Open City Data Pipeline

Collecting, Integrating, and Predicting Open City Data

Stefan Bischof[1,2], Christoph Martin[2], Axel Polleres[2], and Patrik Schneider[2,3]

[1] Siemens AG Österreich, Vienna, Austria
[2] Vienna University of Economics and Business, Vienna, Austria
[3] Vienna University of Technology, Vienna, Austria

Abstract. Having access to high quality and recent data is crucial both for decision makers in cities as well as for informing the public, likewise, infrastructure providers could offer more tailored solutions to cities based on such data. However, even though there are many data sets containing relevant indicators about cities available as open data, it is cumbersome to integrate and analyze them, since the collection is still a manual process and the sources are not connected to each other upfront. Further, disjoint indicators and cities across the available data sources lead to a large proportion of missing values when integrating these sources. In this paper we present a platform for collecting, integrating, and enriching open data about cities in a re-usable and comparable manner: we have integrated various open data sources and present approaches for predicting missing values, where we use standard regression methods in combination with principal component analysis to improve quality and amount of predicted values. Further, we re-publish the integrated and predicted values as linked open data.

1 Introduction

Nowadays governments have large collections of data available for decision support. Public administrations use these data collections for backing their decisions and policies, and to compare themselves to other cities, and likewise infrastructure providers like Siemens could offer more tailored solutions to cities based on these data. Having access to high quality and current data is crucial to advance on these goals. Studies like the Green City Index [8] which assess and compare the performance of cities are helpful, in particular for public awareness. However, these documents are outdated soon after publication and reusing or analyzing the evolution of their underlying data is difficult. To improve this situation, we need regularly updated data stores which provide a consolidated, up-to-date, view on relevant open data sources for such studies.

Even though there are many relevant data sources which contain quantitative based indicators about cities available as *open data*, it is still cumbersome to collect, clean, integrate, and analyze data from these sources: obstacles include different indicator specifications, different languages, formats, and units. Example sources of city data include DBpedia, Geonames, or the UrbanAudit data set included in Eurostat; Urban Audit [4] for example, provides over 250 indicators on several domains for 258 European cities. Furthermore, several larger cities provide data on their own open data portals, e.g.,

[4] http://ec.europa.eu/eurostat/web/cities/

London, Berlin, or Vienna.[5] Data is published in different formats such as RDF, XML, CSV, XLS, or just as HTML tables. The specifications of the individual data fields – (i) how indicators are defined and (ii) how they have been collected – are often implicit in textual descriptions only and have to be processed manually for understanding.

Moreover, data sources like Urban Audit cover many cities and indicators (e.g., population), but show a large ratio of *missing values* in their data sets. The impact of missing values is even aggravated when we combine different data sets, since there is a fair amount of disjoint cities and indicators across those data sets, which makes them hard to integrate. Our assumption though – inspired also by works that suspect the existence of quantitative models behind the working, growth and scaling of cities [1] – is that most indicators in such a scoped domain have their own structure and dependencies, from which we can build prediction models:[6] we aim to evaluate different "standard" regression methods to choose the best fitting model to predict missing indicator values. We follow two approaches for computing such predictions. The first approach is based on a selection of "relevant" indicators as predictors for a target indicator. The second approach constructs the principal components (PCs) of the "completed" data sets (missing values are replaced with "neutral' values [18]), which are then used as predictors. We also compare both approaches according to their performance, prediction accuracy (estimated root mean square error), and coverage (the number of possible predictions).

Contributions and Structure. Our concrete contributions are:

- We analyze and integrate several data sources including DBpedia, Urband Audit, and UNSD Demographic and Social Statistics;
- We provide a system architecture for an "Open City Data Pipeline" including a crawler, wrappers, and ontology-based integration components;
- We evaluate two prediction approaches for filling in missing values, where we combine different standard regression methods and PCs to maximize prediction accuracy;
- We re-publish the integrated and predicated values as linked open data (LOD).

Section 2 describes the imported data sources and the challenges arising when processing/integrating their data. Section 3 presents an architecture overview of the "Open City Data Pipeline" and a lightweight extensible ontology used therein. Section 4 explains methods we used for predicting missing values as well as corresponding prediction error rates. Our Linked Open Data interface to republish the integrated and predicted data is documented in Section 5. Section 6 concludes with several possible future extensions.

Related Work. *QuerioCity* [11] is a platform to integrate static and continuous data with Semantic Web tools. Although it uses partly similar technologies, it works as a single city platform and not as data collection of many cities. While QuerioCity concentrates on data integration, we focus on predicting missing values, and publishing the outcomes as Linked Data. The EU project *CitySDK*[7] provides unifying APIs, including a Linked Data API for mobility and geo data usable across cities. These reusable APIs enable developers to create portable applications and ease service provisioning

[5] http://data.london.gov.uk/, http://daten.berlin.de/, http://data.wien.gv.at/

[6] We refer to "predicting" instead of "imputing" values when we mean finding suitable approximation models to predict indicators values for cities and temporal contexts where they are not (yet) available. These predictions may (not) be confirmed, if additional data becomes available.

[7] http://www.citysdk.eu/

for city administrators. If enough cities adopt CitySDK, its APIs can become a valuable data source for the City Data Pipeline as well. Regarding the methods, the approaches *FeGeLOD* [15] and *Explain-a-LOD* [14] of Paulheim et al. are closely related, however both focus on unsupervised data mining of unspecified features from linked data instead of filling in missing values for specific attributes. Also related is the work by Nickel et al. [12] which focuses on relational learning, i.e., rather learning object relations than predicting numeric attribute values.

2 Data Sources

The Open City Data Pipeline's database contains data ranging from the years 1990 to 2014, but most of the data concerns the years after 2000. Not every indicator is covered over all years, where the highest overlap of indicators is between 2004 and 2011. Most European cities are contained in the Urban Audit data set, but we also include the capital cities and cities with a population over 100 000 from the United Nations Demographic Yearbook (UNYB). [8]

Before integration, locations have varying names in different data sets (e.g., Wien vs. Vienna), a Uniform Resource Identifier (URI) for every city is essential for the integration and enables to link the cities and indicators back to DBpedia and other LOD data sets. We choose to have an one-to-one (functional) mapping of every city from our namespace to the English DBpedia resource, which in our republished data is encoded by `sameAs` relations. We identify the matching DBpedia URIs for multilingual city names and apply a semi-automated technique with three steps using the city's names from Urban Audit and UNYB:

- Accessing the DBpedia resource directly and following possible redirects;
- Using the Geonames API [9] to identify the DBpedia resource;
- For the remaining cities, we manually looked up the URL on DBpedia.

DBpedia. DBpedia, initially released in 2007, is an effort to extract structured data from Wikipedia and publish the data as Linked Data [4]. For cities, DBpedia provides various basic indicators such as demographic and geographic information (e.g., population, latitude/longitude, elevation). The Open City Data Pipeline extracts the URLs, weather data, and the population of a city. While we only integrated a limited subset of indicators from DBpedia for now, we plan to add other indicators like economic and spatial indicators in the future. Since temporal validity of indicators is rarely documented, so we can just assume them to be current, as accessed.

Urban Audit. The Urban Audit (UA) collection started as an initiative to assess the quality of life in European cities. It is conducted by the national statistical institutes and Eurostat. Currently, data collection takes place every three years (last survey in November 2012) and is published via Eurostat. [10] All data is provided on a voluntary basis which leads to varying data availability and missing values in the collected data sets. Urban Audit aims to provide an extensive look at the cities under investigation, since its a policy tool to the European Commission. "The projects' ultimate goal is

[8] http://unstats.un.org/unsd/demographic/products/dyb/dyb2012.htm

[9] http://api.geonames.org/

[10] http://ec.europa.eu/eurostat

Table 1: Urban Audit Data Set

Year(s)	Cities	Indicators	Filled	Missing	% of Missing
1990	177	121	2 480	18 937	88.4
2000	477	156	10 347	64 065	85.0
2005	651	167	23 494	85 223	78.4
2010	905	202	90 490	92 320	50.5
2004 - 2012	943	215	531 146	1 293 559	70.9
All (1990 - 2012)	943	215	638 934	4 024 201	86.3

Table 2: United Nations Data Set

Year(s)	Cities	Indicators	Filled	Missing	% of Missing
1990	7	3	10	11	52.4
2000	1 391	147	7 492	196 985	96.3
2005	1 048	142	3 654	145 162	97.5
2010	2 008	151	10 681	292 527	96.5
2004 - 2012	2 733	154	44 944	3 322 112	98.7
All (1990 - 2012)	4 319	154	69 772	14 563 000	99.5

to contribute towards the improvement of the quality of urban life" [13]. At the city level, Urban Audit contains over 250 indicators and are divided into the categories Demography, Social Aspects, Economic Aspects, and Civic Involvement.

United Nations Statistics Division (UNSD). The UNSD offers data on a wide range of topics, for example, on education, environment, health, technology, and tourism. Our main source is the UNSD Demographic and Social Statistics, [11] which is based on the data collected annually (since 1948) by questionnaires to national statistical offices. The UNSD data marts consist of the following topics: population by age distribution, sex, and housing; occupants of housing units / dwellings by broad types (e.g., size, lighting, etc.); occupied housing units by different criteria (e.g., walls, waste, etc.).

The collected data has over 650 indicators, wherein we dropped the most fine-grained indicator level to keep a balanced set of indicators in favor of keeping a more course-grained set of indicators, e.g., keeping *housing units total* instead of *housing units by room size*. However, for future work it would be interesting to split these indicators and calculate new indicators regarding their granularity.

Future Data Sources. At the point of writing, the data sources are strongly focused on European cities and demographic data. Hence, we need to integrate further national and international data sources. A promising candidate is the *County and City Data Book* (CCDB) of U.S. Census Bureau. [12] The CCDB offers two data sets; one covering Area and Population, Crime, Government and Climate for cities larger then 20 000 inhabi-

[11] http://unstats.un.org/unsd/demographic/default.htm

[12] https://www.census.gov/statab/ccdb/ccdbcityplace.html

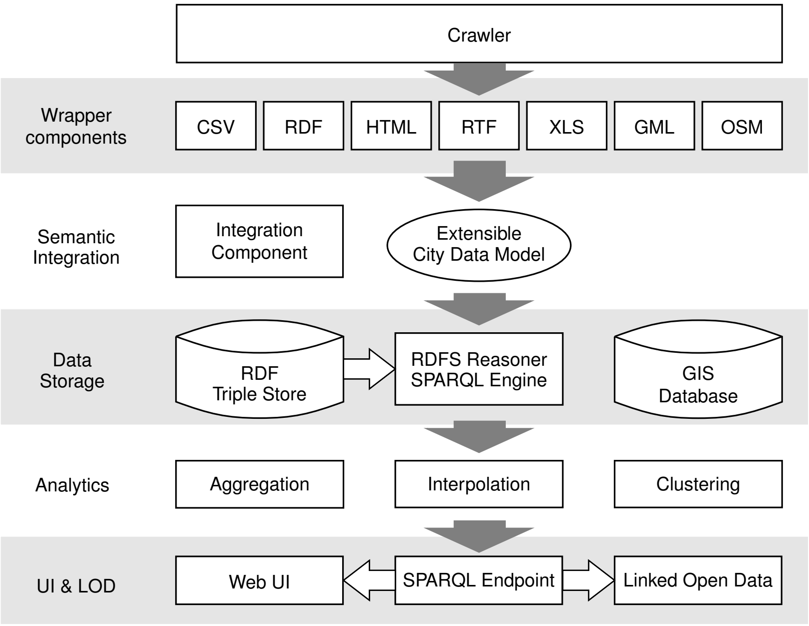


Fig. 1: City Data Pipeline architecture showing components for crawling wrapping, cleaning, integrating, and presenting information

tants; and another covering Population by Age, Sex, and Race, Education, Income and Poverty for locations with 100 000 population and more.

The Carbon Disclosure Project (CDP) is an organization based in the UK aiming at "[...] using the power of measurement and information disclosure to improve the management of environmental risk" [7]. One of several environmental governance projects of CDP is *CDP cities* introduced in 2011. In this project, CDP has data collected on more than 200 cities worldwide. CDP cities offers a reporting platform for city governments using an online questionnaire covering climate-related areas like Emissions, Governance, Climate risks, Opportunities, and Strategies.

3 System Architecture

The City Data Pipeline collects data, organizes this data into indicators, and shows these indicators to the user. This section introduces the system which is organized in several layers (see Figure 1): *crawler*, *wrapper components*, *semantic integration*, *data storage*, *analytics*, and *external interfaces* (user interface, SPARQL endpoint, and linked data).

Crawler. The City Data Pipeline semi-automatically collects data from various registered open data sources in a periodic manner dependent on the specific source. The crawler currently collects data from 32 different sources, e.g., DBpedia, UN open data, Urban Audit, as well as data sets of several cities. Adding new data sources is still a manual process, where the mapping of the usually tabular data has to be provided by mapping scripts. However, a semi-automatic process would be an appealing extension for future work.

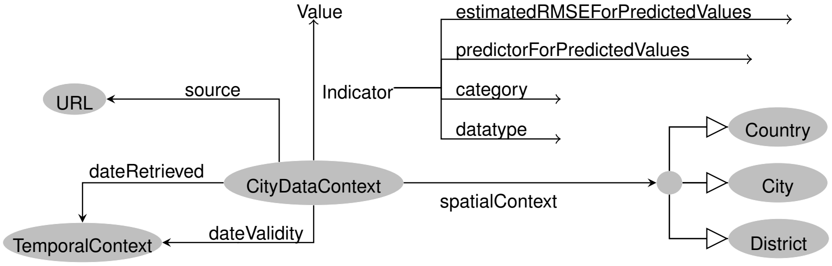


Fig. 2: Excerpt of the City Data Model ontology

Wrapper Components. As a first step of data integration, a set of wrapper components parses the downloaded data and converts it to a source specific RDF. The set of wrapper components include a CSV wrapper to parse and clean CSV data, a wrapper for extracting HTML tables, a wrapper for extracting tables of RTF documents, a wrapper for Excel sheets, and a wrapper for cleaning RDF data as well. All of these wrappers are customizable to cater for diverse source-specific issues. These wrapper components convert the data to RDF and preprocess the data before integrating the data with the existing triple store. Preprocessing contains the usual data cleansing tasks, unit conversions, number and data formatting, string encoding, and filtering invalid data.

Semantic Integration (Ontology). To be able to access a single KPI such as the population number, which is provided by several data sources, the semantic integration component *unifies the vocabulary* of the different data sources through an ontology (see Figure 2). The semantic integration component is partly implemented in the individual wrappers and partly by an RDFS [6] ontology (extended with capabilities for reasoning over numbers by using equations [2]) called*City Data Model*. [13] The ontology covers several aspects: spatial context (country, region, city, district), temporal context (validity, date retrieved), provenance (data source), terms of usage (license), and an extensible list of indicators.

Indicator is the super property of all the indicator properties mapping CityDataContexts to actual values. Each Indicator of the ontology contains, a name, description, a unit of measurement, a data type, and is grouped into one of the following *categories*: (a) Demography, (b) Social Aspects, (c) Economic Aspects, (d) Training and Education, (e) Environment, (f) Travel and Transport, (g) Culture and Recreation, and (h) Geography. To integrate the source specific indicators the ontology maps data source specific RDF properties to City Data Model properties, e.g., it maps dbpedia:population to citydata:population by an RDFS subPropertyOf property. A CityDataContext is an anchor connecting a set of data points to a spatial context, a temporal context, and a data source. When importing an input CSV file containing the indicators as columns and the cities as rows then each row corresponds to (at least) one CityDataContext. The SpatialContext class collects all resources with spatial dimension especially, country, province, region, city, and district. Depending on the available data the ontology allows storage of data points at each of these levels. Furthermore entities of differ-

[13] http://citydata.wu.ac.at/ns#

ent granularity can be connected by the property locatedIn. The dateValidity property maps a CityDataContext to a point in time where the values are valid. Additionally the property periodValidity can indicate what the validity period is (possible values are biannual, annual, quarterly, monthly, weekly, daily, hourly or irregular). Whereas the dateRetrieved property records the date and time of the data set download. The source property links a CityDataContext to the corresponding data source.

Data Storage. To store the processed data we use Jena TDB[14] as *triple store* for RDF data, and PostGIS/PostgreSQL as a *GIS database* for geographic information. GIS databases allow us to compute missing information such as areas of cities or districts, or lengths of certain paths. Subsequent subsystems can access the RDF data via a SPARQL interface. The SPARQL engine provides RDFS reasoning support by query rewriting (including reasoning over numbers [2]).

Analytics, UI & LOD. The analytics layer includes tools to fill in missing data by using statistical methods. Section 4 describes the missing value prediction in detail. The results are also stored in the RDF triple store and the SPARQL engine provides access to them. Section 5 explains the frontend, user interface, SPARQL endpoint, and publishing data as Linked Open Data. Bischof et al. [3] describe the system components in more detail.

4 Prediction of Missing Values

After integrating the different sources, we discovered a large number of missing values in our data sets. We identified two reasons for that:

- As shown in Table 1 and 2, we can observe a large ratio of missing values due to incomplete data published by the data providers;
- More severely, when we combine the different data sets even more missing values are introduced, since there is a fair amount of disjoint cities and indicators.

Base Methods. Our assumption is that every indicator has its own distribution (e.g., normal, Poisson, etc.) and relationship to other indicators. Hence, we aim to evaluate different regression methods and choose the best fitting model to predict the missing values. We measure the prediction accuracy by comparing the root mean squared error in % (RMSE%) [21] of every regression method:

$$\mathit{RMSE}\% = \left(\frac{\sqrt{\frac{\sum_{t=1}^{n}(y_t - y'_t)^2}{n}}}{y_{max} - y_{min}} \right) \times 100 \qquad (1)$$

where n is the amount of predictions, y_t is the observed (actual) value on t, ${y'}_t$ is the predicted value on t, and y_{max} (resp. y_{min}) the maximum (resp. minimum) value of the observed values.

In the field of Data Mining [21,10] various regression methods for prediction were developed. We chose the following three “standard” methods for our evaluation due to their robustness and general performance.

[14] http://jena.apache.org/documentation/tdb/

K-Nearest Neighbour Regression (KNN) denoted as M_{KNN} is one of the most wide-spread data mining techniques applied in a variety of domains. As stated in [10], the algorithm is simple, easily understandable and reasonably scalable. KNN can be used in variants for clustering as well as regression.

Multiple Linear Regression (MLR) denoted as M_{MLR} has the goal to find a linear relationship between one target and several predictor variables. The linear relationship can be expressed as a regression line through the data points, where the most common approach is *ordinary least squares* to measure & minimize the cumulated distances [10].

Random Forest Decision Trees (RFD) denoted as M_{RFD} involve the top-down segmentation of the data into multiple smaller regions represented by a tree with decision and leaf nodes. Each segmentation is based on splitting rules, which are tested on a predictor. Decision nodes have branches for each value of the tested attribute and leaf nodes represent decision on the numerical target. A random forest is generated by a large number of trees, which are build according to a random selection of attributes at each node. We use the algorithm introduced by Breiman [5].

Preprocessing. The preprocessing starts with the extraction of the base data set from our RDF triple store. We use SPARQL queries with the fixed period of *2004–2011* and produce an initial data set as a matrix with tuples of the form $\langle$*City*, *Indicator*, *Year*, *Value*$\rangle$. Based on the initial matrix, we perform a basic *entity recognition* similar to Paulheim et al. [15] as follows:

- Removing boolean and nominal columns, as well as all weather related data and sub-indicators in the U.N. data set, e.g., *occupants of housing units with 2 rooms*;
- Merging the dimensions year & city, resulting in $\langle$*City Year*, *Indicator*, *Value*$\rangle$;
- Transposing the initial matrix by moving the indicators into the columns, resulting in tuples of the form $\langle$*City Year*, $\mathit{Indicator}_1\mathit{Value}, \ldots, \mathit{Indicator}_n\mathit{Value}\rangle$;
- Deleting columns and rows which have a missing values ratio of 90%.

Our initial data set from UA, UN, and DBpedia contains 3 399 cities with 370 indicators. After performing the first three steps, we have 237 indicators left. By merging city and year and transposing the matrix we create 13 482 city/year rows. And after deleting the cities/indicators with a missing values ratio of 90% we have the final matrix of 4 438 rows (city/year) with 207 columns (indicators).

Approach 1 - Building Complete Subsets. In the first approach, we try to build models for a target indicator by directly using the available indicators as predictors. For this, we are using the correlation matrix of the data to find indicators which are suitable predictors. Subsequently, we build a complete subset from our data, i.e. we first perform a projection on our data table, keeping only the predictors and the specific target as columns. More detailed, our approach has the following steps on the initial data set, the matrix A_1 and a fixed number of predictors n (we test this approach on different $n's$):

1. Select the target indicator I_T;
2. Calculate the corr. matrix A_C of A_1 between I_T and the remaining indicators;
3. Create the submatrix A_2 of A_1 with I_T and the n "best" indicators (called the predictors). The predictors are selected according to the highest absolute correlation coefficients in A_C;
4. Create the complete matrix A_3 by deleting all rows in A_2 with missing values;

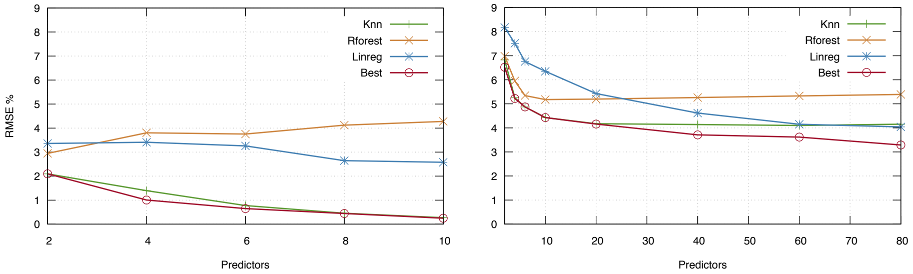
5. Apply *stratified tenfold cross-validation* (see [21]) on A_3 to obtain ten training- and test sets. Then, train the models M_{KNN}, M_{MLR}, and M_{RFD} using the training sets. Finally, calculate the mean of the ten $RMSE\%$ based on the test set for each model and choose the best performing model M_{Best} accordingly;
6. Use method M_{Best} to build a new model on A_2 to predict the missing values of I_T.

The performance of the regression methods were evaluated for two to ten predictors. Two regression methods have their best RMSE% with ten indicators: 0.27% for KNN and 2.57% for MLR. Whereas RFD has the best RMSE% of 4.12 with eight indicators. Figure 3a gives an overview of the results. By picking the best performing regression for every indicator (red line) the avg. RMSE% can be reduced only slightly. For ten predictors the avg. RMSE% improves to 0.25% over KNN with 0.27%. Depending on n, we fill in between 122 056 for ten and 296 069 values for two predictors. For a single city and ten predictors, the number of predicted values range from 7 to 1 770. The limited number of filled in values is due to the restriction of using the complete matrix for the regression methods.

Approach 2 - Principal Component Regression. In the second approach, we omit the direct use of indicators as predictors. Instead, we first perform a Principal Component Analysis (PCA) to reduce the number of dimensions of the data set and use the new compressed dimensions, called *principal components* (PCs) as predictors. As stated in [10], the PCA is a common technique for finding patterns in data of high dimensions. Parts of the evaluation is similar to Approach 1, but we have an additional step where we impute all the missing values with *neutral* values for the PCA. The neutral values are created according to the *regularized iterative PCA algorithm* described in [18]. This step is needed to perform the PCA on the entire data set. The following steps are evaluated having an initial data set A_1 as a matrix and a predefined number of predictors n (we test this approach also on different $n's$):

1. Select the target indicator I_T;
2. Impute the missing values in A_1 using the regularized iterative PCA algorithm resulting in matrix A_2 and remove the column with I_T;
3. Perform the PCA on the A_2 resulting in a matrix A_3 of a maximum of 80 PCs;
4. Append the column of I_T to A_3 creating A_4 and calculate the correlation matrix A_C of A_4 between I_T and the PCs;
5. Create the submatrix A_5 of A_4 on the selection of the PCs with the highest absolute correlation coefficients and limit them by n;
6. Create submatrix A_6 of A_5 for validation by deleting empty rows for I_T;
7. Apply stratified tenfold cross-validation on A_6 with the Step 5 from Approach 1, which results in the best performing model M_{Best};
8. Use the method M_{Best} to build a new model on A_5 (not A_6) for predicting the missing values of I_T.

Figure 3b shows the RMSE% for the different methods and different number of predictors. On average over all indicators, KNN works best closely followed by MLR. However, for 80 predictors MLR performs best with an avg. RMSE% of 4.04%, where KNN has an avg. RMSE% of “only” 4.15%. The RFD algorithm provides reasonable results for a lower number of predictors, but starts yielding worse results for 20 predictors and more. MLR improves steady up to 80 predictors. KNN is performing best up to



(a) Approach 1 (Building Complete Subsets)

(b) Approach 2 (PCA Regression)

Fig. 3: Prediction results

70 predictors. As assumed, the overall results improve by selecting the best performing regression method for each indicator. The red line in Figure 3b shows the avg. RMSE% with the best regression method chosen. We also can see, that the results improve with more predictors reaching the best result of 3.29% with 80.

As mentioned, we have two properties to evaluate the quality of our approaches. First, it is important to build models which are able to predict many (preferably) all missing values. Second, the prediction accuracy of the models is essential, since the Open City Data Pipeline can fulfill its purpose of publishing high-quality, accurate data and predictions. Prediction accuracy in Approach 1 is higher, which we relate to the reduced size of the data set. However in Approach 1, we fill in at the maximum 296 069 values with 2 predictors (having an avg. RMSE% of 2.01%), which is about 66% of Approach 2. Due to the reduced number of predictions, we will apply Approach 2 for publishing the filled in values.

5 Publishing as Linked Data

City Data Context. The resources (cities) and properties in the City Data namespace (http://citydata.wu.ac.at/) are published according to the linked data principles. The ontology (as described in Section 3), contains all City Data property and class descriptions. Each city is assigned a dereferencable URI, e.g., http://citydata.wu.ac.at/resource/Ljubljana for the capital of Slovenia. Depending on the HTTP `Accept` header the server will return either an HTML, RDF/XML, or Turtle representation after a HTTP 303 redirect. The city resources are linked to the Linked Open Data cloud via owl:sameAs to the corresponding DBpedia resources.

Prediction Data. The whole prediction is based on the present data in the triple store. The *preprocessing* is written in Python and *prediction* and *evaluation* is developed in R [16] using "standard" packages. As mentioned before, we only publish the predicted values from Approach 2. After the best regression method is selected for a particular indicator, we use this method to fill in all the missing values and publish them as a *new indicator with a prefix* in the CityDataContext. The threshold for publishing is an avg. RMSE% of 30% which leads to 28 indicators (e.g. *price of a m3 of domestic water euro*) being dropped. We also add the the source and the year for the prediction. We then introduce two new properties for each indicator describing the quality of the data by the avg. RMSE% and the regression method used. In future work, we aim to publish the data using the PROV Data Model [9].

Interface. A simple Java powered web interface allows users to select exactly which subset of the data should be shown. The interface provides programmatic access via HTTP GET to allow external tools such as data visualization frameworks, to query the database. The web application communicates with the Jena triple store via SPARQL 1.1. Users can select one or more of the 450 *indicators*. The list also shows how many data points are available per indicator and for how many cities data points are available for this indicator. Next the user can select one or several of more then 5 260 *cities* for which we collected data. For a few cities we even have information on the individual districts available. In these cases the user can select one or several of the districts. Optionally the user can specify a *temporal context*, for which year the database should be queried. This feature allows to compare several cities with each other at a certain point of time instead of listing data of all available times.

6 Conclusions and Future Work

In this paper we have presented the *City Data Pipeline*, an extensible platform for collecting, integrating, and predicting open city data from several data providers including DBpedia and Urband Audit. We have developed several components including a data crawler, wrappers, an ontology-based integration platform, and a missing value prediction module, which is a crucial component since we have sparse data sets. For this, we have developed two approaches, one based on predicting a target indicators directly from other indicators, and one based on predictors from components calculated by Principal Components Analysis (PCA). We applied for both approaches three basic regression methods (e.g., Multiple Linear Regression) and selected the best performing one. They were compared regarding the number of filled in values and prediction accuracy, concluding that the PCA-based approach will be used for future predictions. Filled in missing values are then published as linked open data for further use.

Our future work includes variants and extensions of the presented data sets, methods, and the system itself. Regarding the data sets, we already mention in Chapter 2 several sources, e.g., U.S. Census, which are needed to cover a wider range of cities worldwide. Regarding the methods, we have not yet investigated how the different data sets relate to each other, since our missing values prediction is based on the entire data set. It would be interesting to investigate, if indicators can be predicted from one data set to another. Crucial for this approach would be a source data set, i.e., Urband Audit, which has a high amount of overlapping cities with the target data sets. Further, we aim to extend our basket of base methods with other well established regression methods. Promising candidates are Support Vector Machines with a linear/non-linear kernel [19], Neural Networks or Bayesian Generalized Linear Model [20]. We also plan to develop a wrapper for OpenStreetMap (OSM) data sets and thus including Spatial Open Data. This opens up extensions for new indicators and analytical features. New indicators can be generated directly from the spatial data of OSM, e.g., generating the amount of public green space by aggregating all the parks. Regarding the analytical possibilites we could introduce *spatial relations* [17] including containment, neighboring, and disjointness, which gives us the possibility to query these relations but also aggregate the indicators accordingly. Furthermore, we are in the process of improving the user interface to make the application easier to use. For this we investigate several libraries for more advanced information visualization.

Acknowledgements. Stefan Bischof has been partially funded by the Vienna Science and Technology Fund (WWTF) through project ICT12-015.

References

1. Bettencourt, L.M.A., Lobo, J., Helbing, D., Kühnert, C., West, G.B.: Growth, innovation, scaling, and the pace of life in cities. Proc. of the National Academy of Sciences of the United States of America 104(17), 7301—7306 (2007)
2. Bischof, S., Polleres, A.: RDFS with Attribute Equations via SPARQL Rewriting. In: Proc. of ESWC 2013, pp. 335–350. Springer (2013)
3. Bischof, S., Polleres, A., Sperl, S.: City data pipeline. In: Proc. of the I-SEMANTICS 2013 Posters & Demonstrations Track. pp. 45–49 (2013)
4. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia - A crystallization point for the web of data. J. Web Sem. 7(3), 154–165 (2009)
5. Breiman, L.: Random forests. In: Machine Learning. pp. 5–32 (2001)
6. Brickley, D., Guha, R., (eds.): RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation (2004), http://www.w3.org/TR/rdf-schema/
7. CDP Worldwide: Carbon Disclosure Project - About Us. https://www.cdp.net/en-US/Pages/About-Us.aspx (January 2015)
8. Economist Intelligence Unit (ed.): The Green City Index. Siemens AG (2012)
9. Gil, Y., Miles, S.: Prov model primer. Tech. rep., W3C Note (Apr 2013)
10. Han, J.: Data Mining: Concepts and Techniques, Third Edition. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2012)
11. Lopez, V., Kotoulas, S., Sbodio, M., Stephenson, M., Gkoulalas-Divanis, A., Aonghusa, P.M.: Queriocity: A linked data platform for urban information management. In: Proc. of ISWC 2012, pp. 148–163. Springer (2012)
12. Nickel, M., Tresp, V., Kriegel, H.: Factorizing YAGO: scalable machine learning for linked data. In: Proc. of WWW 2012. pp. 271–280 (2012)
13. Office for Official Publications of the European Communities: Urban Audit. Methodological Handbook (2004)
14. Paulheim, H.: Generating possible interpretations for statistics from linked open data. In: Proc. of ESWC 2012. pp. 560–574 (2012)
15. Paulheim, H., Fürnkranz, J.: Unsupervised generation of data mining features from linked open data. In: Proc. of WIMS 2012. p. 31 (2012)
16. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008), http://www.R-project.org
17. Renz, J.: Qualitative Spatial Reasoning with Topological Information. Springer (2002)
18. Roweis, S.T.: EM algorithms for PCA and SPCA. In: Advances in Neural Information Processing Systems 10 (NIPS 1997). pp. 626–632 (1997)
19. Sanchez, V.: Advanced support vector machines and kernel methods. Neurocomputing 55(1–2), 5 – 20 (2003)
20. West, M., Harrison, P.J., Migon, H.S.: Dynamic generalized linear models and bayesian forecasting. Journal of the American Statistical Association 80(389), 73–83 (1985)
21. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, Third Edition. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2011)