GSDF : Document and User Manual

Granular Spatial Data Fusion (GSDF) project tries to match the same location, which have been described differently in different sources by managing apparent inconsistencies between them. This project uses aviation accident data as an example to illustrate ideas. Its objective is to determine those accidents that occurred in the same locations. To this end, required datasets were selected from DBPedia and Kaggle datasets.

Note: to run the scripts and programs, please use MS SQL Server (2014 or higher).

1. DBPedia and Aircrash Data:

Aviation accident extracted from two data sources (DBPedia, Kaggle) and imported to relational databases.

Each data source has its own database.

To Create DBPedia database run “sqlDBPedia.sql” script.

To Create Kaggle database run “sqlAircrash.sql” script. We named this database as **Aircrash**.

1.1 Structure of the databases:

Both databases have similar structure:

Table	**Description**
SourceData	Data was extracted from DBPedia via its SPARQL endpoint “Site” field contains location of accident. This field has been tokenized into “Location” table. Data from Kaggle downloaded as a csv file. “Location” field contains location of accident. This field has been tokenized into “Location” table.
Location	Tokens of “Site” or “Location” field. The tokens was searched in GeoNames by using “search” web service
GeonameMain	Result of “search” web service call
GeoFound	Start id (sid) and end id (eid) in “GeonameMain” for each location
GeoNameHier	Result of “hierarchy” web service call
AlternativeName	Alternative names of locations (imported from GeoNames dump data)

1.2 Geocode the locations

Call “SP_GranularMaxCovering” stored procedure to geocode a location. Parameter of this procedure is @idLoc, which indicate to a location id. This location id can be obtained from “SourceData” or “Location” tables.

e.g. EXEC [dbo].[SP_GranularMaxCovering] @idLoc = 11

2. GSDF Database:

All geocoded locations of DBPedia and Kaggle are stored in the GSDF database.

The accident locations from two data sources are matched by using GSDF stored procedures and functions.

To Create GSDF database run “sqlGSDF.sql” script.

Note:

Prefix “air” in name of objects point to “Aircrash” data.

Prefix “dbp” in name of objects point to “DBPedia” data.

Prefix “airdbp” in name of objects point to pair data of both data sources.

2.1 Table Description:

Table	**Description**
finalresultAirCrash	Aircrash geocoded data
finalresultDBPedia	Dbpedia geocoded data
airAlternativeName	Alternative names of aircrash location
dbpAlternativeName	Alternative names of dbpedia location
GeonameHier	Hierarchy information about locations (both aircrash and dbpedia)
airdbpParentChild	Pair locations of aircrash and dbpedia that have parent-child relationship based on hierarchy information
airSelected	Selected locations from aircrash dataset for matching process (based on country of locations) Each selection phase make a new “round”
dbpSelected	Selected locations from dbpedia dataset for matching process (based on country of locations) Each selection phase make a new “round”
MapLocationGoogle	Coordinates of location based on Google Map API call
MapLocation	Coordinates of location based manual search on Google Map. This task was done by a student team.
Neighbours	List of countries and their neighbours with land or maritime border.
GroundTruth	Shows real match between pair locations of aircrash and dbpedia which is determined manually after data blocking

	This task was done with the help of a student team by using “MapLocationGoogle” and “MapLocation” tables.
GroundTruthUnique	Unique form of “GroundTruth” table For each pair of locations one record has been selected.
SelectedSim	Contains all selected pair of locations and its similarities: • Toponym similarity (LevSim) Related Procedure: Z090-selectedLevSim • Geographical similarity (DistSim) which was calculated of based on Distance, Related Procedures: Z060-selectedSim UpdateDistance Z070-selectedSim UpdateDistSim • Hierarchical similarity (GranSim) based on “GeonameHier” table Related Procedures: Z050-selectedSim UpdateParentChild Z080-selectedGranSim
QualityMetric	Contains • Result of matching algorithm (FinalSim) Related Procedure: Z094-QualityMetric Append Related Function: FinalSim • Real match from “GroundTruth” table (Match) • Threshold value for similarity (T) • True Positive (TP) • False Positive (FP) • False Negative (FN)
Threshold	Threshold values for similarity measurement. In order to remove sensitivity to the similarity threshold value, the experiments were done for a range of threshold values, which lie inside [0, 1] with an interval of 0.05.
airdbpCountryNull	Determines quality metrics for locations, which there is no information about their country.

2.2 Programs

All programs have a numbered prefix to determine the order of operations. In addition to these programs, there are other programs that have been used internally and so there is no distinct explanation for them in this document.

Program Type **P:** Stored procedure **F:** Functions (used internally) **V:** View

Program Type	**Program Name**	**Descriptions**
P	Z010-airSelected Make	Selects locations from “aircrash” dataset for a new round of algorithm execution based on country information.
P	Z011-dbpSelected Make	Selects locations from “dbpedia” dataset for a new round of algorithm execution based on country information
P	Z020-selectedSim Append	Makes pair of new selected locations from “aircrash” and “dbpedia”. In this procedure, “fn_haveGap” function determines that two locations in selected pair have gap or fall into same block.
P	Z030-airMapLocation Append	Adds new selected locations from “aircrash” to “MapLocation” table.
P	Z031-dbpMapLocation Append	Adds new selected locations from “dbpedia” to “MapLocation” table.
V	Z032-mapLocation Query	Shows new appended locations into “MapLocation” table to user. User can determine coordinates of locations based on Google Map API or his/her search on Google Map.
P	Z040-GroundTruth Append	Adds new pairs of locations to “GroundTruth” Table
V	Z041-GroundTruth Query	Shows new pairs in “GroundTruth” table. User can determine that the pair indicate a “match” or “no match”
P	Z042-Finish GroundTruth	Finalizes “GroundTruth” table in current round of execution
P	Z043-GroundTruthUnique Append	Removes duplicate records from “GroundTruth” table
P	Z050-selectedSim UpdateParentChild	Sets “ParentChild” field in “selectedSim” table from hierarchy information. A null value indicates there is no parent-child relation between two locations.

P	Z060-selectedSim UpdateDistance	Calculates “Distance” in “selectedSim” table based on coordinates of locations.
P	Z070-selectedSim UpdateDistSim	Calculates geographical distance similarity (“DistSim” in “selectedSim” table) for each pair of location based on “Distance” field.
P	Z080-selectedGranSim	Calculates hierarchy similarity (“GranSim” in “selectedSim” table) for each pair of location based on hierarchy information.
P	Z090-selectedLevSim	Calculates “LevSim” in “selectedSim” table based on toponym string similarity for each pair of location. Main names and alternative names have been considered.
P	Z094-QualityMetric Append	Determines result of matching process: • True Positive (TP) • False Positive (FP) • False Negative (FN) by using “FinalSim” function
P	Z095-result Query	Shows result of matching algorithm (Precision, Recall , F-Score) for all threshold values in “Threshold” table
P	Z100-Finish SelectedSim	Finishes current round of execution

Note 1: To show the gradual processing property of the proposed algorithm, the data have been processed at different rounds. Each round has its own ID. In the simulations, in each round the locations of a country are added to the in-process dataset.

Note 2: The blocking algorithm determines which pair of locations should be evaluated precisely. This idea has been implemented in “Z020-selectedSim Append” procedure by using “fn_haveGap” function.

The “fn_haveGap” function uses neighboring countries and latitude/longitude for data blocking.

3. Sample Scenarios

In this section, sample scenarios are presented to show how the algorithms works.

Note:

The uploaded database contains only part of the data, not all of them to make it faster to download and test. So just, use round IDs 50 to 60 in running the following procedures

3.1 Calculating Distance

For distance calculating, we used “Z060-selectedSim UpdateDistance” procedure. To show what is calculated we wrote similar procedure “Z061-selectedSim ShowDistance”.

Command: EXEC [dbo].[Z061-selectedSim ShowDistance] @round = 54

Columns:

Column	Description
airID	ID of aircrash location in “finalresultAirCrash” table
dbpID	ID of dbpedia location in “finalresultDBPedia” table
airLocation	Title of aircrash location
airLat	Latitude of aircrash location
airLng	Longitude of aircrash location
dbpLocation	Title of dbpedia location
dbpLat	Latitude of dbpedia location
dbpLng	Longitude of dbpedia location
Distance	Geographical distance between two locations (Calculated by “calcGeoDistance” function)

Sample Results:

airID	dbpID	airLocation	airLat	airLng	dbpLocation	dbpLat	dbpLng	Distance
459	47	Cairo, Egypt	30.06263	31.24967	Cairo International Airport, Egypt	30.12194	31.40556	16
1746	4	Near Aswan, Egypt	24.09082	32.89942	Aswan_International_Airport	23.96436	32.81997	16
1834	47	Near Cairo, Egypt	30.06263	31.24967	Cairo International Airport, Egypt	30.12194	31.40556	16
1751	47	Near Ayayda, Egypt	30.36082	31.50907	Cairo International Airport, Egypt	30.12194	31.40556	28
3822	185	Zifta, Egypt	30.7142	31.24425	Ityai el Barud, Egypt	30.86667	30.66667	57
2556	185	Near Wadi Natrun, Egypt	30.43785	30.19499	Ityai el Barud, Egypt	30.86667	30.66667	65

3.2 Calculating Distance Similarity

For “Distance Similarity” calculating, we used “Z070-selectedSim UpdateDistSim” procedure. To show what is calculated we wrote similar procedure “Z071-selectedSim ShowDistSim”.

Command: EXEC [dbo].[Z071-selectedSim ShowDistSim] @round = 54

Columns:

Column	**Description**
airID	ID of aircrash location in “finalresultAirCrash” table
dbpID	ID of dbpedia location in “finalresultDBPedia” table
Distance	Geographical distance between two locations (Calculated by “calcGeoDistance” function)
DistSim	Distance Similarity : if [Distance]>100 then 0 else (100-[Distance])/100

Sample Results:

airID	**dbpID**	**Distance**	**DistSim**
459	47	16	0.84
1746	4	16	0.84
1834	47	16	0.84
1751	47	28	0.72
3822	185	58	0.42
2556	185	66	0.34
3822	47	67	0.33
2662	185	81	0.19
1751	185	98	0.02
2049	47	99	0.01
1834	185	105	0
459	185	105	0
2556	47	122	0
1541	47	148	0

3.3 Calculating Granularity Similarity

For “Granularity Similarity” calculating, we used “Z080-selectedGranSim” procedure. To show what is calculated we wrote similar procedure “Z081-selectedGranSim”

Command: EXEC [dbo].[Z081-selectedGranSim] @round = 54

Columns:

Column	Description
airID	ID of aircrash location in “finalresultAirCrash” table
dbpID	ID of dbpedia location in “finalresultDBPedia” table
airLocation	Title of aircrash location
dbpLocation	Title of dbpedia location
GranSim	Hierarchy similarity of two location by using “HierCommonFinal” function

Sample Results:

airID	dbpID	airLocation	dbpLocation	GranSim
1746	4	Near Aswan, Egypt	Aswan_International_Airport	0.8
1834	47	Near Cairo, Egypt	Cairo International Airport, Egypt	0.8
459	47	Cairo, Egypt	Cairo International Airport, Egypt	0.8
2556	185	Near Wadi Natrun, Egypt	Ityai el Barud, Egypt	0.8
3828	4	Beni Sueif, Egypt	Aswan_International_Airport	0.75
3828	185	Beni Sueif, Egypt	Ityai el Barud, Egypt	0.75
3828	47	Beni Sueif, Egypt	Cairo International Airport, Egypt	0.75
3822	47	Zifta, Egypt	Cairo International Airport, Egypt	0.6
2875	47	Off Sharm el Sheikh, Egypt	Cairo International Airport, Egypt	0.6
2847	47	Off Port Said, Egypt	Cairo International Airport, Egypt	0.6

3.4 Calculating Toponym Similarity

For “Toponym Similarity” calculating, we used “Z090-selectedLevSim” procedure. To show what is calculated we wrote similar procedure “Z091-selectedLevSim”.

Command: EXEC [dbo].[Z091-selectedLevSim] @round = 54

Columns:

Column	Description
airID	ID of aircrash location in “finalresultAirCrash” table
dbpID	ID of dbpedia location in “finalresultDBPedia” table
airLocation	Title of aircrash location
dbpLocation	Title of dbpedia location
LevSim	String similarity between location names (main name and alternative names of location) by using “ToponymSim” function

Sample Results:

airID	dbpID	airLocation	dbpLocation	LevSim
459	4	Cairo, Egypt	Aswan_International_Airport	0.4
802	4	El Arish, Egypt	Aswan_International_Airport	0.333333
1437	4	Luxor, Egypt	Aswan_International_Airport	0.5
1541	4	Menzalah Lake, Egypt	Aswan_International_Airport	0.3
1746	4	Near Aswan, Egypt	Aswan_International_Airport	1
1751	4	Near Ayayda, Egypt	Aswan_International_Airport	0.291667
1803	4	Near Bir Lahfan, Egypt	Aswan_International_Airport	0.25
1834	4	Near Cairo, Egypt	Aswan_International_Airport	0.4
1950	4	Near El-Thamad, Egypt	Aswan_International_Airport	0.294118
2049	4	Near Isma'iliya, Egypt	Aswan_International_Airport	0.294118

3.5 Calculating similarity of two locations and comparing with ground truth data

“Z093-QualityMetric” procedure has been written to show the results ,which is actually similar to “Z094-QualityMetric Append” procedure.

Command: EXEC [dbo].[Z093-QualityMetric] @round = 54

Columns:

Column	Description
airID	ID of aircrash location in “finalresultAirCrash” table
dbpID	ID of dbpedia location in “finalresultDBPedia” table
airLocation	Title of aircrash location
dbpLocation	Title of dbpedia location
FinalSim	Calculated similarity by “FinalSim” function
RP	Indicates match/no_match value based on user decision (stored in "GroundTruth” table) 0 for no_match and 1 for match
T	Threshold value for similarity
P	if FinalSim > Threshold then 1 else 0
TP	True positive (if P=1 and RP=1 then 1 else 0)
FP	False positive (if P=1 and RP=0 then 1 else 0)
FN	False Negative (if P=0 and RP=1 then 1 else 0)

Sample Results:

airID	dbpID	airLocation	dbpLocation	Final Sim	RP	T	P	TP	FP	FN
459	4	Cairo, Egypt	Aswan_International_Airport	0.414	0	0.75	0	0	0	0
459	47	Cairo, Egypt	Cairo International Airport, Egypt	0.914	1	0.75	1	1	0	0
459	185	Cairo, Egypt	Ityai el Barud, Egypt	0.384	0	0.75	0	0	0	0
802	4	El Arish, Egypt	Aswan_International_Airport	0.394	0	0.75	0	0	0	0
802	47	El Arish, Egypt	Cairo International Airport, Egypt	0.372947	0	0.75	0	0	0	0
802	185	El Arish, Egypt	Ityai el Barud, Egypt	0.379714	0	0.75	0	0	0	0
1437	4	Luxor, Egypt	Aswan_International_Airport	0.444	0	0.75	0	0	0	0

3.6 Final result

To view final results run: EXEC [dbo].[Z095-result Query]

Columns:

Column	**Description**
T	Threshold values
NSTP	Sum of True Positive (TP)
NSFP	Sum of False Positive (FP)
NSFN,	Sum of False Negative (FN)
Precision	TP/(TP+FP)
Recall	TP/(TP+FN)
F_Score	(2*Precision*Recall)/(Precision + Recall)

Output:

T	**NSTP**	**NSFP**	**NSFN**	**Precision**	**Recall**	**FScore**
0.05	997	115779	0	0.009	1.000	0.017
0.1	997	115764	0	0.009	1.000	0.017
0.15	997	115705	0	0.009	1.000	0.017
0.2	997	115498	0	0.009	1.000	0.017
0.25	997	111856	0	0.009	1.000	0.018
0.3	996	79702	1	0.012	0.999	0.024
0.35	982	30355	15	0.031	0.985	0.061
0.4	973	11266	24	0.079	0.976	0.147
0.45	960	5307	37	0.153	0.963	0.264
0.5	950	1927	47	0.330	0.953	0.490
0.55	942	1091	55	0.463	0.945	0.622
0.6	936	765	61	0.550	0.939	0.694
0.65	927	472	70	0.663	0.930	0.774
0.7	898	252	99	0.781	0.901	0.837
0.75	869	100	128	0.897	0.872	0.884
0.8	690	63	307	0.916	0.692	0.789
0.85	574	49	423	0.921	0.576	0.709
0.9	493	44	504	0.918	0.494	0.643
0.95	236	16	755	0.937	0.238	0.380
1	167	5	824	0.971	0.169	0.287

3.7 Python codes

To compare results with other algorithms, a number of techniques referenced in previous works have been implemented in python 3.7.

Code Name	**Technique**	**Training Dataset**	**Output Dataset**
probit reg2	Probit Regression	F2.csv	ProbitReg2.csv
probit reg3		F3.csv	ProbitReg3.csv
linear+poly reg2	Linear Regression Polynomial Regression	F2.csv	LineReg2.csv PolyReg2.csv
linear+poly reg3		F3.csv	LineReg3.csv PolyReg3.csv
ANN2	Artificial neural network	F2_10.csv	MLP2.csv
ANN3		F3_10.csv	MLP3.csv

Note 1: In used datasets, there are common columns:

1. DistSim: Geographical Distance similarity
2. LevSim: Toponym similarity based on Levenshtein distance
3. GranSim: Hierarchy similarity

“DistSim” and “LevSim” were used in “probit reg2”, “linear+poly reg2”, and “ANN2”

“DistSim”, “LevSim”, and “GranSim” were used in “probit reg3”, “linear+poly reg3”, and “ANN3”.

Note 2: Input file of all implemented techniques is “selectedSimEval.txt” which is read by programs after training phase. This file contains only part of the data (100 000 records) to make it faster to download and test.

Note 3:

Degree of polynomial regression is 4.

Settings of ANN used in the implementation are:

Activator: Rectifier,   Hidden Layer: 8,8,8,   solver: adam