Review Analysis Interview Subject14

**Demographics**

**Title** Research Scientist

**Company** Company1

**Education** PhD in engineering

**Experience in data analysis** 10 years (all text)

**Experience in review analysis** 1 year

**Gender** Male

## Summary

**Project Goals**

Help people search for hotels using “viewpoints” (perspectives) identified from a review corpus. People don’t know what they’re looking for in a new place, so they aim to give ideas about what kind of hotel they might want, via conversational bot.

These “viewpoints” come from reviews because they reflect real experiences.

**Data**

Uses proprietary reviews (over 14 million)

Dumps in a database using python.

Cleaning:

Excludes reviews that are not Japanese using some heuristics (ex: no hiragana)

Some reviews are truncated, these he excludes.

**Labeling Tasks**:

Finding candidate sentences to label: use a sentence as a seed to find similar sentences (via these embeddings: [A Simple but Tough-to-Beat Baseline for Sentence Embeddings](https://openreview.net/forum?id=SyK00v5xx)).

1. binary annotations of whether the sentence is relevant to the hotel or not
2. Cluster sentences to find candidates for entailment
   1. This is the second labeling task: annotated entailment of two phrases (ex: I enjoyed the clean room → the room was clean)
3. Iterative clustering makes hierarchy of reviews
   1. Considers these clusters to be aspects. Defined ~6 aspects this way (buses, room service, location, etc.).

Downstream:

* the user will input some rough requirements, and the system can find what aspects users like/care about through dialogue. The user interface is still an open question.

Quality

* Unsure of the quality of labels, but enough labels that the quality is sufficient.
* Before crowdsourcing, he does his own annotations. This helps him find errors in the instructions.
* Still, labeling is iterative (updates rules after some batches).
* Debugging the labels:
  + Uses the annotation agreement to evaluate the quality and clarity of instructions.
  + Sometimes fixes annotated examples by himself
  + Uses the jackknife method-- debug annotated corpus by training model on 4/5s and test on the last part. Repeat four times. Useful for finding errors because the classification score is low for marginal cases.

**Training data size:**

3 systems:

* One uses all data (14 million) for similarity search indexing
* Binary classifier, whether a phrase is about a hotel (9k sentences)
* Entailment model: 31k examples (every pair of sentences get a label. Some are duplicated because not all sentences are unique.)

**How do you debug errors in the model?**

Evaluates the model on the test set.

The subject iterates on cleaning the training data (more than on the BERT model).

**Is scalability a problem?**

No.

**What tools and languages do you use?**

Python

Yahoo! Japan crowdsourcing (cheaper and easier than say AltAir)

Pytorch or Tensorflow

BERT (for searching similar phrases)

Data stored on google cloud (migrated the needed sentences from bigQuery)

**How do you present your results or collaborate?**

Other people use the dumped sentences (knowing which are relevant to the hotel is useful for many downstream tasks).

Meetings once a week. Makes some google slides.

* Some classification accuracy values, some examples to explain results

**What are the bottlenecks you run into or the things you spend the most time on? (Are they different?)**

The subject originally didn’t know that the corpus had incomplete sentences. Noticed it when the output had incomplete sentences, and then had to spend a significant amount of time fixing it

Preprocessing time.

* **What features/tools do you wish you had?**

User interface to see phrases would be useful to find errors. Checking reviews in the browser means a lower chance of finding errors.

It could be helpful to have a tool for auditing labels.

Ex: flagging annotation errors