Zero-Shot Relation Extraction via Reading Comprehension

Omer Levy Minjoon Seo Eunsol Choi Luke Zettlemoyer
University of Washington
Presented by Xiaodong Yu
University of Illinois, Urbana-Champaign

What's Relation Extraction (RE)?

- Given a sentence and a pre-defined list of relation types, detect the relation between entities.
- Relation types example:
  - Located, Family, Ownership, …
  - Too many relations. In this paper, they use around 120 relations.

- Example:
  - Sentence: Turing obtained his PhD from Princeton.
  - Relation: educated_at(Turing, Princeton)

Idea

- Detecting relations by answering several simple questions associated with each relation.
- Map each relation R(x,y) to several natural language questions about x whose answer is y.
- Example:
  - Sentence: Turing obtained his PhD from Princeton.
  - Relation: educated_at (x (“Turing”),y (Princeton))
  - Question: Where did x (“Turing”) graduate from? In which university did x (“Turing”) study?
  - Answer: Princeton.

Approach

- Goal: Given a relation R, a sentence S and an entity *X*, find Y that R(X, Y) holds.
- Approach:
  - Change R(X, ?) to several questions q
  - Use a reading comprehension model to answer the questions q
  - If one of the questions has the answer, then use this answer as Y.
  - If not, then X doesn’t have this relation R.
- Advantage:
  - Zero Shot. For each new relation in test, just use the same reading comprehension model to answer the questions about this new relation. They only need users to input the information of the questions associated with the new relation.

Key Problems

- What training data to use?
- How to generate the questions for each relation? How to verify that the questions do have the expected answer?
- How to train the reading comprehension model? Given a sentence and a question, how do they find the answer? What if the relation doesn’t exist in this sentence and the problem is unanswerable?

What training data to use?

- WikiReading dataset (Hewlett et al., 2016):
  - Each instance is: R(e,a) and the corresponding Wikipedia document D for e.
  - Assume that R(e, a) can be derived from D.

- For each R(e,a), they take the first sentence s in D that contains both e and a.

How to generate the questions for each relation?

- Crowdsourcing : Ask internet users to annotate.
- For each relation, they present the annotator with 4 sentences, and a masked entity X and entity Y. Annotators need to think about the questions about X whose answer is Y.
  - The wine is produced in the X region of **France**.
  - Question: In which country is X?
- After generating all the questions, they present some sentences and questions to annotators. If the annotators' answer matches the expected entity Y, then this question is valid.

How to generate negative examples?

- The reason for negative examples:
  - All the questions annotated by internet users are positive examples. All the questions have the answer in the sentence.
  - Not every relations exist in every sentence.
  - The model need to learn when a relation doesn’t exist in a sentence.
- Naïve method:
  - For each sentence, and entity X, they ask some questions about X, which pertains to other relations that does not exist in this sentence.
  - Example: Turing obtained his PhD from Princeton.
  - Question: Who is Turing’s father?

How to train the reading comprehension model?

- They call QA system as a black box to answer the questions. In principle, any QA system can work in their system.
- In this paper, they use BiDAF model (Seo et al., 2016), a RNN model.
- Input: sentence S, and a question Q.
- Output: Confidence score y^{start} and y^{end}, for every potential start word and end word.
- Then to answer the question, they just need to find the largest probability, $P(i,j) = P(y_i^{start}) * P(y_j^{end}), \quad 1 \leq I, j \leq N$

How to train the reading comprehension model?

- Problem: What about negative examples? QA assumes all the answers can be found in the sentence.
- They concatenate a trainable bias b to the confidence score.

$$z^{start} = [\, y^{start}, b] \qquad z^{end} = [\, y^{end}, b]$$

- Before: Find the largest

$$P(i,j) = P(y_i^{start}) * P(y_j^{end}), \quad 1 \le i, j \le N$$

- Now: Find the largest

$$P(i,j) = P(z_i^{start}) * P(z_j^{end}), \quad 1 \le i, j \le N+1$$

- If i = j = N+1, it means this question is unanswerable.

Evaluation

- Comparison Systems:
  - RNN Labeler (Hewlett et al. 2016). Have a good result on WikiReading (Hewlett et al. 2016)
  - Miwa and Bansal, 2016. Have a robust performance across a number of dataset.

Unseen entities

- Entities never appear in training data. For example, all the person names entities in training data are Turing, but all the person names entities in test data are Steve Jobs.

	Precision	Recall	F1
RNN Labeler	62.55	62.25	62.40
Miwa & Bansal	96.07	58.70	72.87
Question Ensemble	88.08	91.60	89.80

Unseen questions

- When they ask questions about a relation, what if this question hasn't been seen in the training data?
- Not sure why they evaluate this. This seems to be a evaluation of QA system, and they just call the State-of-the-art QA system as a black box.

	Precision	Recall	F1
Seen	86.73	86.54	86.63
Unseen	84.37	81.88	83.10

Unseen Relations

- Their key point. Their method can do zero-shot RE. So what's the performance of a new relation in test data?

	Precision	Recall	F1
RNN Labeler	13.28	5.69	7.97
Miwa & Bansal	100.00	0.00	0.00
Question Ensemble	45.85	37.44	41.11

- Though 41.11 F1 is not a high result, they set a bar for future work, which is important.

Conclusion

- They reduce the problem of extracting relations to answering simple questions about the relation.
- Pretty good result on RE.
- More importantly, support Zero-Shot.