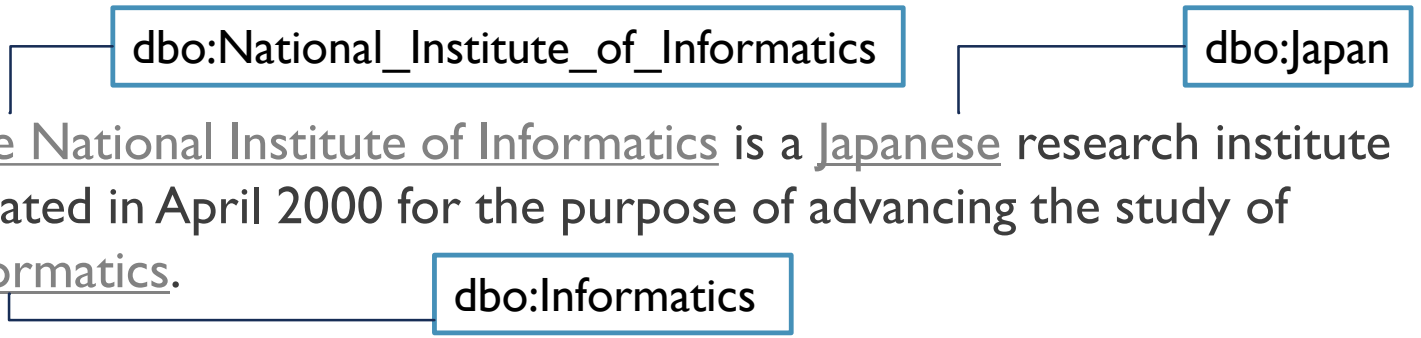
COMBINING WORD AND ENTITY EMBEDDINGS FOR ENTITY LINKING

JOSE G. MORENO, ROMARIC BESANCON, ROMAIN BEAUMONT, EVA D'HONDT,
ANNE-LAURE LIGOZAT, SOPHIE ROSSET, XAVIER TANNIER, AND BRIGITTE GRAU

Presentation By
Piyawat Lertvittayakumjorn (Peter)
31st May 2017

ENTITY LINKING

- Entity Linking (or Entity Disambiguation) task consists of connecting an entity mention that has been identified in a text to one of the known entities in a knowledge base
- Input: A text with the offsets of entity mentions
- Output: One corresponding entity in KG for each entity mention
- E.g.



Text from: https://en.wikipedia.org/wiki/National_Institute_of_Informatics

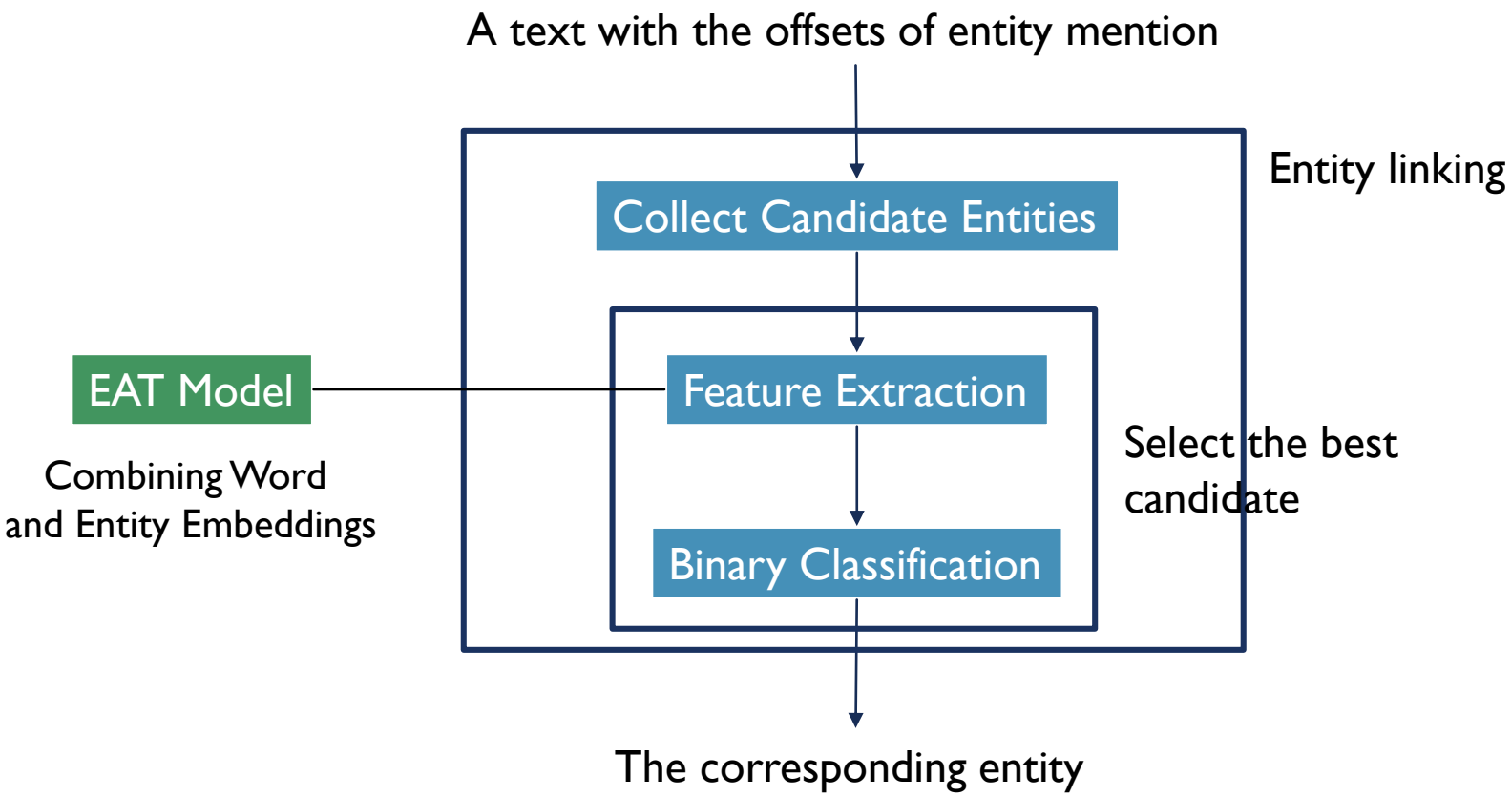
A DIFFICULT EXAMPLE

- Trump ① has been plagued with a series of leaks coming from sources within the White House ② since his inauguration ③

①	②	③
A. Trump (a playing card)	A. White_House	A. Inauguration
B. Donald_Trump	B. White_House_Fellows	B. United_States_presidential_inauguration
C. Ivanka_Trump	C. The_White_House_Project	C. Papal_inauguration
D. Eric_Trump	D. White_House_(film)	D. Irish_presidential_inauguration
E. Tour_de_Trump		
F. Trump_Airlines		

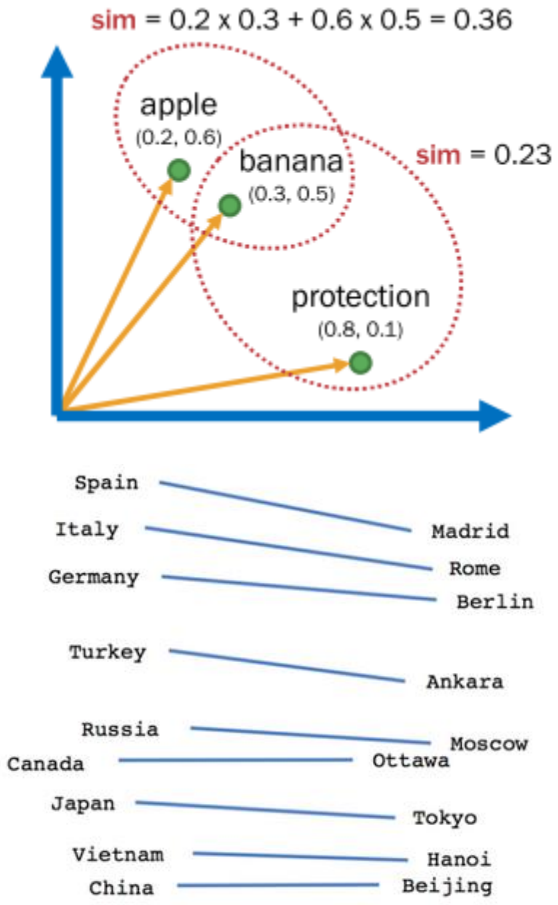
Text from: http://www.politico.com/story/2017/05/28/trump-fake-news-media-238894

FRAMEWORK FOR ENTITY LINKING



WORD EMBEDDING

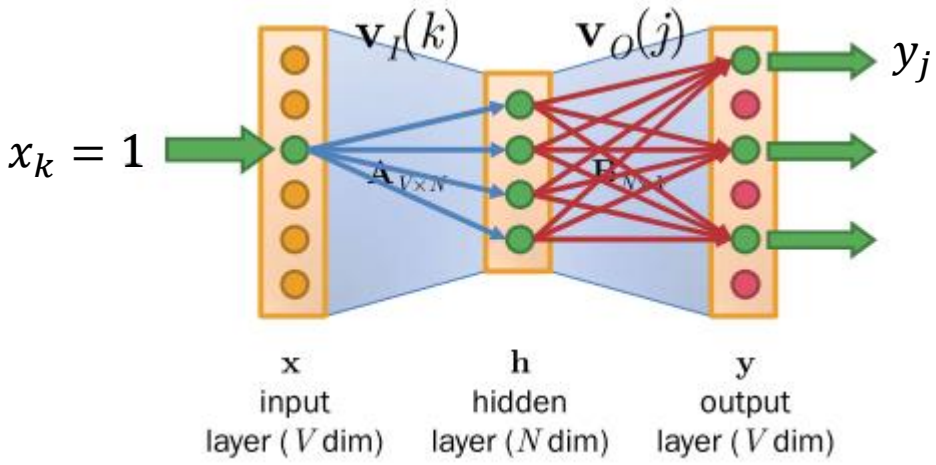
- Represents a word by a vector (in a vector space) w.r.t. its context
- The dot product of two word vectors = Word Similarity
- The distance = their pointwise mutual information (PMI)
- The difference vector embeds the relationship of the two words
- Word2Vec is Google's NLP tool for computing word embeddings



Source: (i) Word2Vec: Foundation, Application, and Beyond (By Prachya Boonkwan) / (ii) https://www.tensorflow.org/tutorials/word2vec

SKIP GRAM (MIKOLOV+, 2013)

- Given a word w_k, find its C context words by using a neural network
  - The size of input and output layers = |V| (No. of vocabs)
  - The size of the hidden layer = N (the length of embedding vectors)
  - The matrix A stores embeddings of |V| words



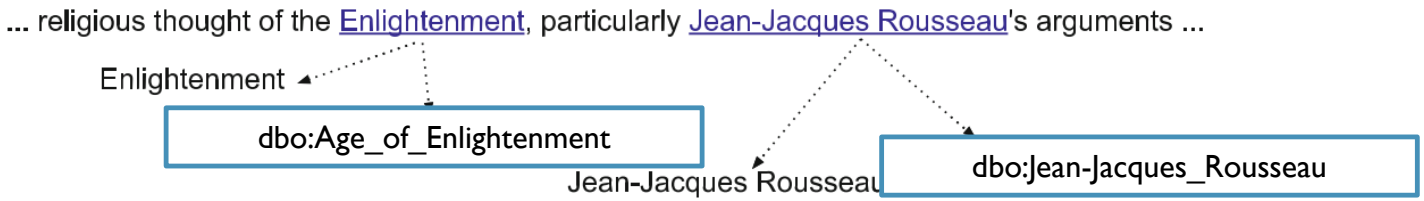
$$P(w_j|w_k) = \frac{\exp(y_j)}{\sum_{i=1}^{|V|}\exp(y_i)} = \frac{\exp(v_O(j)^T v_I(k))}{\sum_{i=1}^{|V|}\exp(v_O(i)^T v_I(k))}$$

Popular open source implementations are such as Gensim and Hyperwords

Source: Word2Vec: Foundation, Application, and Beyond (By Prachya Boonkwan)

WORD & ENTITY EMBEDDING

- Consider a text in Wikipedia, the entity mention is an anchor text which represents both text and entity (wikipage).



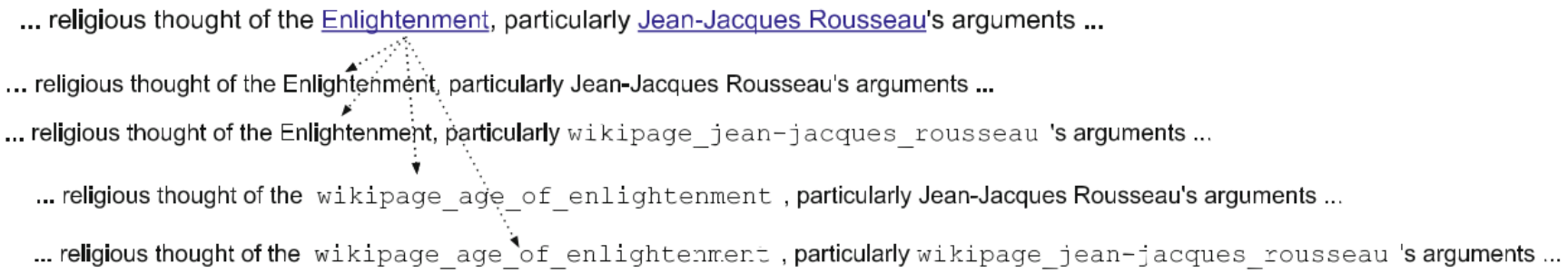
- Since every entity, represented by an anchor text, has its own context words, the entity could have an embedding vector as normal words have.
- This concept is called as Extended Anchor Text (EAT)

THE EAT MODEL - OVERVIEW

- Consider the word c in a text, c can be either
  - a normal word, i.e., c = (w), or
  - an extended anchor text, i.e., c = (w, e),
    where e is a corresponding entity of this anchor text
- We will find embedding vectors of all w and all e. Formally,
  - The full vocabulary size $F = V \cup \xi$ where
    - V is a set of all possible words and
    - ξ is a set of all entities
  - The embedding function $f: F \to R^d$

THE EAT MODEL - TRAINING

- For EAT, c can represent both a word and an entity, so we have to separate the training text before feeding to the skip gram model



- At the output layer,

$$P(c_j|c_k) = \sum_{a_j \in c_j} \sum_{a_k \in c_k} \frac{\exp(v_O(a_j)^T v_I(a_k))}{\sum_{i=1}^{|F|} \exp(v_O(a_j)^T v_I(a_k))}$$

THE EAT MODEL - ADVANTAGES

- The skip gram model is usable without requiring any modification. (Only the training texts are modified.)
- Words and entities are represented in the same vector space, so it is easy to find the distance between a word and an entity

EXPERIMENT 1: EVALUATION OF THE EMBEDDINGS

- Using the (semantic) analogy dataset for word embeddings,
  - E.g., Paris:France Rome: _____ Answer = Italy
  - Answer the word x whose vector V_x is closest to $V_{Rome}+V_{France}-V_{Paris}$
- For evaluating entity embedding, we map each word to the corresponding entity first and use the entity vector for finding the answer
- The family relation task is not done due to missing entities
  - E.g., boy:girl dad:_____ Answer = mom

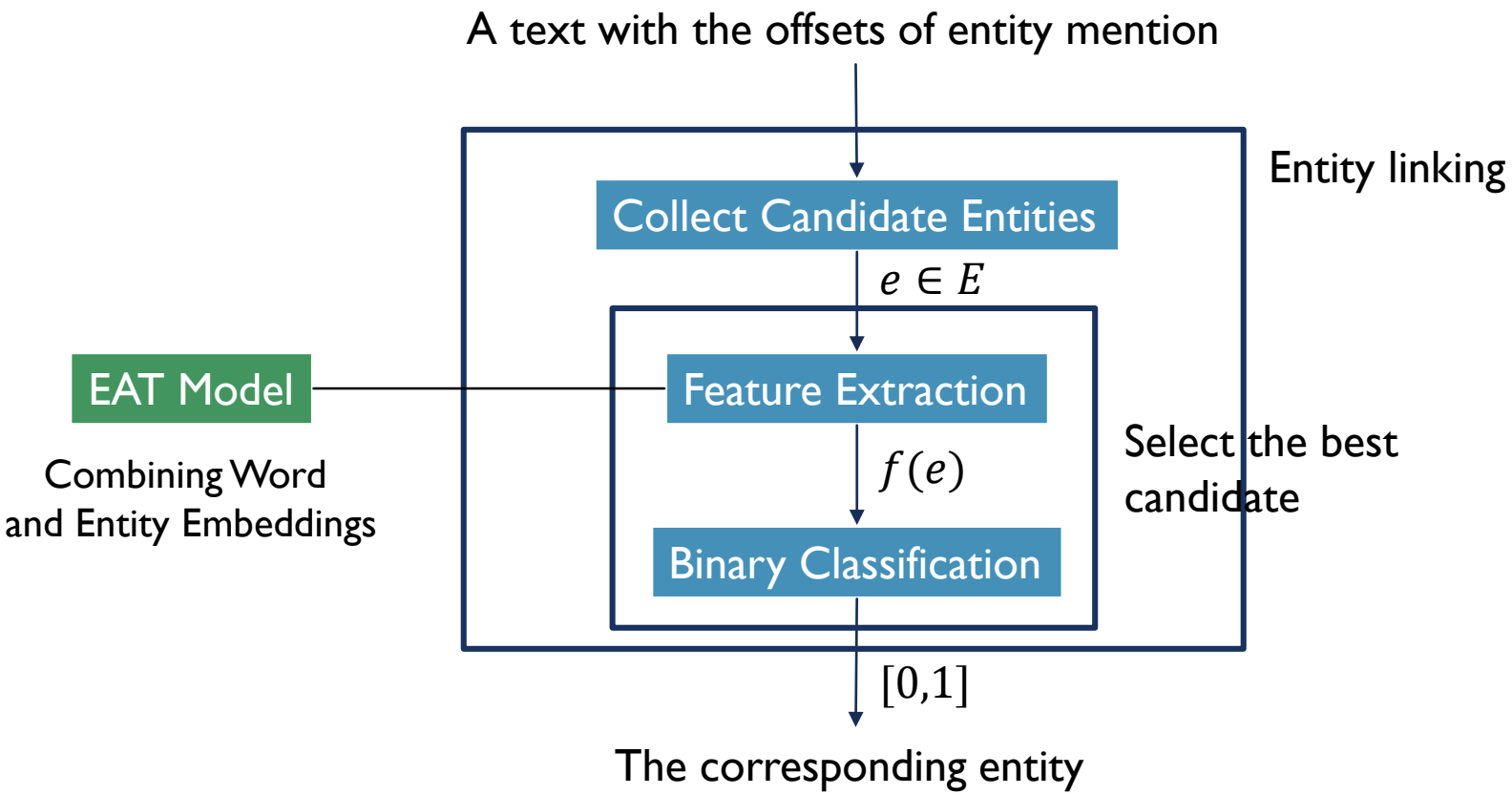
EXPERIMENT I: RESULTS

- Accuracy by semantic subgroup in the analogy task for words/entities using a high value (EAT-hyperwords) or low value (EAT-Gensim) as frequency threshold.

Subgroup	EAT-*hyperwords*			EAT-*Gensim*	
	words	entities	entity→word	words	entities
capital-com-countries	95.7%	63.0%	87.5%	75.7%	77.5%
capital-world	77.0%	37.3%	81.3%	49.7%	80.0%
currency	8.2%	0.0%	5.2%	0.0%	0.0%
city-in-state	72.3%	25.8%	62.6%	31.7%	89.8%

- High Threshold => Low |F| => High Accuracy / High missing rate
- In the entity→words column, some entities were replaced by their respective word when the entities were not part of the vocabulary

FRAMEWORK FOR ENTITY LINKING



GENERATION OF CANDIDATE ENTITIES

Given a target entity mention q

1. Recognize the type of the entity mentioned t_q
   - Person, Location, or Organization
2. Define the context in terms of surrounding named-entity mentions (ignore general noun/pronoun mention)
3. Collect variations of the entity mention q'
   - In case of acronym, search for entity mentions of the same type whose initials match the acronym
   - Search for entity mentions who include the target entity mention as a substring

GENERATION OF CANDIDATE ENTITIES

4. Create a set of candidate entities e
   a. [q or a variation of q] = e
   b. [q or a variation of q] = [a variation of e (alias/translation)]
   c. [q or a variation of q] is included in [e or a variation of e]
   d. String similarity: Levenshtein distance of [q or a variation of q] and [e or a variation of e] is less than 3
   e. Index all the known forms of the entities in the KG as documents. Select all close variants e w.r.t. their tf-idf similarity using Lucene
5. Filter to keep only entities that have at least one of the expected entity types

FEATURE EXTRACTION

- Extract features for each candidate *e*
  - 3 binary features for 4a, 4b, and 4c
  - 2 real-value features for 4d and 4e
  - 1 global similarity score = log(no. of inlinks if *e*)
  - 2 textual similarity scores
  - 4 EAT-based similarity scores

FEATURE EXTRACTION
2 TEXTUAL SIMILARITY SCORES

- For an entity mention q and a candidate entity e
  - d(q) the document in which q appears
  - w(e) the Wikipedia page associated to e
  - r(e) a text combining the set of entities that are in relation with e in the KG
- Calculate TF-IDF vectors for d(q), w(e), and r(e) denoted as d, w, and r, respectively.
- The two features are
  - Cosine similarity of d and w
  - Cosine similarity of d and r

FEATURE EXTRACTION
4 EAT-BASED SIMILARITY SCORES

- For an entity mention q and a candidate entity e
  - s(q) is the sentence number whose sentence contains the entity mention q
  - p(q) is a paragraph containing the sentences no. s(q)-1, s(q), s(q)+1
- Let $\boldsymbol{e}$ be the embedding vector of the entity $\boldsymbol{e}$ and $\boldsymbol{w_i}$ be the embedding vector of the word $\boldsymbol{w_i}$

FEATURE EXTRACTION
4 EAT-BASED SIMILARITY SCORES (CONT'D)

- The four EAT-based features are

$$EAT_1(e, p(q)) = \frac{\sum_{w_i \in p(q)} cos(\boldsymbol{e}, \boldsymbol{w}_i)}{||p(q)||}$$

$$EAT_2(e, p(q)) = cos(\boldsymbol{e}, \frac{\sum_{w_i \in p(q)} \boldsymbol{w}_i}{||p(q)||})$$

$$EAT_3(e, p(q)) = \frac{\sum argmax_{\substack{w_i \in p(q) \\ i=1 \ldots k}} cos(\boldsymbol{e}, \boldsymbol{w}_i)}{k}$$

$$EAT_4(e, w_m) = cos(\boldsymbol{e}, \boldsymbol{w}_m)$$

In the experiment, k = 3

BINARY CLASSIFICATION

- Training
  - Positive Examples: (entity mention, the correct candidate entity)
  - Negative Examples: (entity mention, other wrong candidate entity)
  - Limiting the number of negative examples to be 10 times the number of positive examples
- Testing: The candidate entity with the highest probability is selected as the answer
- If no candidate is generated or if all candidates are rejected by the classifier, it means that the entity mention does not link to any entity in the KG (these are referred as NIL entities)

MODEL SELECTION

- Many models were tested with a 10-fold cross validation using non-EAT features
  - Adaboost
  - Random Forests
  - Decision Trees
  - SVM (both linear and RBF kernels)
- Combining Adaboost with Decision Trees as base estimator turned out to be the best classifier on the training data
- Further results are obtained using this classifier

EXPERIMENT 2: ENTITY LINKING DATASET

- Use the benchmark from the EDL (Entity Discovery and Linking) task of the TAC 2015 evaluation campaign.

	TAC 2015 training	TAC 2015 testing
Nb. docs.	168	167
Nb. mentions	12175	13587
Nb. mentions NIL	3215	3379

- The KG used in this campaign is built from Freebase.
- After removing some irrelevant entity types (such as music, book, medicine and film), we have 8M entities in total.
- Among them, 46% have an associated content in Wikipedia and can thus be associated with an embedding representation.

EXPERIMENT 2: ENTITY LINKING EVALUATION OF CANDIDATE GENERATION

	$\|C\|$	$\|C_{NIL}\|$	C_{AVG}	Recall(C)
All candidates				
Training	6843513	781	562.1	95.60%
Test	8339648	499	613.8	94.19%
Entity type filtering				
Training	3179795	952	261.2	92.43%
Test	3810382	626	280.4	90.36%
Lucene+Null simil filtering				
Training	1723470	952	141.6	90.27%
Test	1921577	625	141.4	87.95%

- Without type filtering, 95% recall with large C_{AVG}
- With type filtering, C_{AVG} reduced by more than a half. The recall is decreased (around 90%), but the Entity Linking score is improved
- The candidates returned only by Lucene and the candidates for which the similarity scores are both null were not often the right ones => Remove

EXPERIMENT 2: ENTITY LINKING EVALUATION OF ENTITY LINKING

- The results for entity linking are evaluate using precision, recall, and F1 score.

$$P(nil) = \frac{N(e_t = \text{NIL} \wedge e_r = \text{NIL})}{N(e_t = \text{NIL})} \quad R(nil) = \frac{N(e_t = \text{NIL} \wedge e_r = \text{NIL})}{N(e_r = \text{NIL})}$$

$$P(link) = \frac{N(e_t = e_r \wedge e_t \neq \text{NIL})}{N(e_t \neq \text{NIL})} \quad R(link) = \frac{N(e_t = e_r \wedge e_t \neq \text{NIL})}{N(e_r \neq \text{NIL})}$$

$$P(all) = \frac{N(e_t = e_r)}{N(e_t)}$$

- where e_r is a correct entity and e_t is from the prediction

EXPERIMENT 2: ENTITY LINKING RESULTS

- Entity Linking Results with the EAT feature(s)

	Baseline	$+EAT_1$	$+EAT_2$	$+EAT_3$	$+EAT_4$	$+EAT_{1/2/3/4}$
P(nil)	0.598	0.604	0.608	0.605	0.605	**0.606**
R(nil)	0.815	0.830	0.825	0.828	0.830	**0.838**
F(nil)	0.690	0.699	0.700	0.700	0.700	**0.704**
P(link)	0.796	0.806	0.800	0.804	0.806	**0.814**
R(link)	0.699	0.706	0.706	0.706	0.707	**0.710**
F(link)	0.745	0.752	0.750	0.752	0.753	**0.759**
P(all)	0.728	0.737	0.735	0.737	0.737	**0.742**

- Baseline uses every feature except EAT-based features

EXPERIMENT 2: ENTITY LINKING DISCUSSION

- Using the EAT-based features outperforms the baseline
- Each individual EAT feature yields comparable results and the combined features give the best results.
- Achieve better results than the best state-of-the-art from the participants of TAC-EDL 2015*
- The EAT model performs well for incomplete names such as a mention of person without her surname, a vague word representing a company or a place.

* The measure used in TAC-EDL is stricter than the one used in this paper. It might not be appropriate to compare directly.

CONCLUSION

- Present the EAT model -- jointly representing words and entities into a unique space w.r.t. their contexts
- Require no extra alignment task (mention-to-entity) or corpus concatenation (KG+text) as previous works do
- Integrate EAT into the entity linking task without any effort
- Results: the individual EAT features as well as their combination helps improve classical similarity metrics.
- It also hypothetically achieves the first position in the EDL campaign of the employed dataset.

Q&A SESSION

All questions and comments are very welcome.