Stage 4 Report

April 14, 2017

1 Stage 4: Combining Two Tables

Trang Ho, Thomas Ngo, Qinyuan Sun

1.1 Pipeline

In this project stage, we have two tables AOM and WHED with schema as following: - WHED(**a_id**, a_name, a_city, a_prov, a_country, a_web) - AOM(**person_id**, a_name, a_city, a_prov, a_country, a_email_server)

The AOM table contains information on affiliation name, city, province/state, country, and email server. The information is manually provided by conference attendants; and consequently, information on the AOM table may be incomplete, inconsistant, or inaccurate. For example, conference attendants might provide affiliation names at the school level (e.g. Wisconsin School of Business) instead of those at the university level (e.g. University of Wisconsin - Madison)

On the other hand, the WHED table contains standardized information on affiliation name, city, province/state, country, and website domain. We therefore aim to map each individual's affiliation in the AOM data to an affiliation in the WHED data, and to keep the affliation information in WHED table as part of the merging.

In the stage 3, we have applied the entity matching to WHED and AOM tables to obtain a list of matching tuples for individuals and affliations. In this stage 4, we narrow down our list to the US only, which will tentatively be analyzed in the subsequent stage. The list of matched tuples can be found in file matched_tuples.csv (refer to the below for the directory of the file).

To merge the two tables, we use the information in WHED as the anchor for affiliations. Hence, we keep the columns in WHED and remove all columns related to affiliations in AOM. The final table has the following schema:

- MergedTable(**person_id**, a_id, a_name, a_city, a_prov, a_country, a_web)

File directory:

- The final table (i.e. table E): merged_tuples.csv
- The set of matches between AOM and WHED (i.e. table A & B): matched_tuples.csv
- The Python script that you used to merge the two tables AOM and WHED:

cs838-spring2017/stage4/src/merge.py

1.2 Statistics of Merged Table

```
In [1]: import py_entitymatching as em
        MergedTable = em.read_csv_metadata('merged_tuples.csv', key = 'person_id')
        print("Number of tuples:", MergedTable.shape[0])
        print("Number of columns:", MergedTable.shape[1])
```

```
Metadata file is not present in the given path; proceeding to read the csv file.
```

```
Number of tuples: 3230
Number of columns: 7
```

```
In [12]: MergedTable.head(n = 4)
```

Out[12]:

	a_id	person_id	a_name	a_city	a_prov	a_country	a_web
0	26	6378	abilene christian university	abilene	texas	united states	http://www.acu.edu
1	26	33444	abilene christian university	abilene	texas	united states	http://www.acu.edu
2	110	4676	adelphi university	garden city	new york	united states	http://www.adelphi.edu
3	110	8429	adelphi university	garden city	new york	united states	http://www.adelphi.edu

1.3 Code for Merging

```
In [ ]: import py_entitymatching as em
        df = em.read_csv_metadata('matched_tuples.csv', key = '_id')
        # aom = em.read_csv_metadata(path_to_csv_dir + '_aom.csv', key = 'person_id')
        # whed = em.read_csv_metadata(path_to_csv_dir + '_whed.csv', key = 'a_id')
        # df.head()

        #use rename_col() to rename columns
        #use drop_cols() to drop merged colums
        # modify df to get the final results
        drop_list = ['rtable_a_name','rtable_a_city','rtable_a_prov','rtable_a_country','rtable_a_email
        df = em.drop_cols(df, drop_list)

        df = em.rename_col(df,'ltable_a_id','a_id')
        df = em.rename_col(df,'ltable_a_name','a_name')
        df = em.rename_col(df,'ltable_a_city','a_city')
        df = em.rename_col(df,'ltable_a_prov','a_prov')
        df = em.rename_col(df,'ltable_a_country','a_country')
        df = em.rename_col(df,'ltable_a_web','a_web')
        df = em.rename_col(df,'rtable_person_id','person_id')


        # only one tuple in WHED should be matched to one tuple in aom.
        df_new = df.drop_duplicates(subset=['person_id'], keep = False)
        em.set_key(df_new,'person_id')
        df_new = em.drop_cols(df_new,'_id')
        df_new.head(n = 5)
        df_new.to_csv('merged_tuples.csv', index=False)
```