Appendix: On Generating Fact-Infused Question Variations

Arthur Deschamps
ByteDance Ltd.
Singapore
arthur.deschamps1208@gmail.com

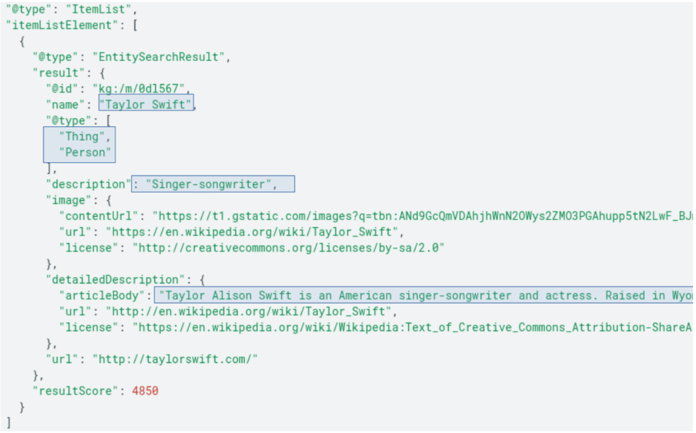
Sujatha Das Gollapalli,* See-Kiong Ng
Institute of Data Science
National University of Singapore
{idssdg,seekiong}@nus.edu.sg

1 Data Collection

Candidate Named Entities: We used the Stanza library from Stanford[1] for annotating named-entity information from the SQuAD passages. Among the 18 named-entity types, we only considered entity-types PERSON, NORP (Nationalities/religious/political group), FAC (Facility), ORG (Organization), GPE (Countries/cities/states), LOC (Location), PRODUCT, EVENT, WORK_OF_ART, and LANGUAGE for which entity information is easier to obtain from knowledge resources in contrast with entity types that refer to concepts such as MONEY, CARDINAL, DATE.

Filtering Entity Search Results: We used the Entity Search API from Google[2] and retrieved the top-20 results using the entity names as query strings. For each query, the API provides a ranked list of results and each result element comprises of (name, type, description, detailed-description) information of the entity. Partial JSON output for the query “Taylor Swift" is shown in Figure 1 for illustration. The SQuAD dataset[3] contains questions and their answer spans marked in passage contexts. The passage containing the entity mention can be used for filtering out irrelevant search results using similarity thresholds and type-match constraints.

1. *Similarity-based matching*: For ambiguous names (such as “Michael Jordan"[4]), similarity between the source passage and the “detailed description" field can be use to identify the correct entity result assuming that the passage has other relevant information pertaining to the entry. After normalizing the text (lowercase, remove punctuation and stopwords), we picked the most similar result according to cosine similarity of the descriptive fields from the search result and the SQuAD passage after applying a similarity threshold of value 0.1.



```
"@type": "ItemList",
"itemListElement": [
  {
    "@type": "EntitySearchResult",
    "result": {
      "@id": "kg:/m/0dl567",
      "name": "Taylor Swift",
      "@type": [
        "Thing",
        "Person"
      ],
      "description": "Singer-songwriter",
      "image": {
        "contentUrl": "https://t1.gstatic.com/images?q=tbn:ANd9GcQmVDAhjhWnN2OWys2ZMO3PGAhupp5tN2LwF_BJ
        "url": "https://en.wikipedia.org/wiki/Taylor_Swift",
        "license": "http://creativecommons.org/licenses/by-sa/2.0"
      },
      "detailedDescription": {
        "articleBody": "Taylor Alison Swift is an American singer-songwriter and actress. Raised in Wyo
        "url": "http://en.wikipedia.org/wiki/Taylor_Swift",
        "license": "https://en.wikipedia.org/wiki/Wikipedia:Text_of_Creative_Commons_Attribution-ShareA
      },
      "url": "http://taylorswift.com/"
    },
    "resultScore": 4850
  }
]
```

Figure 1: Output from Google’s Entity Search API

2. *Entity-type matching*: For a query such as “America", the results from Entity Search API also include those of type VisualArtwork (a sculpture) as well as the country in North America. Knowing the entity-type of “America" as referenced in the SQuAD passage can help in discarding irrelevant results. Though the entity types in the search results from Google Entity Search are different from the NER type scheme followed in the Stanza library, and we do not have information on their types due to proprietary reasons,[5] we devised a set of match rules for handling equivalent types in both the entity type schemes for the SQuAD related searches. For example, “creative types" from Google includes “Book, Music Album, Soft-

[1] https://stanfordnlp.github.io/stanza/

[2] https://developers.google.com/knowledge-graph

[3] We used SQuAD version 1.1 as is the practice for QG.

[4] https://en.wikipedia.org/wiki/Michael_Jordan_(disambiguation)

[5] https://en.wikipedia.org/wiki/Knowledge_Graph

ware Application, Video Game", etc. up to twenty types. All these types are considered to be equivalent to the Stanza types “PRODUCT" or “WORK_OF_ART".

Using the two filtering steps above, were able to obtain about $62,473$ entity descriptions for the SQuAD dataset. The precision of these two heuristic filtering rules was found to be $\sim$ 97% based on the subset of about 1600 entity descriptions examined by the crowdworkers.

2 Simple Bigram Approximate Kernel

Let $A_b^i = \{d_A^i, t_A^i, h_A^i\}$ represent the set of dependency bigrams or the edges in the dependency parse tree for a sentence A. d_A^i and h_A^i represent respectively, the dependent and head nodes of the i-th bigram and t_A^i refers to the type of the dependency edge. Given the bigrams for two sentences A and B, the Simple Bigram Approximate Kernel (Özateş et al., 2016) defines similarity between them using the following formula:

$$SBAK\ (A,B)\ = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} sim(A_b^i, B_b^j)}{m+n}$$

In the above formula, m and n are the number of words in A and B respectively and

$$sim(A_b^i, B_b^j) = [s(d_A^i, d_B^j) + s(h_A^i, h_B^j)] * q(t_A^i, t_B^j)$$

The function $s(a,b) = 1$ if a and b match and 0, otherwise whereas q can be set based on the dependency edge type. Thus, two bigrams can get a partial score even if only the heads or dependent nodes match and based on how q is set, even when edge types do not match. Thus the SBAK similarity function is able to assign an approximate score for partially matching dependency bigrams. We used the list of dependency relations such as possessive, prepositional modifier, determiner etc. identified as not very useful by Ozates, et al (Özateş et al., 2016) with q set to 0.5 when the dependency edge types do not match and 1 otherwise in our computation.

3 Fact Extraction for Entities

The entity description fields are long and often comprise multiple sentences. Based on empirical observation that crowdworkers only use parts of these descriptions or specific “facts" from these descriptions during paraphrasing, we employed the following steps to extract facts from the entity descriptions and map the best matching fact with a specific paraphrase instance.

1. If the description has multiple sentences, perform co-reference resolution to replace all references to the entity mention with the actual entity name in all sentences.[6]

2. For each sentence:

   - Extract propositions using MinIE.[7]
   - Extract noun phrases and combine them with entity name to form “NP facts".

3. The set of sentences, propositions, and NP facts comprise the list of facts for a given entity description.

Entity Description: International Business Machines Corporation is an American multinational technology company headquartered in Armonk, New York, with operations in over 170 countries.
-International Business Machines Corporation is American multinational technology company -American multinational technology company be headquartered in Armonk in over QUANT_O_1 countries -American multinational technology company be headquartered in Armonk -International Business Machines Corporation is American multinational technology company headquartered in Armonk with operations in over QUANT_O_1 countries -Armonk is in New York

Table 1: Propositions extracted by MinIE

Example propositions from MinIE for the “IBM" example in the paper are shown in Table 1. To further improve the recall of facts, we also combined short noun phrases with the entity name to form “NP facts" as well as used full sentences.

The objective of fact extraction is to extract parts of the entity descriptions that best match a specific paraphrase for creating an instance for *FIQG*. Not only is addressing inaccuracies and completeness of fact extraction beyond the scope of this work, in on-going work, we are investigating how the entire text and multiple questions can be handled within the model using semantic features based on dependency parse and semantic role labeling information instead of this intermediate step of fact extraction.

References

Şaziye Betül Özateş, Arzucan Özgür, and Dragomir Radev. 2016. Sentence similarity based on dependency tree kernels for multi-document summarization. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2833–2838, Portorož, Slovenia.

[6] https://demo.allennlp.org/coreference-resolution

[7] https://github.com/uma-pi1/minie