, we calculate the defined probability and use the following equation to calculate the cost.

$$J_{ec} = -\Sigma_{(e_i c_j) \in \epsilon_{ec}} w_{ij} log P(e_i|c_j) \tag{2}$$

If the selected nodes were contextual word-contextual word, or entity-entity, the cost would be in the same form:

$$J_{ee} = -\Sigma_{(e_i e_j) \in \epsilon_{ee}} w_{ij} log P(e_i|e_j) \tag{3}$$

$$J_{cc} = -\Sigma_{(c_i c_j) \in \epsilon_{cc}} w_{ij} log P(c_i|c_j) \tag{4}$$

And if the selected edge in the iteration is in the KG, the cost function is:

$$J_{kg} = -\Sigma_{(e_i e_j) \in \epsilon_{kg}} w_{ij} log P(e_i|e_j) \tag{5}$$

And the new cost function is:

$$J_{news} = J_{ec} + J_{ee} + J_{cc} \tag{6}$$

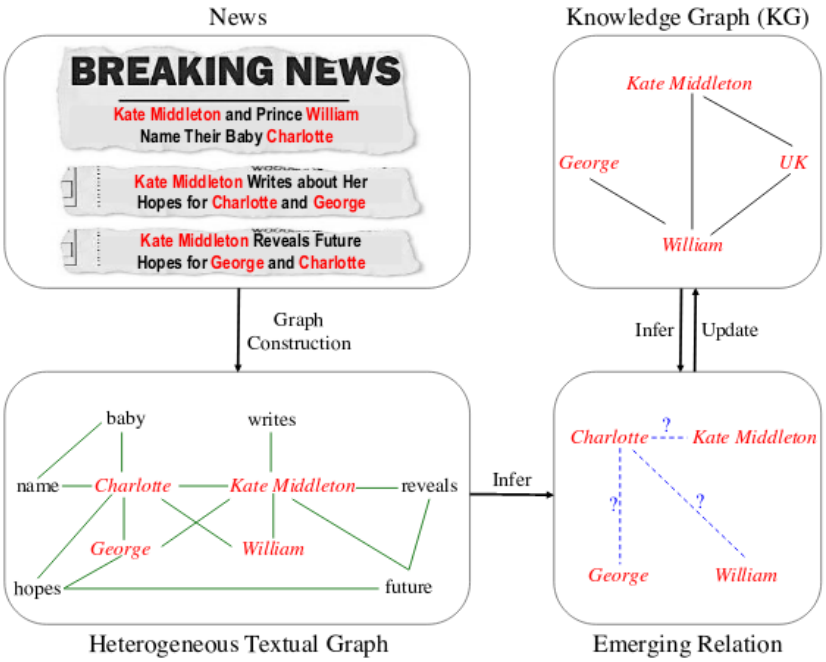

Figure 2: Detecting emerging relations by inferring from the heterogeneous textual graph and the KG. The entities are in red. The co-occurrence links in the heterogeneous textual graph are in green, and the relations in the KG are in black. Reprinted from [7]

The final total cost function is dependent upon the hyper parameter θ, which we introduced earlier. So the total cost function is:

$$J_{total} = (1-\theta)J_{news} + \theta J_{kg} \tag{7}$$

Parameter $\theta \in [0,1]$ is a guiding parameter that trades off between the contribution of the KG and HTG to the word embedding. As it can be seen from the equation, the bigger the θ is, the more the word embedding is reliant on the KG and a smaller θ means that the embedding is more reliant upon the HTG. Finally, we use gradient descent to optimize the total cost function.

So in brief, for each contextual word and entity we produce an embedding as a d-dimensional vector. Based on the algorithm in the paper, we randomly initialize the embedding, and try to optimize them based on the defined cost function using gradient descent.

We use algorithm 1 for implementing the joint embedding of news and KG graphs.

Algorithm 1 Joint embedding of the news and the KG

Input: The heterogeneous textual graph $\mathcal{G}_{news} = \mathcal{G}_{ec} \cup \mathcal{G}_{ee} \cup \mathcal{G}_{cc}$, the KG $\mathcal{G}_{kg}$, the guiding parameter θ, the number of negative samples k, and the number of embedding iterations T.
Output: Entity embeddings $\mathcal{S}$ and word embeddings $\mathcal{T}$.

```
Initialize entity embeddings S randomly
Initialize contextual word embeddings T randomly
while iter ≤ T do
    Generate a random number γ ∈ [0, 1]
    if γ ≤ θ then
        EMBEDDING UPDATE(S, S, G_kg, k)
    else
        EMBEDDING UPDATE(S, T, G_ec, k)
        EMBEDDING UPDATE(S, S, G_ee, k)
        EMBEDDING UPDATE(T, T, G_cc, k)
    end if
end while

function EMBEDDING UPDATE(S, T, G, k)
    Sample an edge from G and draw k negative edges
    Update node embeddings S and T
end function
```

4.3 Detecting Emerging Relations with Positive Cases Only

In order to begin detecting emerging relations, we must first combine the two vector embedding of each entity-entity pair. Any function that takes two vectors and outputs a single vector of the same dimensions will work, but for our case we chose to use the average of the two vectors.

$$\vec{x} = h(\vec{s_i}, \vec{s_j}) = \frac{1}{2}(\vec{s_i} + \vec{s_j}) \tag{8}$$

In the above equation, $\vec{s_i}$ and $\vec{s_j}$ represent the embedding of two entities, and $\vec{x}$ represents the joint embedding of the entity-entity pair. Applying this function to every entity-entity pair defines the set $\mathbf{X}$ containing all entity-entity pair embedding.

$$\mathbf{X} = \{\vec{x} \ : \ (e_i, e_j) \in \epsilon_{ee}\} \tag{9}$$

Next, we define the two sets $\mathbf{P}$ and $\mathbf{U}$, as the subsets of $\mathbf{X}$ containing positive labels and unlabeled entity-entity pairs. That is, $\mathbf{P}$ is the set of entity-entity pairs where both entities exist in the KG (i.e. $z = 1$), and the set $\mathbf{U}$ is the set of entity-entity pairs where at least

one entity does not exist in the KG (i.e. $z = 0$). An emerging relation must be from the set **U** where one entity in the pair is present in the KG and the other entity is not.

Because we know the KG labels of the entity-entity pairs but do not know the actual relationship labels (i.e. y), we want to train a classier, f, on KG labels in order to infer the actual relationship labels, y. To do this, we train a classifier, g, that is then adjusted to classifier f by a constant factor. Ultimately, we want to predict the following probability:

$$P(y = 1|\vec{x}) = \frac{P(z = 1|\vec{x})}{P(z = 1|y = 1)} \tag{10}$$

The classifier $g(\vec{x})$, will output $P(z = 1|\vec{x})$, which is the probability that an entity-entity pair will have positive KG label. The next step is estimate the denominator, $P(z = 1|y = 1)$, using the classifier $g(\vec{x})$. To do this, we assume that entity-entity pairs are chosen randomly from $\mathbf{P} \cup \mathbf{U}$ which allows us to make the following simplification:

$$P(z = 1|\vec{x}, y = 1) = P(z = 1|y = 1) \tag{11}$$

Thus, using the classifier $g(\vec{x})$, on a randomly selected subset of entity-entity pairs **S**, we can estimate $P(z = 1|y = 1)$ with the following equation:

$$P(z = 1|y = 1) \sim \epsilon = \frac{1}{|\mathbf{S_p}|} \sum_{\vec{x} \in \mathbf{S_p}} g(\vec{x}) \tag{12}$$

In the above equation, $\mathbf{S_p}$ is the subset of **S** which have positive KG labels. The value of ϵ becomes the constant factor which we use to adjust the classifier g to the classifier f. The following equation describes the transformation.

$$f(\vec{x}) = P(y = 1|\vec{x}) = \frac{g(\vec{x})}{\epsilon} \tag{13}$$

Lastly, we must define a threshold probability that will define whether the entity-entity pair defines an emerging relationship or not. We chose to use a threshold of 91%. Thus, an emerging relation is defined as any entity-entity pair, which does not already have a connection in the KG, and whose probability of being an emerging relation is greater than 91%.

4.4 Incremental Update of the KG

In our initial project proposal we defined this as a goal we hoped to reach, but thought it would be the most challenging to accomplish given the time frame. Unfortunately, this was the case and we were unable to finish the implementation of this feature.

5 EXPERIMENTS

5.1 Data Processing

We use two real-world data sets in this project.

- **BBC News** [6]: Documents in five topical areas are collected from the BBC news website from 2004 to 2005. We consider each sentence in the document as a piece of news. The link for this data set is http://mlg.ucd.ie/files/datasets/bbc-fulltext.zip
- **DBpedia Ontology**: we plan to use DBpedia or some subset of it as our KG. It contains the entities and relations. The current release is 3.5. Link to data set is https://wiki.dbpedia.org/data-set-35.

Table 1 summarizes the statistics of **BBC News** data sets. One example in the *DBpedia Ontology* is as follows,

```
<owl:Class rdf:about="http://dbpedia.org/ontology...">
  <rdfs:label xml:lang="en">basketball player<...>
  <rdfs:subClassOf .../ontology/Athlete">...
</owl:Class>
```

We can see that this entry has two entities, *basketball player* and *Athlete*. The relation (i.e. *Basketball player* is a **subclass** of *Athlete*) is also encoded here.

In this project, we sampled a subset of news articles from the political category in the BBC news corpus to run the experiment.

	News	Heterogeneous textual graphs							
	|D|	EE		EC		CC		KG	
		V	E	V	E	V	E	V	E
BBC	20	155	390	1199	2421	1074	7034	106	241

Table 1: Statistics of BBC News dataset (sampled) used in this

5.2 Results

Table 2 shows the performance assessment with various d (*e.g.* 4, 10, 18) and θ (*e.g.* 0.2, 0.5, 0.7). We notice that the precisions are rather low whereas the recalls are surprisingly high. We manually checked the predictive results. The prediction were comprised of many false positive cases. Hence, our precision is low.

Parameter	Precision	Recall	F1 score
$d = 4, \theta = 0.7$	0.08	0.94	0.07
$d = 10, \theta = 0.7$	0.08	1.00	0.07
$d = 18, \theta = 0.7$	0.08	1.00	0.07
$d = 18, \theta = 0.2$	0.08	0.94	0.07
$d = 18, \theta = 0.5$	0.08	1.00	0.07
$d = 18, \theta = 0.8$	0.08	1.00	0.07

Table 2: Performance assessment with various d and θ

5.3 Case study

In this section, we present two case studies to show the effectiveness of our algorithm.

Figure 3 shows the entity-entity relation learned from the news article. The selected entity in this figure is "Kennedy" (highlighted in cyan color). 6 nearest neighbors of "Kennedy" are "Democrats", "Labour", "Manchester", "Jeremy", "Straw", "Balls" (highlighted in blue color). Surprisingly, our algorithm captures the fact that John F. Kennedy was affiliated with Democratic party.

Another interesting example in our study is the entity pair (Rome, EU). Rome was considered as new entity and EU as known entity

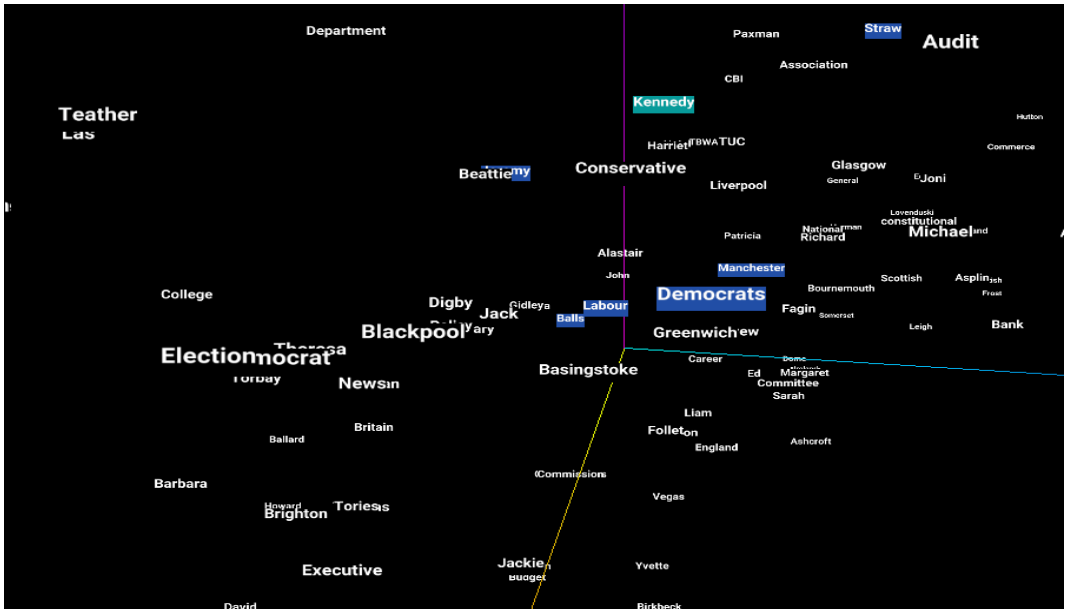

Figure 3: Visualization of entity-entity relation in 3-dimensional space

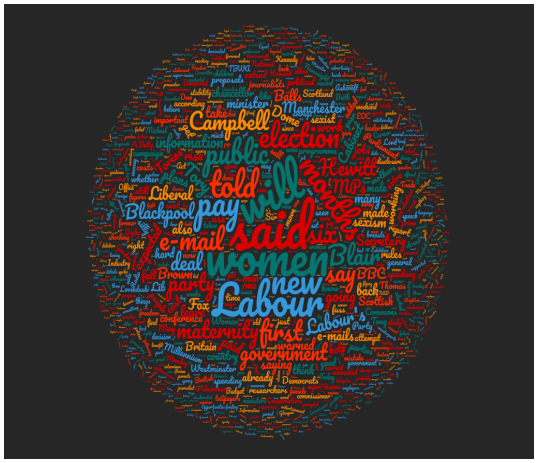

Figure 4: WordCloud representation of the sampled BBC news dataset

in our experiment. Both entities were mentioned within the same sentence in our dataset. Our model not only successfully predicted this relation, but also gave us another plausible prediction, (Rome, Europe).

6 DISCUSSION

Here,we discuss the problems we encountered and challenges we overcame.

- The Stanford Tokenizer (NER): As mentioned earlier, we pass the news to the tokenizer to get the entities and their type, which is either person, location, organization, or none. If it's none we take it as a contextual word, otherwise it's an entity. The NER tokenizer produced different labels for some words, such as FIFA, and Blaire. When we wanted to extract the sub-graphs and use them to produce the joint embedding, we had problems in the updating part because of this multi-labeling. This problem led us not to use the whole dataset, and not having a proper dataset caused poor predictions.
- Dataset: The Stanford Tokenizer's errors, as we discussed earlier, prevent us from using the whole dataset, so we use a small part and removed the problematic words. So the ground truth was a small subset of the whole ground truth, and this is a reason for the loew precision.
- Dimension of the joint embeddings: We used a d-dimensional vector for the joint embeddings, and different d's have different results. We checked the results for 3 different dimesions. The comparison is shown in the figure 5. We should mention that because of the small dataset problem we discussed earlier, the difference is not fully observable.
- Parameter θ: as mentioned earlier, parameter θ is a guiding parameter that trades off between news and the knowledge graph. Different θs produce different embedding and thus have different results. We checked the results for 3 different θs. The comparison is shown in the figure 6. We should mention that because of the small dataset problem we discussed earlier, the difference is not fully observable.

7 CONCLUSION

In conclusion, we have implemented an algorithm for detecting emerging relations using the combination of a Heterogeneous Textual Graph constructed from news articles and an existing Knowledge Graph derived from DBpedia. The nodes within the HTG are comprised of both entities and contextual words, and the edges

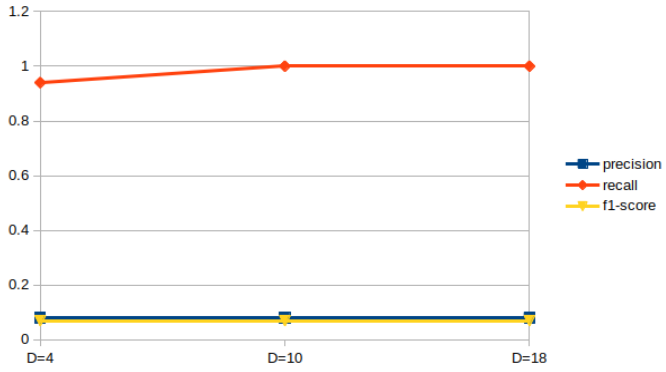

Figure 5: Performance assessment with various d

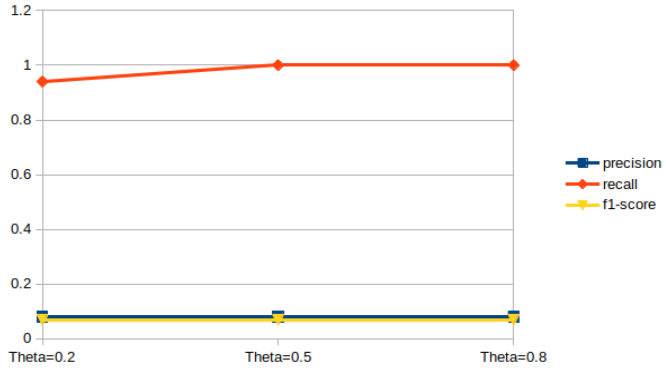

Figure 6: Performance assessment with various θ

between them were defined by the co-occurrence frequencies of the words within a 5-word sliding window. The nodes within the KG are comprised of only entities and the edges that connect them represent relationships that we are currently aware of. Using these graphs we are able to embed the words from the news into a d-dimensional vector, which are then combined using a function h for every possible entity-entity pair. These combined embedding are then fed to classifier which predicts the probability that an entity-entity pair is an emerging relation. We defined any entity-entity pair with greater than 91% probability to be an emerging relation.

8 TASK ASSIGNMENT

- Data cleaning; Constructing a heterogeneous textual graph from news - Hui
- Joint Embedding of the news and KG
- Implement HEER and detect Emerging Relations - Eric
- Experiment and evaluation
- Report writing - Eric, Hui

9 REPOSITORY

We implement the project in python, and the implementation and all the documentation can be found on https://github.com/jiaowoshabi/cs247.git

REFERENCES

[1] Matt Gardner and Tom Mitchell. 2015. Efficient and expressive knowledge base completion using subgraph feature extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1488–1498.

[2] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 541–550.

[3] Ni Lao, Tom Mitchell, and William W Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 529–539.

[4] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, 1003–1011.

[5] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*. Association for Computational Linguistics, 455–465.

[6] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[7] Jingyuan Zhang, Chun-Ta Lu, Mianwei Zhou, Sihong Xie, Yi Chang, and S Yu Philip. 2016. Heer: Heterogeneous graph embedding for emerging relation detection from news. In *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 803–812.