Relation Extraction using Pattern Generation and Semantic Embeddings

Danish Ahmed

University of Paderborn
DICE Group

Supervisor:
Dr. Ricardo Usbeck

June 19, 2019

Overview

Motivation

- 12,500+ words
  - 1000 sentences approximately?
- Almost 60 triples in infoxbox

Education	Swiss Federal Polytechnic (1896–1900; B.A., 1900) University of Zurich (Ph.D., 1905)

Example

Sentence: *Albert attended a Catholic elementary school in Munich.*
Triple: dbr:Albert_Einstein dbo:education dbr:Catholic_school

- Extend existing and create new knowledgebases
- Minimize proofreading

Problem Statement

Goal

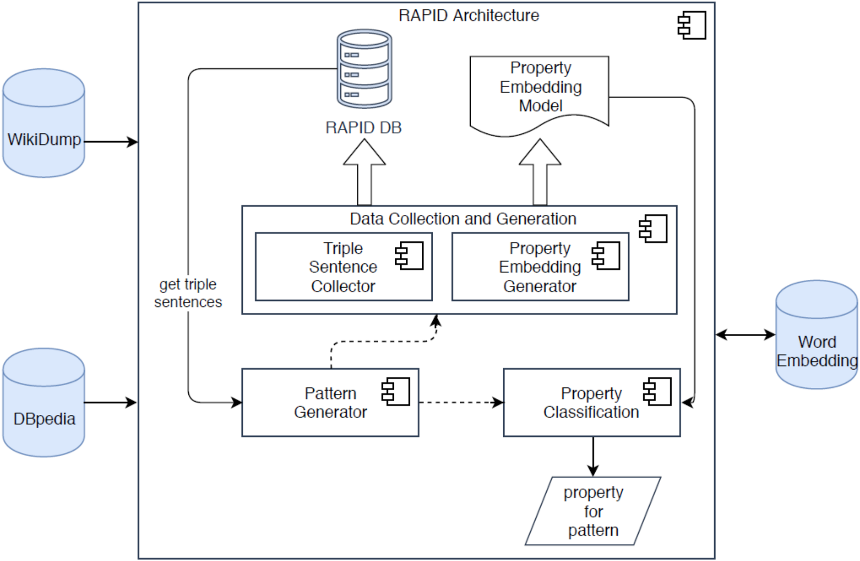
Identify relations that could hold between entities and to disambiguate between relations.

RAPID

$RAPID(D) = \bigcup_{i=1}^{T}(e_x, p, e_y) : e_x \neq e_y \mid D, E$

- D is a document set having sentence(s)
  - $D = \{se_1, se_2, ..., se_n\}$
- E is the set of entities in Document
  - $E = \{e_1, e_2, ..., e_i\}$
- (e_x subject, p predicate, e_y object)
- T number of relations determined by RAPID

RAPID Architecture



Preprocessing

- Index creation using Elasticsearch
- Triple selection using DBpedia
- Harvesting sentences

Candidate Sentences (CSE)

$$\{\bigcup_{i=x}^{y} se_i : (se_x \supset l_s) \wedge (se_y \supset l_o)\} \vee \left\{\bigcup_{i=y}^{x} se_i : (se_y \supset l_s) \wedge (se_x \supset l_o)\right\}$$

Extending Sentences

$$CSE = CSE \cup \left\{\bigcup_{k=y+1} se_k : (se_k \supset l_s) \oplus (se_k \supset l_o)\right\}$$

Sentence Refinement (Coreference Resolution)

$$CSE = \bigcup_{i=1}^{n} se_i : (se_i \supset l_s) \wedge (se_i \supset l_o)$$

Property Embedding Model Generation

- Words that semantically reflects the property
  - Property label (L_p)
  - Property Comment (L_c)

Label Set (L)

$L = L_p - (L_c - SP)$

- Vocabulary Expansion (WordNet)
  - Synonyms
  - Hyponyms

Synonyms - *spouse*

{partner, mate, better half, spouse, married person}

Hyponyms - *spouse*

{married man, wife, honeymooner, polygamist, monogamist, husband, married woman, bigamist, helpmeet, consort, monogynist, newlywed, hubby, helpmate}

Property Embedding Model Generation

LabelSet (L)

$L = L \cup wordNet(L) : wordNet(L) = \{synonyms(L) \cup hyponyms(L)\}$

- Vocabulary Normalization
  - Lemmatization

Lemma

$Lemma(\{employ, employed, employing\}) \rightarrow employ$
$Lemma(employee),\ Lemma(employer),\ Lemma(employment) \rightarrow ?$

- Lexical-related words

Word in Label set (l_w)

$l_w = lemma(nn2vb(word))$

Property Vector Generation

- Vocabulary is a semantic representation of a property
- HashSet?
    - Cannot handle words outside vocabulary
- Single vector representation for each property
- Use of Word Embeddings
    - Word2Vec
    - Global Vectors for Word Representation (Glove)
    - fastText

Property Vector Generation

$$P = \{p_1,\ p_2,\ \dots,\ p_i\}$$

$$\downarrow$$

$$l_{p_i} = \{w_1,\ w_2,\ \dots,\ w_j\}$$

$$\downarrow \qquad \downarrow \qquad \qquad \downarrow$$

$$\begin{bmatrix} v_{11} \\ v_{12} \\ \cdot \\ \cdot \\ \cdot \\ v_{1d} \end{bmatrix} \begin{bmatrix} v_{21} \\ v_{22} \\ \cdot \\ \cdot \\ \cdot \\ v_{2d} \end{bmatrix} \cdots \begin{bmatrix} v_{j1} \\ v_{j2} \\ \cdot \\ \cdot \\ \cdot \\ v_{jd} \end{bmatrix} \longrightarrow \begin{array}{c} u_i \\ \begin{bmatrix} \frac{\sum_{k=1}^{j} v_{k1}}{j} \\ \frac{\sum_{k=1}^{j} v_{k2}}{j} \\ \\ \frac{\sum_{k=1}^{j} v_{kd}}{j} \end{bmatrix} \end{array}$$

Property Embedding Model (PEM)

$$u_i = \left[\frac{\sum_{k=1}^{j} v_{k1}}{j}, \frac{\sum_{k=1}^{j} v_{k2}}{j}, \ldots, \frac{\sum_{k=1}^{j} v_{kd}}{j}\right]$$

```
affiliation -0.017893536 -0.045529217 -0.0051208185 -8.97618E-4
almaMater 0.013885535 0.04698701 0.0010437527 0.05139419 0.04159
bandMember 0.052005745 -0.07762692 -0.015077149 0.01401134 -0.04
birthPlace 0.021396212 0.045171227 0.014700594 0.004971906 -0.02
ceo 0.02789023 -0.0367615 -0.00690852 -0.04308021 -0.008731632 -
child 0.010353886 0.014000095 -0.022126973 0.023273598 0.0015987
club -0.021402836 -0.009516117 0.04230016 -0.02254374 0.04469683
deathPlace 0.018342787 0.04798468 -0.0023197616 0.08208557 -0.03
debutTeam -0.0192695 0.019238498 -0.026177414 -0.009202471 0.034
department -0.07548131 0.037946284 0.013026152 -0.03266698 -0.0
district 4.9547944E-4 -0.004887579 0.023869252 -0.022869527 -0.0
doctoralAdvisor -0.01991654 0.004269961 0.05408489 7 0.02728245 0
doctoralStudent -0.005935667 0.0089201685 0.009387679 0.01739716
```

Figure: PEM file

Pattern Generation

- Semantic subgraphs generation between entity nodes
- Generated subgraphs filtration
- Subgraph refinement
- Pattern representation

Semantic Subgraphs Generation

Goal

- Let $G = (V, E)$ be original semantic graph
- x_1 and x_2 are entities
- $\mathrm{G}' = (\mathrm{V}', \mathrm{E}') : (\mathrm{V}' \subset \mathrm{V} \wedge \mathrm{E}' \subset \mathrm{E}) \wedge (\exists \mathrm{x}_1 \in \mathrm{V}' \wedge \exists \mathrm{x}_2 \in \mathrm{V}')$

- Each subgraph equivalent to a pattern *pt*
- Determine node representing entities
  - No intersection between node sets of subject (*IWS*) and object (*IWO*)
- Make combinations from *IWS* and *IWO*, and determine shortest undirected path for each combination
  - Set of semantic graph edges
  - Add possession typed dependency nodes from G to G' if in MTYPD set
  - Determine the root node

Generated Subgraphs Filtration

Example

Input: *Ahmed studied at University of Paderborn, and his thesis was under the supervision of Dr. Ricardo.*

G_1' *Ahmed* (0), *Dr. Ricardo* (15)

```
[studied/VBD
    nsubj>Ahmed/NNP
    conj:and>[supervision/NN
            cop>was/VBD
            case>under/IN
            nmod:of>[Ricardo/NNP case>of/IN compound>Dr./NNP]]]
```

G_2' *Ahmed* (7), *Dr. Ricardo* (15)

```
[supervision/NN
      nsubj>[thesis/NN nmod:poss>[Ahmed/NNP case>'s/POS]]
      cop>was/VBD
      case>under/IN
      nmod:of>[Ricardo/NNP case>of/IN compound>Dr./NNP]]
```

Subgraph Refinement

Rule

If a nominal modifier is followed by the same nominal modifier, then the semantic graphs can be split into two

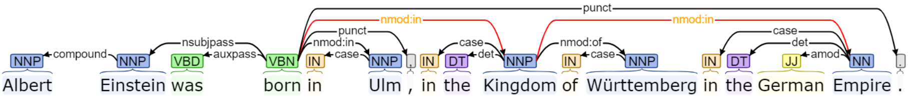


Figure: Inclusion of another object node

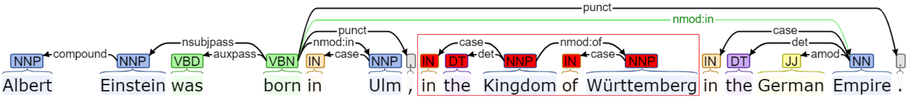


Figure: Node removal and edge creation

- Domain (%D%) and Range (%R%) replacement

Pattern Representation

Pattern Format

{rootToSubjectPath}rootLemma{rootToObjectPath}

- Two representations
  - Generalized
  - Extended

Example

Input: *Ahmed studied at University of Paderborn, and his thesis was under the supervision of Dr. Ricardo.*
Subject: *Ahmed*
Object: *Dr. Ricardo*
G. Pattern: *{(NN)-nsubj>(NN)-nmod:poss>%D%(NNP)} supervise{(NN)-nmod:of>%R%(NNP)}*

Property Recognition

- Given input sentence(s)
  - Determine entities and their entity type
  - Find entity nodes from original semantic graph
  - Make entity combinations and generate patterns
  - Classify each pattern against properties satisfying domain and range
- Means to score a pattern

Support

- Pattern pt occurrence in classification property cp_i itself

$$sup(pt, cp_i) = \frac{freq(pt) : pt \in PT_i}{totalPatternInProperty(cp_i)}$$

Specificity

- Pattern pt occurrence in other properties OP

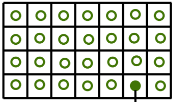
$$spc(pt, cp_i) = \frac{freq(pt) : pt \in getPatterns(OP)}{\sum_{k=1}^{sizeOf(OP)} totalPatternsInProperty(op_k)}$$

Pattern Similarity

- Select sequences from pattern ($seq_{in_{subj}}$, $seq_{comp_{subj}}$)
- Align sequence by matrix formation
- Populate matrix with similarity score

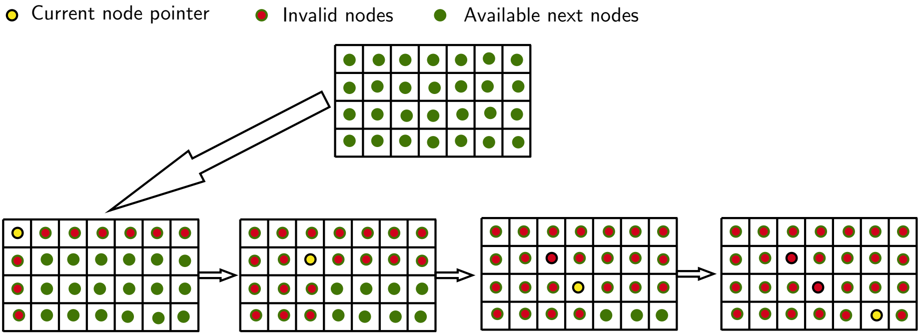
Each Matrix block m_{ij}

$$\{0.5 + cos(POS_{seq_{1j}}, POS_{seq_{2i}}) + WESim(label(seq_{1j}), label(seq_{2i}))\}$$
$$\times \{0.5 + cos(typD_{seq_{1j}}, typD_{seq_{2i}})\}$$



$$\begin{aligned} m_{35} &= (0.5 + cos(VB, VBD) + WES(deny, refuse)) \\ &\times (0.5 + cos(acl : relcl, acl : relcl)) \\ &\Longrightarrow (0.5 + 0.816 + 0.442) \times (0.5 + 1) \\ &\Longrightarrow 2.637 \\ norm(2.637) &= 0.722 \end{aligned}$$

Forskippo Semantic Similarity (FSS)



Forward Skipping Rules

- No previous block selection
- Available matrix starts from $m[i+1][j+1]$
- Select any node as next node from available matrix
- No next moves once current pointer at last row

Pattern Similarity

- Create summation matrix of same size
- Populate matrix by maximum summation similarity value that can be obtained starting from each matrix block

Pattern Similarity Score

$$ps(pt_i, pt_{comp}) = \frac{FSS(seq_{in_{subj}}, seq_{comp_{subj}}) + FSS(seq_{in_{obj}}, seq_{comp_{obj}})}{2}$$

- Similarity with respect to property?
  - Normalized words from pattern
  - Get vector representation from source embedding model
  - Calculate mean vector v_{pmean}

Embedding Similarity $ems(v_{pmean}, cp_i)$

$$getSimilarity(v_{pmean}, v_{cp_i}) : v_{pmean} = [\frac{\sum_{k=1}^{n} v_{k1}}{n}, \frac{\sum_{k=1}^{n} v_{k2}}{n}, \ldots, \frac{\sum_{k=1}^{n} v_{kd}}{n}]$$

Confidence Value

- Unsupervised learning
- α as pattern boosting parameter
- β as embedding boosting parameter

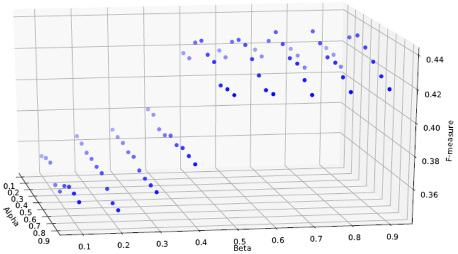
Confidence $\varrho(pt_{in}, cp_i)$

$$\{\alpha(sup(pt_{in}, cp_i)) + (1-\alpha)(spc(pt_{in}, cp_i))\} \times ps(pt_{in}, pt)$$
$$+ \beta(ems(v_{pt_{iMean}}, cp_i))$$

1. ForEach Property *cp* in *CP*
    1. Calculate embedding similarity of input pattern with candidate property
    2. get pre-generated patterns for candidate property: *trainPatterns*
    3. ForEach Pattern pt_{cp} in *trainPatterns*
        1. calculate confidence
        2. compare maximum score

Evaluation

- OKE train dataset (96 files)
- Semantic comparison using Source and Property Embedding Models
  - Word2Vec
  - Glove
  - fastText



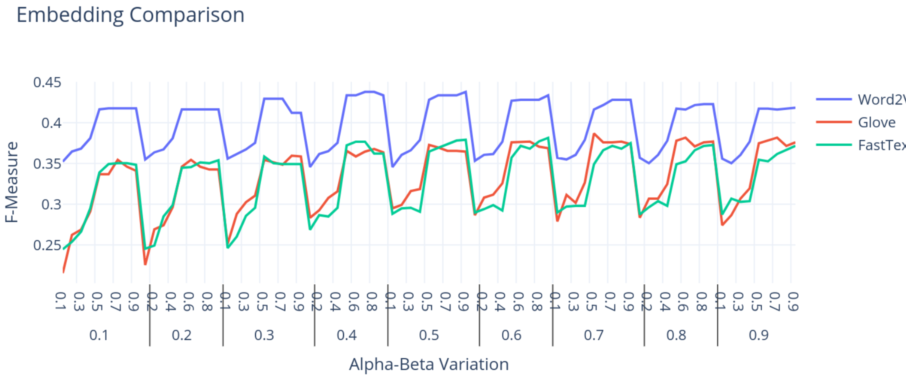
Embedding	Precision	Recall	f-measure
Word2Vec	0.30223880597014924	0.7941176470588235	0.4378378378378378
Glove	0.250	0.8539325842696629	0.38676844783715014
fastText	0.250	0.8045977011494253	0.38147138964577654

Discussion

Embedding	(Alpha, Beta)
Word2Vec	(0.4, 0.7), (0.4, 0.8), (0.5, 0.9)
Glove	(0.7, 0.5)
fastText	(0.6, 0.9)

- Corpus Independence
- High Recall
  - Coreference Resolution
- Low Precision
  - Property for each pattern
- Better results with $\beta \geq 0.5$
  - Property Embedding Model performs good
- Better results with $\alpha \geq 0.4$
  - FSS performs satisfactory but can be improved
- Same f-measure for multiple parameter combinations
  - Property parameter dependency

Embedding Comparison



Empirical Threshold

Goal

Get rid of False Positive results

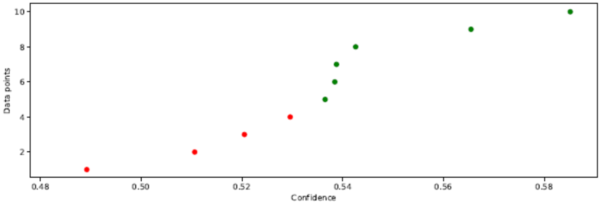


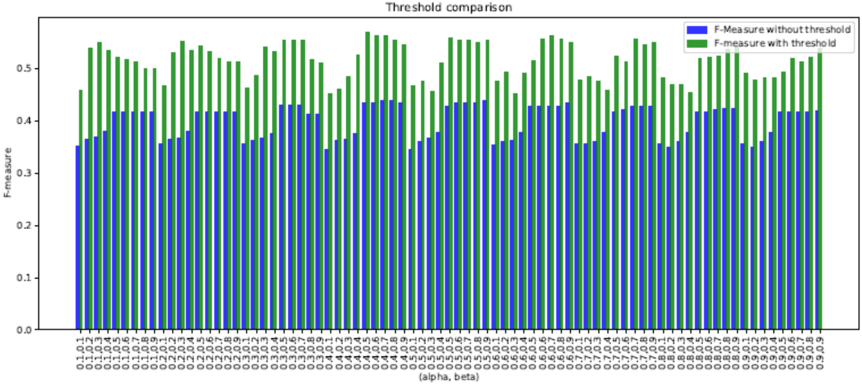
Figure: dbo:birthPlace

Threshold

$\theta = \frac{(\mu_{FP}+\sigma^2_{FP})+(\mu_{TP}-\sigma^2_{TP})}{2}$

- $(0.51268521 + 0.55048492)/2 \implies 0.531585065$

Word2Vec Results after Empirical Threshold



Limitations And Future Work

- Limitations
  - Single property per pattern
  - False results due to lack of Sentiment Analysis
- Future Work
  - Multiple properties per pattern if $>$ threshold
  - Named Entity Recognition Enhancement
  - Generate Patterns for other DBpedia properties
  - Supervised Learning

Live Demo

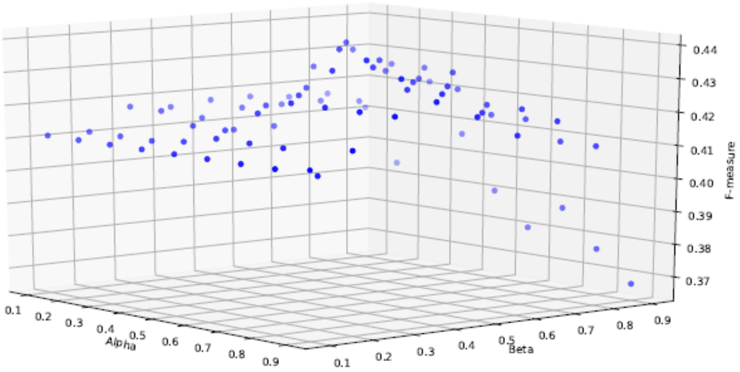
- http://danish-thesis.cs.upb.de:3000/

Acknowledgement

- Special Thanks to Dr. Ricardo Usbeck
- Prof. Dr. Axel Ngonga
- Parents and my dearest brother Ali Shah

Questions?

Modified Equation for Confidence



Confidence

$$\beta\,\{\alpha(sup(pt_{in}, cp_i)) + (1-\alpha)(spc(pt_{in}, cp_i))\} \times ps(pt_i, pt)$$
$$+\ (1 - \beta)(ems(v_{pt_{iMean}}, cp_i))$$