QPredict: Using low quality volunteered geospatial data to evaluate high-quality authority data

A case study on building footprint data

TIMO HOMBURG, Mainz University Of Applied Sciences, Germany
STEFFEN STAAB, Universität Stuttgart & University of Southampton, Germany & UK
FRANK BOOCHS, Mainz University Of Applied Sciences, Germany

High-quality, typically administrative, geospatial data should adhere to established measurement and representation practices and be protected from malicious attacks. However, this kind of geospatial data may only be infrequently updated due to its often prolonged production process compared to a data source of volunteered geographic information such as OpenStreetMap. Existing approaches typically try to quality-assure geospatial data by comparing it to another reference data set of perceived higher quality - often another administrative dataset facing a similar update cycle. In contrast, this article tries to determine whether actual changes present in volunteered geographic information data such as OpenStreetMap, which also need to be applied in an administrative dataset (i.e., consists of actual changes in the real world), can be identified automatically. To that end, we present QPredict, a machine learning approach observing changes in volunteered geospatial data such as OpenStreetMap to predict issues with a target (administrative) data set. The algorithm is trained by exploiting geospatial object characteristics, intrinsic and extrinsic quality metrics and their respective changes over time. We evaluate the effectiveness of our approach on two data sets representing two mid-size cities in Germany and discuss our findings in terms of their applicability in use cases.

CCS Concepts: • **Information systems** → *Data structures*; **Geographic information systems**; • **Applied computing** → **Cartography**.

Additional Key Words and Phrases: Map Change Prediction, Machine Learning, Spatial Data Quality

ACM Reference Format:
Timo Homburg, Steffen Staab, and Frank Boochs. 2025. QPredict: Using low quality volunteered geospatial data to evaluate high-quality authority data: A case study on building footprint data. 1, 1 (January 2025), 36 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Reliable, high-quality geospatial data matters in legal affairs as well as in life-threatening situations. For example, firefighting or emergency rescue operations must not be hindered by incomplete geospatial data or low accuracy of buildings' geometries. Therefore, quality assurance for geospatial data constitutes a significant concern. In general, a variety of geospatial data sets might be used in critical situations, but whereas quality assurance is enforced for

Authors' addresses: Timo Homburg, Mainz University Of Applied Sciences, Lucy-Hillebrand Straße 2, Mainz, Germany, 55128, timo.homburg@hs-mainz.de; Steffen Staab, Universität Stuttgart & University of Southampton, Universitätsstraße 1 University Road, University of Southampton, 56075, Germany & UK, Steffen.Staab@ipvs.uni-stuttgart.de; Frank Boochs, Mainz University Of Applied Sciences, Lucy-Hillebrand Straße 2, Mainz, 55128, Germany, frank.boochs@hs-mainz.de.

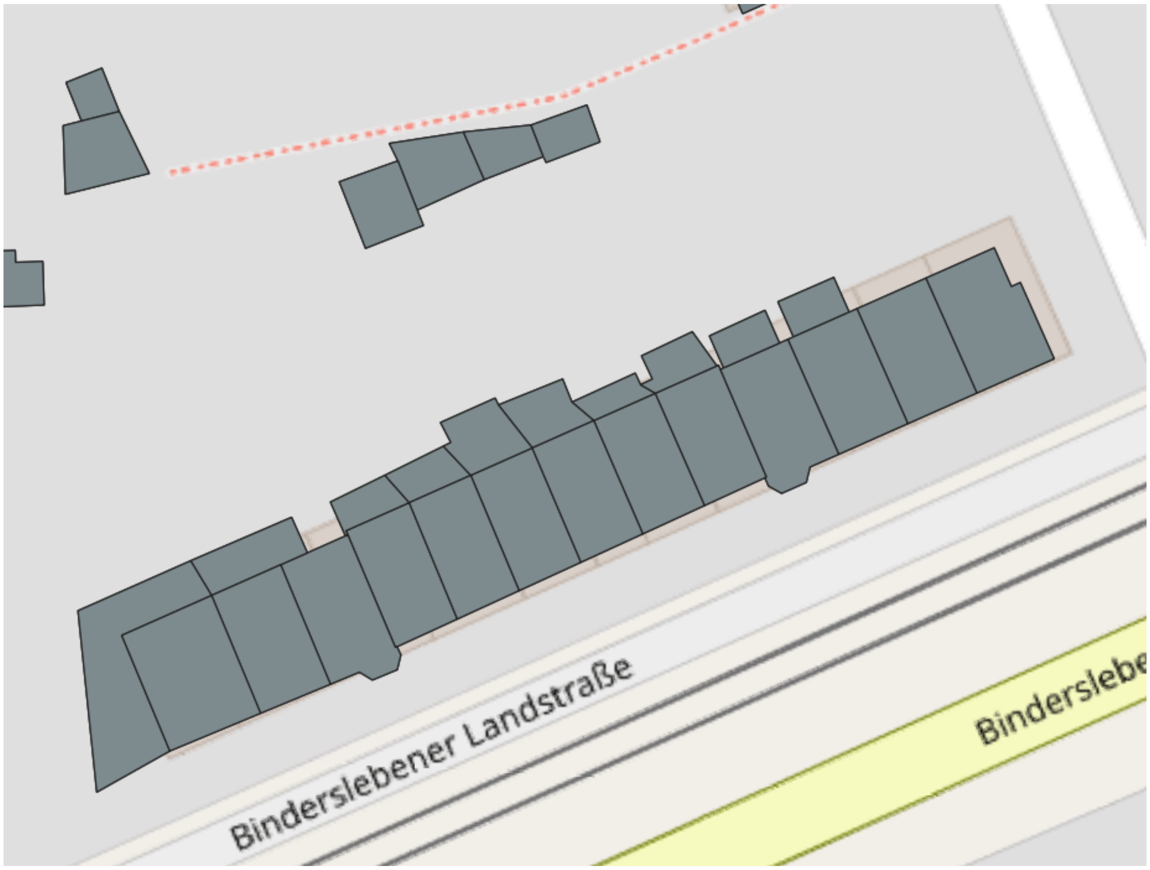


Fig. 1. Example of geometric inter-data-set inconsistency: Buildings in dark grey (target data set: mapping agency of Thuringia, Germany) and in light brown (source data set: OpenStreetMap). In this case, all buildings in the target data set also exist in the source data set. However, the geometries in the source data set were not modeled precisely, as can be observed by looking at light brown buildings that do not overlap 100% with the target dataset.

geospatial data sets issued by local, state, or national governments, i.e., by mapping agencies which we call managed geographic information (MGI), volunteered geographic information (VGI) [23, 57], such as OpenStreetMap (OSM) [51], can be modified by everyone. This exposes VGI data to potential risks, for example, malicious attacks, inaccuracies because of the use of non-professional measuring equipment, or wrong attributions because of a lack of experience of the VGI contributors. Hence, one might argue that VGI data alone is of limited use in emergency rescue operations. However, even reference data sets are far from perfect. In particular, their update regimen is limited by a variety of possible reasons, such as the availability of human resources to resurvey certain areas, budgetary requirements of the local mapping agencies, and a lack of legal requirements (updates may only be requested yearly by law), internal workflows of the mapping agency and possibly by technical limitations in the IT infrastructure of the individual mapping agency. This situation implies that various areas covered by MGI data that are the responsibility of the mapping agency may be at least temporarily outdated and thus of low quality.

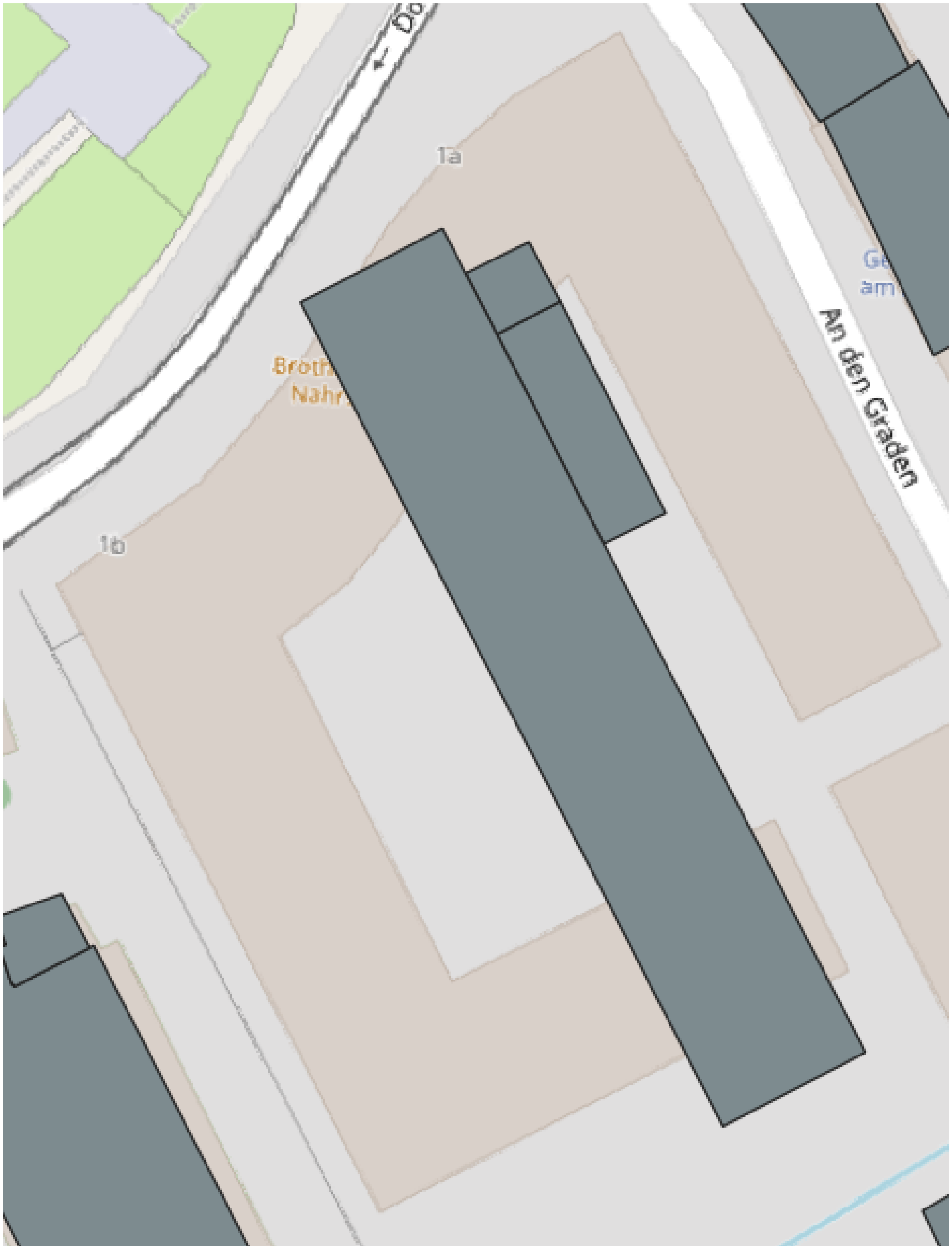


Fig. 2. Example of geometry deletion: Buildings in dark gray are included in the target data set, and other buildings in light brown and other colors in the source data set. Here, the building in gray has been demolished to make way for a new building project, which is already present in the source data set.

For example, Figure 1 depicts a real-world example where we can identify two kinds of datasets. The target dataset in dark grey is MGI data gathered by the mapping agency of Thuringia, Germany. The source dataset is colored light brown and represents equivalent buildings in OpenStreetMap. We call the MGI dataset the target dataset because this dataset depicts the target, which we would like to improve with our classification algorithm. We call the VGI dataset the source dataset because it represents a suspected more recent dataset with changes we would like to incorporate into the target dataset. One might argue that the target data set comprises high-quality MGI, whereas the source data set is VGI with less precise modeling. This difference in modeling might be rooted in the usage of less precise capturing equipment on the side of the OpenStreetMap community or from changes in the geometry that have occurred since its capture in the MGI dataset. If the latter case is true, the responsible mapping agency would like to identify this change to be adopted in the next revision of the target data set. If the former is true, the change could be considered not to be adopted.

Figure 2 displays another common situation. The target data set is outdated and shows an already demolished building. The source data set shows the more recent situation in which a new building has already been constructed in its place. This case will likely occur in VGI data and will later be adapted to official data sources.

For a mapping agency, the problem to be solved is to identify relevant and correct changes in VGI geospatial data and adapt them on time to their official datasets to give a timely and accurate picture of the real world to their customers or other state agencies.

To quality control MGI data for changes that might have happened in the world since its last update, we have approached the main research question of this paper: whether and how a *source data set* (e.g., OSM) of quality perceived lower than the *target data set* (e.g., local MGI geospatial data) could be used for assuring the quality of said target data set. In particular, we want to test if a machine learning classification can identify exactly those changes occurring in a VGI data source, which would at a later time be applied to an MGI data source curated by a mapping agency, i.e., the adaptability of VGI map changes to an MGI data source and to evaluate the performance of such a machine learning approach.

We suggest an original approach, QPredict, following a suggestion in the future work section of [63], that addresses this research question. Its context is depicted in Figure 3.

We assume that a target data set (MGI) and source data set (VGI) represent the same geographical region. Further, we assume that the former data set exists only at time point t_1, but the latter exists at time points t_1 and $t_2 (t_2 > t_1)$. QPredict then identifies changes the source data set underwent between time points t_1 and t_2. It relates the source data sets and these changes to the target data set at t_1 and decides which and how changes should be reflected in the target data set at t_2. We model the QPredict decision as a classification task, learning from historical data snapshots of source and target data sets at various time points.

In this endeavor, we are guided by the assumptions that:

(1) The adaptability of VGI data to MGI data is reflected in data quality metric results
(2) Patterns of data quality metric results repeat throughout the history of VGI and MGI data, reflecting the likelihood of a change to be adapted in MGI data

Related work has defined a wide variety of data quality metrics for geospatial data, e.g., in terms of completeness of geospatial data, the accuracy of geometries, or the (logical) consistency of geometries. Data quality metrics are distinguished as intrinsic data metrics [5] and extrinsic data metrics [22]. Intrinsic data quality metrics may point out inconsistent data configurations but are blind to actual world changes. Extrinsic data quality metrics require a reference

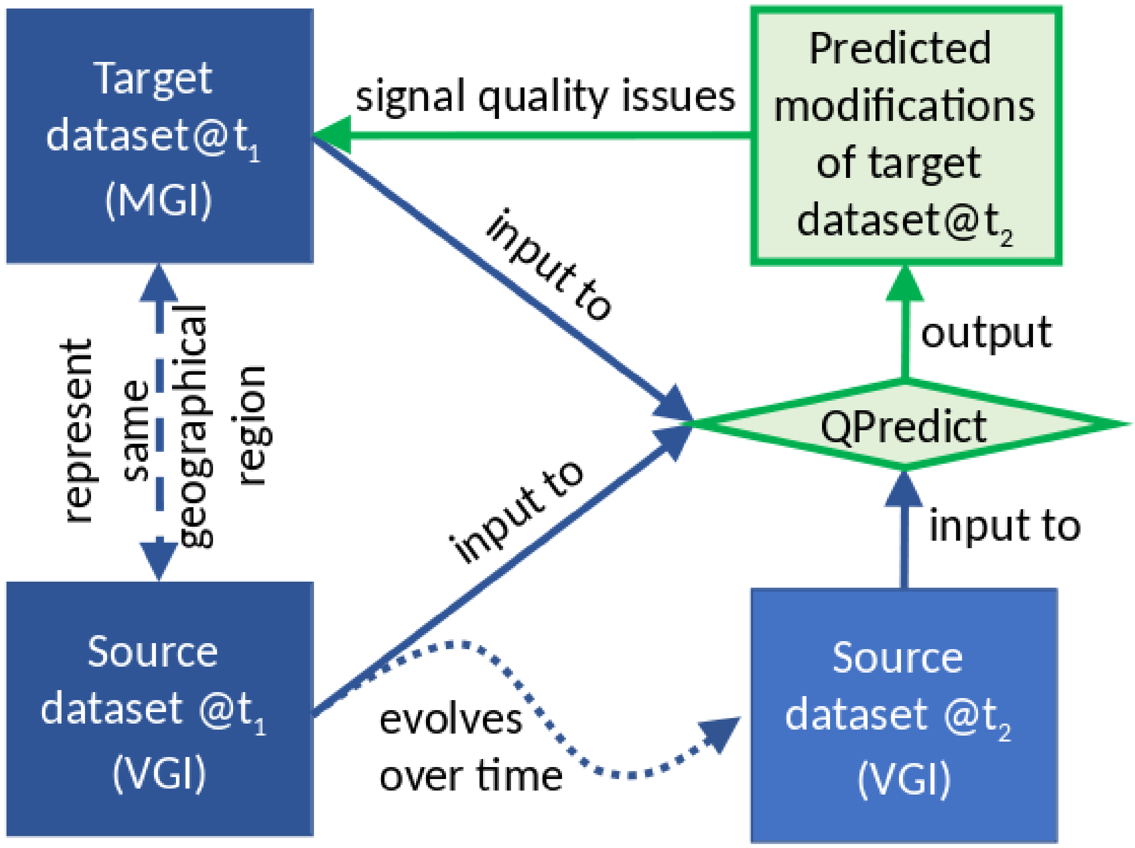


Fig. 3. Application context of QPredict

data set with superior quality to the target data set, which is to be judged. We want to judge the appropriateness of intrinsic, extrinsic, and a combination of both metric types for our classification task.

In the following, we recap the current state of the art of geospatial data quality in Section 2 and distinguish our approach from already existing map change detection approaches. In Section 3, we introduce the principal QPredict followed by a longer elaboration on the types of features we use in Section 4. In Section 5, we present two data sets of the state of Thuringia, Germany, which we use for training and testing QPredict, explaining the machine learning setup we use and our evaluation method. Finally, we discuss the results and limitations of our approach in Section 6 before concluding our work in Section 7.

2 STATE OF THE ART

This section gives an overview of geospatial data quality, geospatial change detection, map inference, and various categories of data quality metrics to measure geospatial data quality.

2.1 Geospatial Data Quality

[34] defines data quality as:

> Quality is the "degree to which a set of inherent characteristics fulfills requirements."

ISO19113 [35], ISO19114 [36], and ISO19157 [33] describe general principles of geospatial data quality and a framework of procedures for determining and evaluating quality. [9, 59] distinguish the data quality dimensions of Lineage [61]: The historical development of the data set, Positional Accuracy [11] of the geometric coordinates in comparison to a target data set, the Completeness [28] of the geospatial data set as referenced in a target data set or a data set specification, its Logical Consistency [38]: Topological consistency„ and attribute consistency of the data set, its Temporal Accuracy [8]: The accuracy of the measurement of time attributes, the temporal consistency of the data set attributes and the validity of data in a given time frame, its Thematic Accuracy: Correctness (correct syntactic annotation of the objects in the data set)[42] and its Semantic Accuracy [7, 53, 54] (correct semantic classification of the objects in the data set).

In addition to situation-specific data quality parameters, [31] suggest that the above categories of data quality can be assessed on any geospatial data set using different kinds of data quality metrics (intrinsic vs. extrinsic [10, 12]) (see Sections 2.2 and 2.3) [44] which are usually combined to achieve a certain data quality result.

2.2 Intrinsic Quality Metrics

Intrinsic geospatial data quality metrics include validity checks of geometries (e.g., well-formedness checks) [3] or geometry accuracy measurements. *Logical consistency* metrics [38] take into account other geometries in the same data set to find contradictions in the combination of attributes [17, 40]. *Vicinity metrics* [4] compare geospatial objects in the contexts of their neighborhoods using changeset analysis. By changeset, we mean a definition of the changes (additions, edits, removals) of a particular geospatial feature from one time point to the next, including meta information as depicted, for example, in OpenStreetMap.[1] In addition, *metadata* such as *lineage* information [61] and information about the data provider may be used to classify *trustfulness* [62]. *Temporal accuracy* [8] metrics may indicate the *freshness* of the data set. [10, 12] describe that intrinsic quality metrics hint at mistakes present in the current geometry. However, [4] suggests that "absolute statements on data quality are only possible with a high-quality reference data set as a basis for comparison." Following this statement, we rely on intrinsic data quality metrics as one source for identifying suitable changes to be adapted in VGI data. Intrinsic data quality metrics should, therefore, constitute an important part of a feature set for change classification.

2.3 Extrinsic Quality Metrics

Geospatial data quality may be extrinsically evaluated against geospatial data, which is considered a gold standard. Commonly, positional accuracy, shape similarity measurements, degrees of overlapping areas using comparative analysis, [19, 22] and a completeness analysis of attributes [44] are measured. Which data set should act as the gold standard data set may be argued about, as the mapping community of OpenStreetMap may, in many cases, produce more detailed and significantly more recent data of a geographical region. However, in this publication, we take the stance that OSM data, as the more recent dataset, will be the standard we are measuring against. Extrinsic data quality metrics can help to identify changes that have eventually been integrated into MGI data. We consider them the second set of indicators of map changes that must be covered in a machine-learning data set.

[1] https://wiki.openstreetmap.org/wiki/Changeset

2.4 Change Detection in Geospatial Data

In a survey conducted in [47], several national mapping agencies within Europe were asked about their data collection practices, particularly how they incorporate VGI data into their daily workflow. The results showed that VGI data is at least of interest and, at best, considered to update MGI data due to the nature of having more up-to-date data. However, before a national mapping agency applies data from VGI resources, they undergo significant checks by various resources, including but not exclusive to their staff surveying the area to be updated. In particular, reporting systems to update MGI data, change detection systems to show possible to-be-applied changes have been developed, and VGI data has been used to update place names and attribute data with a manual check by the mapping agency staff. VGI data is often only used for change detection, but this change detection may need to be manually revised by the mapping agency's staff. It is here that our approach QPredict can make an impact, as it provides a better assessment of changes that are relevant for a mapping agency to apply and might either lower the number of changes that need to be checked for relevance by the mapping agency staff or provide the opportunity to classify changes better. In the following, the types of change detections that the GIS community has conducted are presented and set into perspective to the QPredict approach of this article.

Even though map change detection is a common task in the GIS community, the targets of map change detections have mainly been aerial photographs. A typical classification task then tries to determine how or if an area has changed based on satellite images of different points in time [15, 21, 37, 39, 41]. Those classifications are not based on data quality metrics but rather on features intended for image recognition and constitute a different approach to ours. Therefore, indications of change detections are limited to observations taken from satellite images and cannot consider attributes added to the respective geometries, like, for example, in OpenStreetMap. Also, one needs to consider the costs involved in creating accurate satellite images, which are high compared to volunteered geographic information, which is easily and often freely accessible over the internet. Work on change classifications using machine learning of raster or vector data sets is not known to the authors and constitutes the main difference between QPredict and the aforementioned approaches, as we investigate this possibility by implementing and testing QPredict.

Further related work on change detection focused on updating older manually digitized maps using a perceived higher-quality data set. [24] aimed to superimpose German ATKIS (Amtliches Topographisch-Kartographische Informationssystem) [2] data on geological maps for data enrichment. This approach is different because two perceived high-quality data sources are used. The matching is restricted to one semantic class, and a data quality assessment has only been attempted to validate already matched geometries. They applied an Iterative Closest Point (ICP) change detection between ATKIS data to manually written and digitized geological maps. The results were used to justify a transfer of changes from the ATKIS data set to the digitized geological maps. Even though we would like to detect changes, we pursue the different goals of judging various geometry types of VGI data through data quality assessments for a possible application to MGI data.

The validity of changes in VGI data has been a subject of investigation by respective communities. Tools like OSM-Stats[2] examine the credibility of OpenStreetMap edits using a user contribution activity analysis to detect vandalism. Comparisons of a perceived higher quality data source to a perceived lower quality source (the opposite of our approach) are common in related work [45]. However, these publications aimed to improve the quality of the VGI data, either by detecting mistakes in VGI data or by comparing VGI data to a higher-quality data source. We want to improve MGI data from information gained from VGI data.

[2]http://www.osmstats.neis-one.org

A related area is the classification task of map inference in which vector geometries are inferred from GPS traces, [16]. [55], among others, explored several algorithms to achieve this. A data quality analysis of extrinsic data quality metrics is conducted to evaluate an automatically generated map from GPS traces. Common approaches like the holes and marbles method [6] provide a means to validate a map geometrically and topologically. In this article, we face a different challenge. While VGI data could be considered an uncertain data source and is often created from GPS traces, our paper does not try to infer a new map from given GPS data but instead tries to identify changes in given VGI data, which we often do not know their origin. However, our work connects to the map inferencing domain as inference maps are an alternative data source of a perceived lower quality VGI dataset, i.e., a possible input for our algorithm.

Finally, change detection between maps of varying scales has been performed. [50] performs change detection intending to update settlements on smaller-scale maps from larger-scale maps. They propose a formalized model to distinguish map discrepancies from changes caused by e.g. natural phenomena. This is a related problem and cannot be compared 1:1 as we consider the given maps to have the same scale and do not expect natural disasters to change our map in comparable ways.

[63] investigated the transfer of changes from a VGI data source to an authoritative data set. Often, data sets in VGI data are not aligned regarding accuracy, resolution, and data attributes, so such initial constraints hinder the matching and subsequent change detection process. To mitigate problems concerning the level of detail, the authors propose a set of rules that can be used to identify relevant map changes, i.e., a hierarchical model. This approach differs from ours, as it leaves out attributes and metadata when determining if a change is justified and relies on a set of geometry simplification processes to determine a match.

However, [63] suggests that future work may examine machine learning approaches to solve this particular task, which is precisely where QPredict comes in to attempt this approach. Hence, according to our knowledge, this is the first attempt to use a machine learning task to solve whether feature changes from a perceived lower-quality data source should be transferred to a higher-quality data source.

3 QPREDICT - A NEW APPROACH FOR CHANGE CLASSIFICATION

This section describes the QPredict algorithm using the application context given in Figure 3. This description contains the methodology of the algorithm Section 3.1 by first describing the preparation of corresponding geometries in VGI and MGI data in Section 3.2, then elaborating on how the classification works and how the algorithm is trained Sections 3.3 and 3.4. We illustrate this functionality with an abstract minimum example to explain its use in Section 3.5. Further details on the exact features and the experimental setup follow in Section 4 and Section 5.

3.1 Methodology overview

We now describe the methodology based on which QPredict operates. QPredict is given a source data set at time points t_1 and t_2 and a target data set at time point t_1 (cf. Figure 4). Both data sets depict the same geographical area. They include geospatial objects of building footprints, each containing a geometry (e.g., a polygon), a set of attributes (data describing the geospatial object), and a set of metadata (e.g., currentness) describing the metainformation of the geospatial object.

QPredict operates in two phases: *QPredict-train* and *QPredict-classify* as shown in Figure 5. *QPredict-train* learns how source data should be reflected in a target data set from historical data. *QPredict-classify* uses the classifier produced by QPredict-train and allows for applying classified changes from the lower quality source to the higher quality target

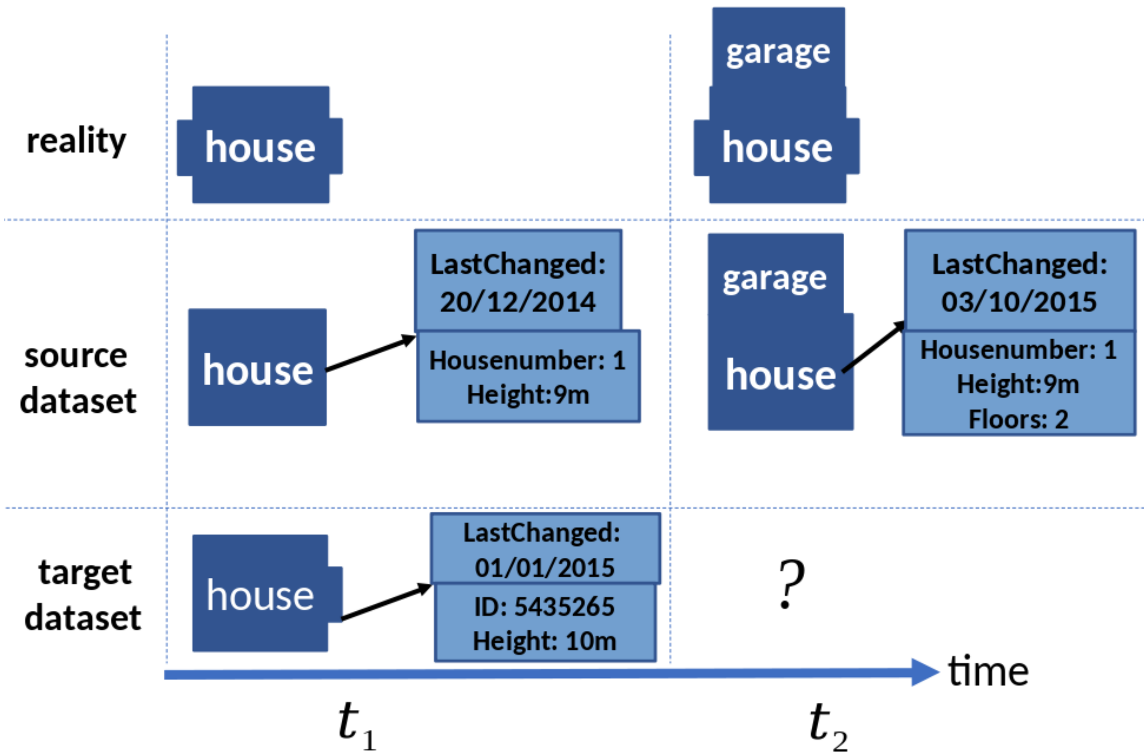


Fig. 4. Source and target data sets representing the real world at time point t_1 and a source data set representing the real world at time point t_2. Metadata and attributes are attached in light blue boxes. QPredict suggests how to change the target data set at t_2 to best represent the real world at t_2.

data set. In the following, we first present how we generate features for *QPredict-train* and *QPredict-classify*, we further discuss the application of *QPredict-classify*, and finally describe the classifier training using *QPredict-train*.

3.2 Change Identification

A preliminary step in the creation process of the feature sets used by *QPredict-train* and *QPredict-classify* for classification is identifying changes in the source and target datasets.

At first, corresponding geospatial objects in the data sets to be compared, i.e., target and source datasets, must be determined. We discuss this process in Section 3.2.1. Next, a change detection process must detect if geospatial objects differ and in which way. We discuss this process in Section 3.2.2. This constitutes the basis for generating feature vectors, which configuration we introduce in Section 3.4.

3.2.1 Change Identification I: Finding corresponding geometries. The first step is to identify changes between two geospatial datasets to identify their corresponding geospatial objects, i.e., corresponding geospatial object pairs in the source data set at t_2 and target data set at t_1. Two geospatial objects from different data sets are considered corresponding if they describe the same object in the real world at possibly different time points. Using this definition, an object at time point t_1, extended at time point t_2, is still considered a corresponding object. For this study, we neglect

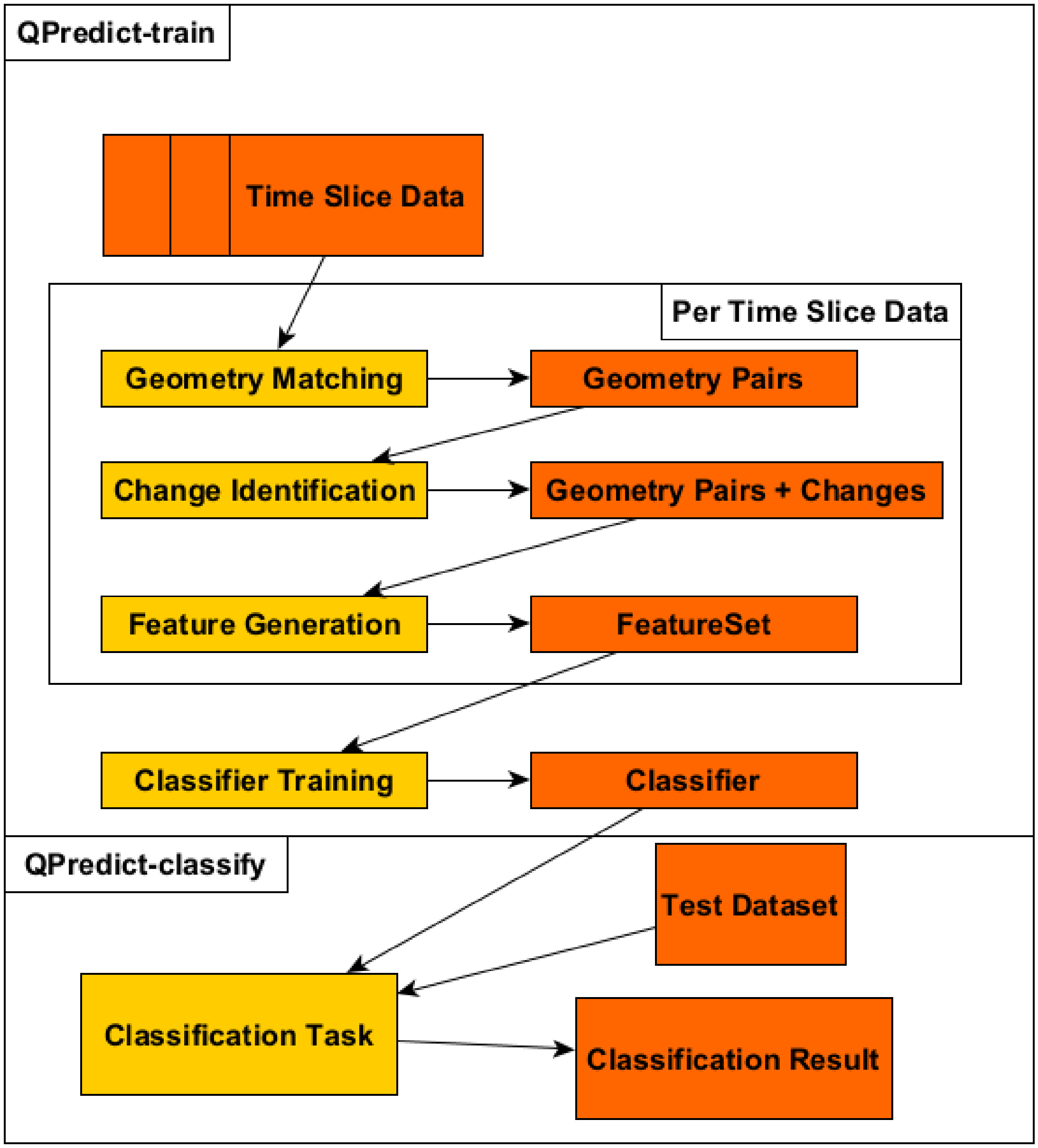


Fig. 5. A visual depiction of the steps of QPredict and its intermediate results which are discussed in this section.

the case in which a geometry from the target data set maps to many geometries of the source data set and vice versa. Instead, we focus on 1:1 matches only. The process to match two sets of geospatial objects $geom_1$ and $geom_2$, follows a matching process described in [49], which we adapt to our needs in Algorithm 1. In particular, we select a different

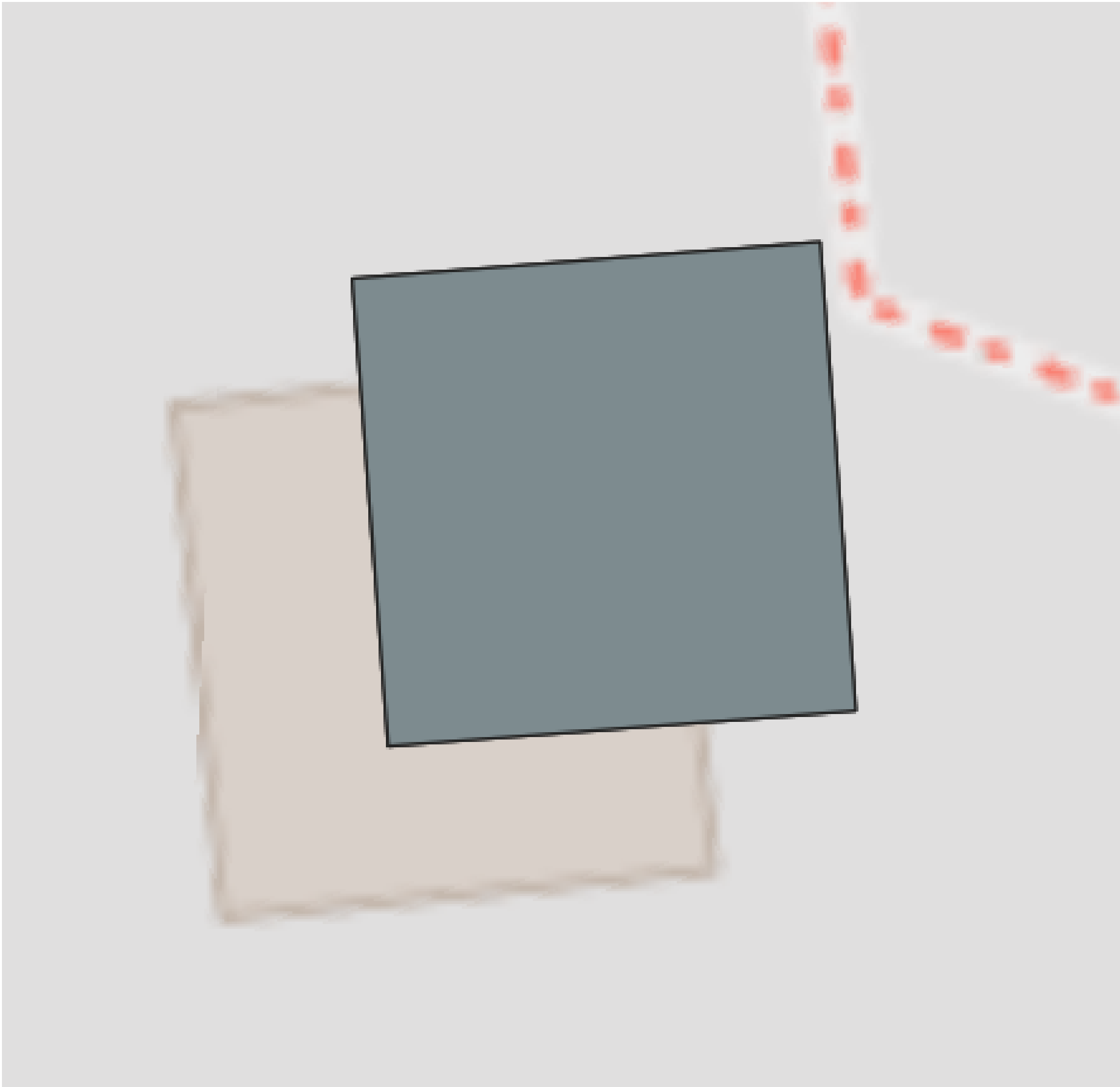


Fig. 6. Example of positional accuracy issues: Buildings in gray are included in the target data set, buildings in light brown in the source data set. The building shape looks the same, but their positional accuracy does not match. This fact can be measured using a distance metric (extrinsic) and possibly using an accuracy metric (intrinsic)

set of metrics used to calculate the similarity score between the given geometry pairs, as elaborated in the later course of the publication.

Algorithm 1 is comprised of the following functions: The first function *calcCorrespondingGeometryScore* takes two geospatial objects *first* and *second* and calculates a matching score between them. The matching score is calculated by deriving the distance of the two centroids of the geospatial objects (line 5). The closer the geospatial objects are to each other, the more likely they are to correspond and the higher their matching score. Next, the geometric similarity metrics

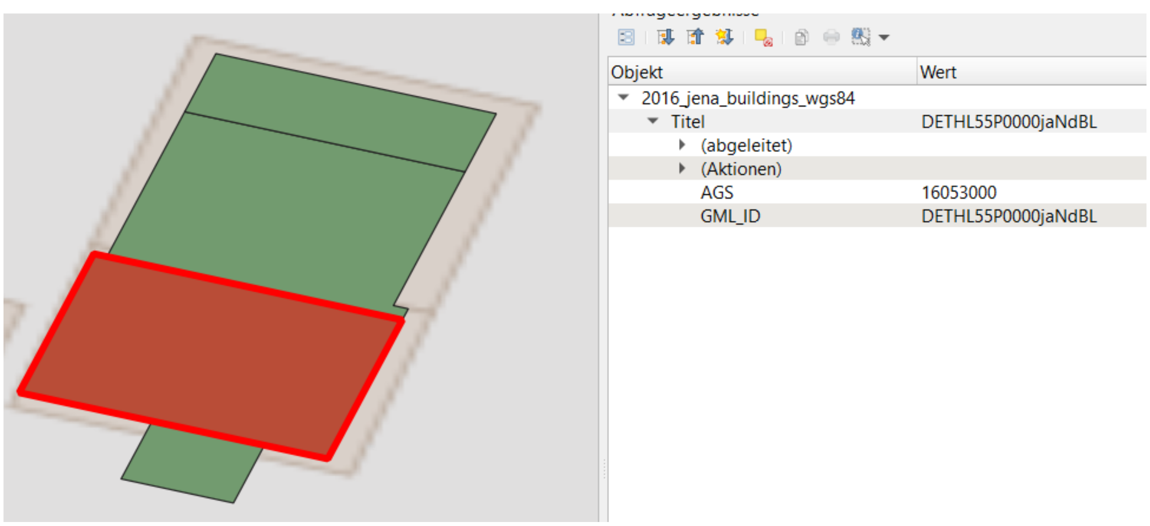


Fig. 7. Attribute changes: Changes in the target data set (example of the GML_ID as a unique identifier of the national mapping agency), which can be tracked by looking at the next revision of the geometry in Figure 8.

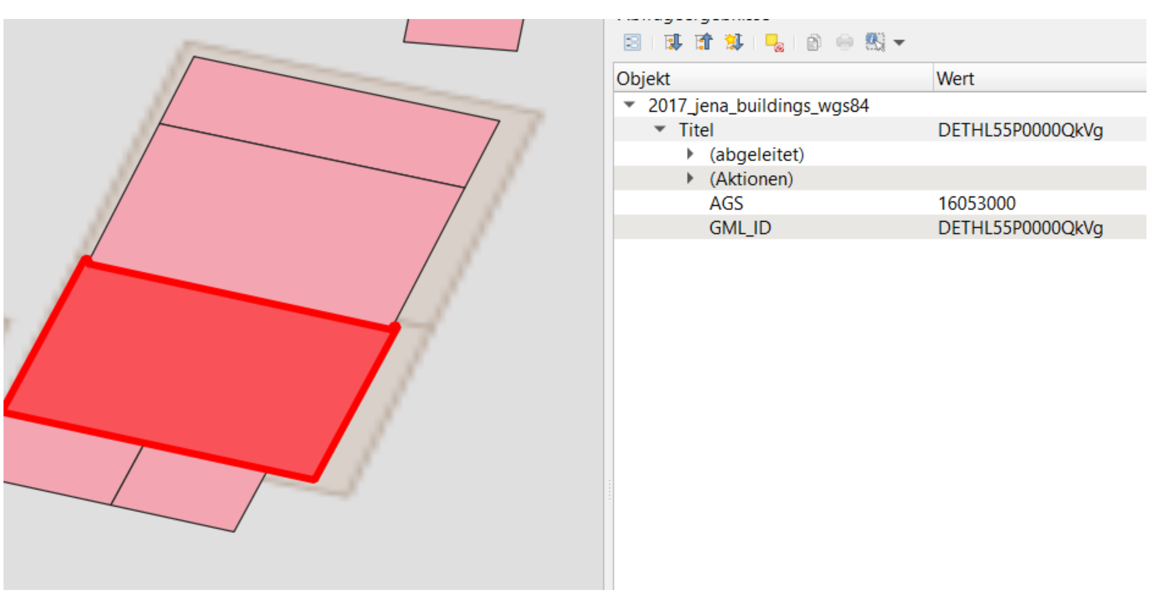


Fig. 8. Attribute changes: Changes in the target data set (example of the GML_ID as a unique identifier of the national mapping agency), which can be tracked by looking at the last revision of the geometry in Figure 7

of HausdorffSimilarity, [32], Frechet similarity, [1], and the intersection percentage of the geospatial objects metrics are applied in line 7 to determine if the geometry's shapes are similar. The matching score decreases considerably if the metric results indicate a similarity below a threshold of 75% of a given geometry metric, indicating that the geometrical shapes are dissimilar. This threshold has been tested to be suitable for our data and may need to be adjusted for map

Algorithm 1 Match corresponding Geometries by Similarity Score: Algorithm in pseudocode

```
1: global List<Metric> geometryMetrics, attributeMetrics
2: global Double distanceThreshold
3: procedure calcEquivGeometryScore(GeoObj o1, GeoObj o2)
4:     Double sim=0;
5:     sim=updateScore(sim,o1,o2, distance(o1,o2))
6:     for geomsim in geometryMetrics do
7:         sim=updateScore(sim,o1,o2, geomsim(o1,o2))
8:     end for
9:     for attsim in attributeMetrics do
10:        sim=updateScore(sim,o1,o2, attsim(o1,o2))
11:    end for
12:    return sim
13: end procedure
14: procedure createEquivGeometryMap(Set<GeoObj> geom_1, Set<GeoObj> geom_2)
15:    Map<GeoObj,GeoObj> geom_matchset=new Map()
16:    for g1 in geom_1 do
17:        Double maxScore=0
18:        currentBestMatch=null
19:        for g2 in geom_2 do
20:            if distance(g1,g2)<distanceThreshold then
21:                Double score=matchCorrespondingGeometryScore(g1,g2)
22:                if score>maxScore then
23:                    maxScore=score
24:                    currentBestMatch=g2
25:                end if
26:            end if
27:        end for
28:        geom_matchset.put(g1,currentBestMatch)
29:    end for
30:    return geom_matchset
31: end procedure
32: procedure updateScore(Double simScore,GeoObj o1,GeoObj o2,Metric m)
33:    if m instanceof GeometryMetric then
34:        simScore+=1-m.calculate(o1,o2)
35:    else
36:        simScore+=m.calculate(o1,o2)
37:    end if
38:    return simScore
39: end procedure
```

data in other areas. The reasoning behind this is that very often, a geometric comparison will have to be the most influential factor in a similarity comparison, as further comparisons between attributes of the given features do not necessarily have to correspond or even be available. Therefore, the positioning and shape of the geometry are decisive factors in the matching score calculation.

Finally, attribute similarity metrics, i.e., String Distance Metrics and number comparisons, applied on annotations and metadata of the geospatial objects contribute to the similarity score (line 10), which is returned (line 12). For each matching attribute, one point is awarded. The idea behind this matching is that even if similarly shaped geometries are present nearby in one place in time, they should be distinguished by sufficiently distinct attributes.

The second function *createEquivGeometryMap* receives the two sets of GeospatialObjects $geom_1$ and $geom_2$ as input. For each element in $geom_1$, it calls *matchCorrespondingGeometryScore* with all GeospatialObjects in $geom_2$. The matching with the highest similarity score is saved in the result of the method. We receive a map of matching geospatial objects from one geospatial object in $geom_1$ to the best matching candidate of $geom_2$ or no candidate if no matching could be determined. We call this result $geospatialobjectpairs$. This resulting map of $geospatialobjectpairs$ is the basis for calculating changes between its elements.

3.2.2 Change Identification II: Determining changes. For each geospatial object pair, changes between its two geospatial objects can be determined by comparing the geometry and attributes of the geospatial objects within the pair using Algorithm 2.

Algorithm 2 Change Detection of Geometry Pairs: Algorithm in pseudocode

```
1: procedure changeDetection(Map<GeoObj,GeoObj> geometrypairs)
2:     Map<Tuple<GeoObj,GeoObj>,String> changeResult;
3:     String geometryStatus="NoChange";
4:     for pair in geometrypairs do
5:         if pair.value==null then
6:             changeResult.put(pair,"Delete")
7:         else if pair.key==null then
8:             changeResult.put(pair,"Add")
9:         else
10:             geometryStatus=checkGeometryDifference(pair)
11:             if geometryStatus=="NoChange" then
12:                 geometryStatus=checkAttributeDifference(pair)
13:             end ifchangeResult.put(pair,geometryStatus)
14:         end if
15:     end for
16:     return changeResult
17: end procedure
18: procedure checkGeometryDifference(Tuple<GeoObj,GeoObj> pair)
19:     Geometry o1geom=pair.key.getGeometry(), o2geom=pair.value.getGeometry()
20:     if o1geom.getPoints().size()!=o2.geom.getPoints().size() then
21:         return "Change"
22:     end if
23:     for int i=0;i<o1.getPoints();i++ do
24:         if !o1.getPoints().get(i).equals(o2.getPoints().get(i)) then
25:             return "Change"
26:         end if
27:     end for
28:     return "NoChange"
29: end procedure
30: procedure checkAttributeDifference(Tuple<GeoObj,GeoObj> pair)
31:     Map<String,String> o1props=pair.key.getProperties(), o2props=pair.value.getProperties()
32:     if o1props.size()!=o2props.size() then
33:         return "Change"
34:     end if
35:     for int i=0;i<o1.getPoints();i++ do
36:         if !o1props.get(i).equals(o2props.get(i)) then
37:             return "Change"
38:         end if
39:     end for
40:     return "NoChange"
41: end procedure
```

As Figures 6 to 8 show, different kinds of changes of the spatial object are possible and occur in the data we work with.

We define a change of a geospatial object as any change of a geometry (addition of coordinates, change of coordinates, deletion of coordinates), any change in the given geometry attributes (attribute addition, deletion, modification), and any newly created and/or deleted geometries since the last known revision of the data set. Geospatial objects without matches represent objects that have possibly been added or removed.

The result of Algorithm 2 is a determination of the nature of the change that has occurred in a geometry pair, i.e., maps geospatial object pairs to a classification that reflects the geospatial object pair's behavior in a given time slice. The algorithm achieves this by first determining a possible change in the geometries of the geospatial objects (function checkGeometryDiffrence, line 18). If the geometries can be considered equivalent, a change of attributes associated with the geospatial objects is conducted (function checkAttributeDifference, line 30). Finally, a map of $geospatialobjectpair$ to change classification is returned by the algorithm.

It should be noted that Algorithm 1 and subsequently Algorithm 2 can and are applied on different geospatial object pairs, e.g. on target dataset of time point t_1 and target dataset of time point t_2, source dataset of time point t_1 and source dataset of time point t_2 and source dataset of time point t_1 and target dataset of time point t_1, depending on the nature of features to be generated for the machine learning classification.

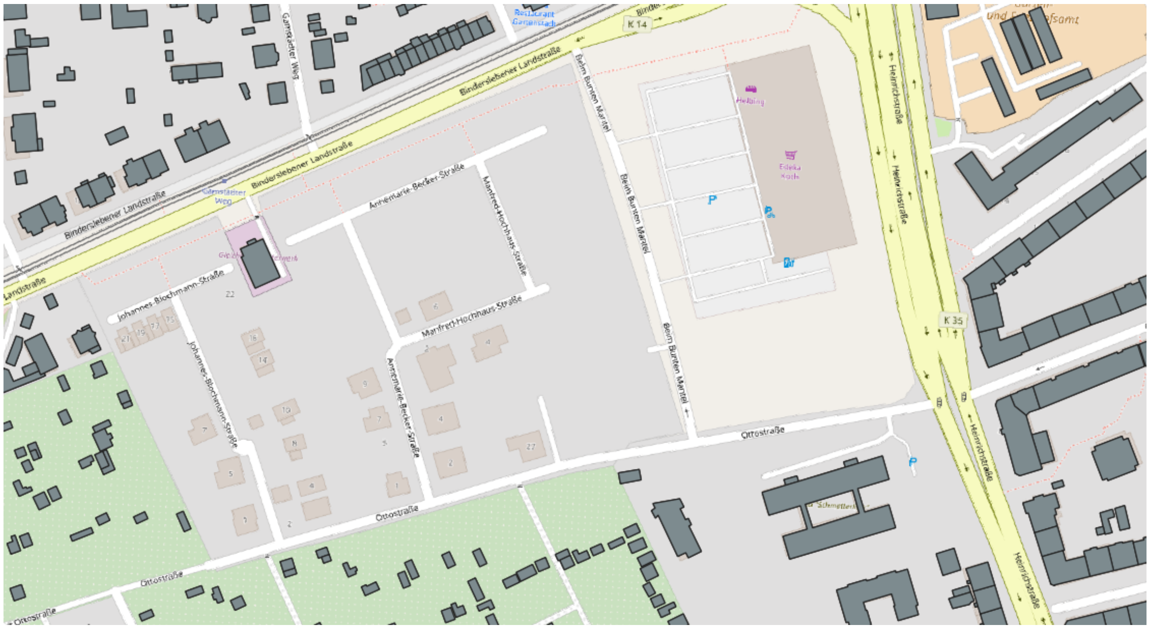


Fig. 9. Example of geometries to be added: A new residential district in Erfurt "Bunter Mantel" has been built. The area is already covered in the source data set of VGI data (light brown) in the center of the graphic. These geometries need to be added to the MGI target dataset in the next revision (e.g. constitute valid and relevant changes for the mapping agency) https://www.erfurt.de/ef/de/leben/planen/stadtplanung/fp_ bp/brv/109398.html

3.3 QPredict-classify

We previously described how to find corresponding geometries and detect changes between them. In this section, we describe how the classification is performed, assuming a classifier that has been trained on a given training set.

To describe *QPredict-classify*, we interpret the detected changes as suggestions for updating the target dataset, so-called transfers. For each change that has been identified to have happened in the source data set from t_1 to t_2, *QPredict-classify* suggests how to modify the target data set from t_1 to best approximate the real world at t_2. For this purpose, *QPredict-classify* classifies the change in the source data set to fall into one of three categories:

(1) *Literal transfer*: Suggest adding (or deleting) an object to the target data set exactly as added or deleted in the source data set. (ML Classes: Add, Delete, e.g. Figure 9)
(2) *Modified transfer*: An object has been changed by adding, moving, or deleting points and/or attributes in the target data set as suggested in the source data set ("Change"). Figures 10 and 11 show examples of the source dataset, and/or its metadata and attributes, changing from timepoint t_1 to timepoint t_2. (ML Class: Change)
(3) *No transfer*: In this case, the change in the source data set under consideration is suggested not to be considered for the target data set. (ML Class: NoChange)

Figure 11 shows examples that, within 4 years, significant additions (visualized in green), deletions (visualized in red), and changes (also shown in green) are common in the areas that we investigate. QPredict should, in the end, be able to classify these according to the classes introduced previously.

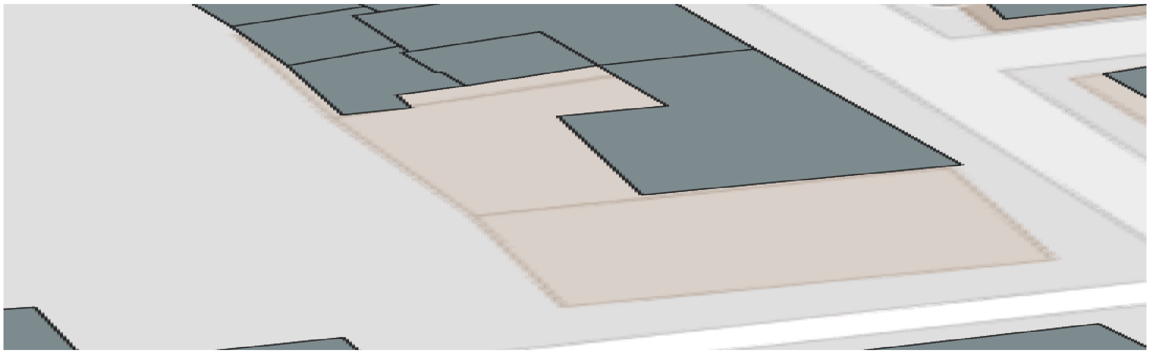


Fig. 10. Example of geometry extension: Buildings in gray are included in the target data set, buildings in light brown in the source data set. Here, the grey geometries represent houses in the MGI target dataset. The source dataset already shows that the houses have been extended significantly, thus indicating a change in the size of an existing geometry.

Fig. 11. Geographical objects in the target data set at time points 2015, 2016, 2017, and 2018: Dark yellow objects existed in 2015, 2016, 2017, and 2018. Light/dark green objects first appeared in 2016/2018, respectively. Light/medium/dark red objects no longer existed as of 2016/2017/2018, respectively.
The image shows that building changes are widespread in the area that is being investigated.

The inference that *QPredict-classify* pursues to derive such a classification (c_1-c_n) is built on the following configuration of feature vectors with *Add*, *Delete*, *Change* and *NoChange* the classification targets

$c_1 = (Add, qm_1, qm_2....qm_n)$
$c_2 = (Delete, qm_1, qm_2....qm_n)$
$c_3 = (Change, qm_1, qm_2....qm_n)$
$c_4 = (NoChange, qm_1, qm_2....qm_n)$

and $qm_0 - qm_n$ being the results of data quality metric calculations.

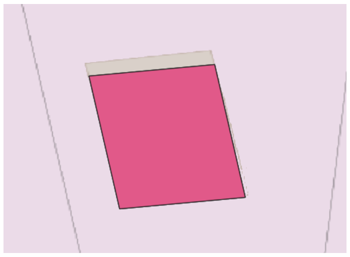
3.4 QPredict-train

In this section, we introduce *QPredict-train*, which constructs training and test set data and creates the classifier.

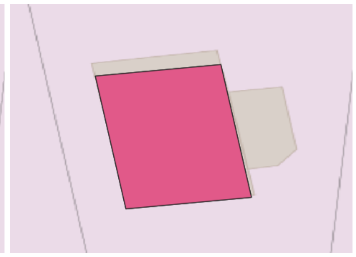
3.4.1 QPredict-train input. The input of *QPredict-train* is depicted in Figure 4. *QPredict-train* receives a target data set at t_1, a source data set at t_1 and a source data set at t_2. To allow a machine learning task to train, a classification target needs to be given as well. Therefore, the input of the target data set at time point t_2 is also given in place of the question mark shown in Figure 4. We call this input the input at time slice $x - y$. For example, QPredict could be trained on a target data set released on the 1st January 2015 (t_1) and use a target data set released on the 1st January 2016 (t_2) for verification. Geospatial objects nearest to the respective time points t_1 and t_2 would be matched from the source data sets. In Figure 4, the nearest source data set geometry to t_1 is last modified on the 20th December 2014. *QPredict-train* may receive an arbitrary number of time slices to train on as input, e.g., 2015-2016, 2016-2017, 2017-2018, to maximize the number of training examples in the area to be investigated.

3.4.2 QPredict-train algorithm. Given an arbitrary input of time slices, the *QPredict-train* algorithm operates as follows:

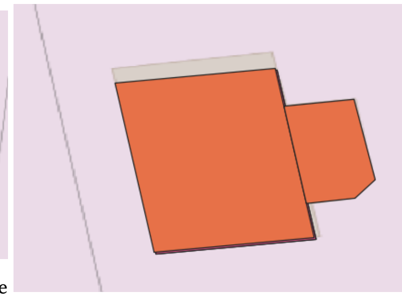
(1) *Target data set Geometry Matching using Algorithm 1:* Algorithm 1 matches the current target geospatial object in t_1 with the geospatial object in the target dataset at t_2 representing the same real-world object (corresponding) using positional matching or a given geometry history. Given a history of geometry changes, the algorithm may use the history to find the corresponding geometries of earlier time slices.
(2) *Source data set Geometry Matching:* Application of Algorithm 1 on the source data set to retrieve a set of source geospatial data objects on two time points. The goal of this matching is for each source geometry in the area of the target data set to find the revision of the source data geometry whose creation date is closest to time point t_2.
(3) *Source to Target Geometry Matching:* Calling Algorithm 1 with a set of source dataset at t_1 and a target data set at t_2. This step matches geometry pairs between target data set geometries and source data set at or closest to t_1.
(4) *Data Quality Metric Calculation:* Data Quality metrics are calculated according to the metric type on the source and/or target data sets, respectively. Examples of where metrics could be applied are shown in Figure 13.
(5) *Classification Generation:* Classifications are extracted by comparing the target data set at t_1 with the target data set at t_2 and added to the machine learning feature vector using Algorithm 2
(6) *Machine Learning Feature Vector Preparation:* Data quality metric results become the basis for the machine learning feature vector of the machine learning classification
$qm_{result1}....qm_{resultn}, class$



(a) 1st January 2015: A house (pink=target dataset, grey=source dataset) with a positional inaccuracy: The source dataset geometry is slightly bigger than the target dataset geometry.



(b) 31st August 2015: A house (pink=target dataset) has been extended by a garage in the source dataset (grey). Positional inaccuracy: The area of the buildings does not match.



(c) The house (orange in the target dataset) at time point t_2 (1st January 2016). A garage in the target dataset has extended the house. *QPredict-train* can learn from this result to improve the classification accuracy

Fig. 12. Real-world application example: A house in the German city of Erfurt represented in OpenStreetMap (grey) and official governmental data (ALKIS) changes over time. OpenStreetMap, in this case, includes a change at an earlier time before it is reflected in governmental data.

(7) *Classifier Training:* The so-created training data set of machine learning feature vectors is used to train a classifier using a chosen machine learning algorithm

3.4.3 *QPredict-train output.*

QPredict-train outputs a classifier trained on several training sets representing time slices (t_1 - t_2, t_2 - t_3....). The classifier is used by *QPredict-classify*, and the training data sets can provide a basis for the performance analysis of the algorithm in a testing environment.

3.5 Running example

We illustrate QPredict using a real-world application scenario from using our dataset representing building footprint data from 2015. One building in Figure 12a is represented in both the target and the source dataset at time point t_1 (1st January 2015) and the nearest appropriate match of the source dataset (25th November 2014). The buildings mostly overlap, with the source dataset geometry's area being slightly bigger than the area of the target dataset geometry - a typical inconsistency. Apparently, Figure 12b shows that the building has been extended by supposedly a garage in the source dataset, which results in a change in the area of the building in the target dataset one year later Figure 12c. This area change is reflected in various data quality metric calculation results, which are being executed during the creation of machine learning feature sets for QPredict. We give some examples of data quality metrics that are subject to change by this change of geometry:

- *Extrinsic Metrics:* e.g. Shape Similarity metrics between target t_1 and source t_2 (e.g. HausdorffDistance)
- *Intrinsic Metrics:* e.g. Geometry Area (source dataset at t_2)
- *Metadata Metrics:* e.g., Change Rate from History, Freshness, Average User Experience (source dataset)

These changes are reflected in the machine learning feature vector created for each of the corresponding geometry pairs considered by QPredict.

When executing *QPredict-train*, we already know if the target dataset geometry is subject to change. The target dataset at time point t_2 has already been given to us for training purposes. In this running example, the house has indeed changed, which is reflected in Figure 12c. When executing *QPredict-classify* on a given test set, the algorithm will provide a classification to suggest the geometry's addition, deletion, or modification or if the geometry should remain unchanged. In this way, we can evaluate the algorithm's efficiency later in the paper.

4 MACHINE LEARNING FEATURES COMPOSITION

In Section 3.3, we have presented how QPredict operates in principle. This section describes the categories of machine learning features that may be used with QPredict and our reasoning for choosing the particular feature sets we use for the QPredict classifications discussed in Section 5.

4.1 Deriving machine learning features from data quality dimensions

As established previously, our machine learning features are derived from extrinsically and intrinsically calculated data quality metrics (cf. Sections 2.2 and 2.3).

Extrinsic features allow us to track geometrical changes between the two datasets, possibly indicating a change. If data quality metrics indicate minimal changes, the machine learning algorithm should lean towards classifying a non-change. If data quality metrics indicate more significant changes, it might lead to a classification towards a geometry change.

Intrinsic features are used to detect weaknesses when representing a single geometry. We assume that geometries that have been, e.g., vandalized in OpenStreetMap should not be candidates for which changes need to be applied. Therefore, intrinsic metrics check the consistency of a geometry in the source data set. Consistent geometries should be more likely candidates for a transfer of changes.

However, we first want to inform the reader of the variety of intrinsic and extrinsic data quality metrics from data quality dimensions that can be considered machine learning features. As choosing suitable machine learning features out of the given data quality dimensions is no trivial matter, we want to explain why we considered these categories of machine learning features beneficial for our classification task. We consider the following categories of derived machine-learning features:

- *Derived features vs. Literal features:* Features either derived from a geospatial object or included in a geospatial object
- *Dynamic features vs. Static features:* Features either only dependent on the current time slice or dependent on many time slices (Geometry Validity vs. Geometry Freshness)
- *Intrinsic features vs. Extrinsic features:* Features measured with or without a target data set
- *data set dependent features vs. Non-data set dependent features:* Features dependent on statistics generated on the given data set (Average Amount of Edits vs. Geometry Simplicity)
- *Imported vs. non-imported features:* Features that have been imported from the source data set and cannot be measured in the target data set vs. features that can also be measured in the source data set (OSM has a detailed history including user behavior and edits, official data lacks this information)
- *Differential features vs. Individual features:* Features measured on the difference of two attributes of compared geographical features (differential) vs. without a comparison (Comparison of building names vs. Geometry Validity)
- *Vicinity dependent features vs. Non-vicinity dependent features:* Features dependent on geographical features in the vicinity of the current geographical feature vs. self-focused features (Neighborhood Freshness vs. Amount Of Attributes)

We found it important to cover any of the aforementioned data quality dimensions in our feature sets, as they may point to a possible change in the geospatial object. Table 1 shows machine learning features matching the categories we use and their source, either from the literature or existing tools.

Machine Learning Feature Type	Title
Differential	Positional Accuracy [18]
Differential Extrinsic	Hausdorff similarity [32]
Intrinsic Vicinity	Neighborhood Freshness
Data set-dependent	Average History Size [48]
Target-derived Intrinsic Individual	Geometry Validity
Intrinsic Imported	Last user to Edit
Intrinsic Imported-derived	User Quality Score (OSM)
Target-derived Intrinsic Dynamic	Edit Frequency
Target-derived	Freshness [48]

Table 1. Examples of the different machine learning feature categories described before. The complete list of machine learning features that were used is given in Appendices B.1 to B.3.

4.2 Data Quality Metric calculation

Figure 13 shows which data is involved in calculating either data quality metric result. Extrinsic metrics are calculated between metadata, geometries, attributes, and a possible geometry vicinity. For metadata metrics, we assume that user experience scores in OpenStreetMap[3] and the edit frequency of the geometry should give us a good enough indication to judge if an edit is likely to be trustworthy, thus increasing the likelihood of a transfer.

Vicinity metrics may point to similar edits in the neighborhood, which do not necessarily indicate a well-intended geometry change but may emphasize assessing other data quality metrics. Similar assumptions are valid for attribute changes.

Intrinsic metrics are calculated on the source data set at time point t_1 and time point t_2 and on the target data set at time point t_1. Therefore, intrinsic machine learning features can stem from three possible sources: The target geospatial object at t_1, the source geospatial object at t_1, and the source geospatial object at t_2. We include machine learning features originating from all three data sources.

4.3 Feature Set Composition

This article aims to investigate the performance of a set of each intrinsic, extrinsic, and combined features (cf. section 4.1) to set a baseline for further refinement of these machine learning feature sets for better classifications. The machine learning features have been selected in such a way that they

(1) Are represented and used in related work
(2) Cover the dimensions of machine learning features described above
(3) Are expected to have a sufficient impact on the result

To that end, we conducted a feature correlation analysis by calculating the GainRatio of each feature candidate. The features with a GainRatio of at least 1%, were selected for inclusion in the feature sets. The detailed composition of the feature sets can be seen in Appendices B.1 to B.3.

[3]https://hdyc.neis-one.org

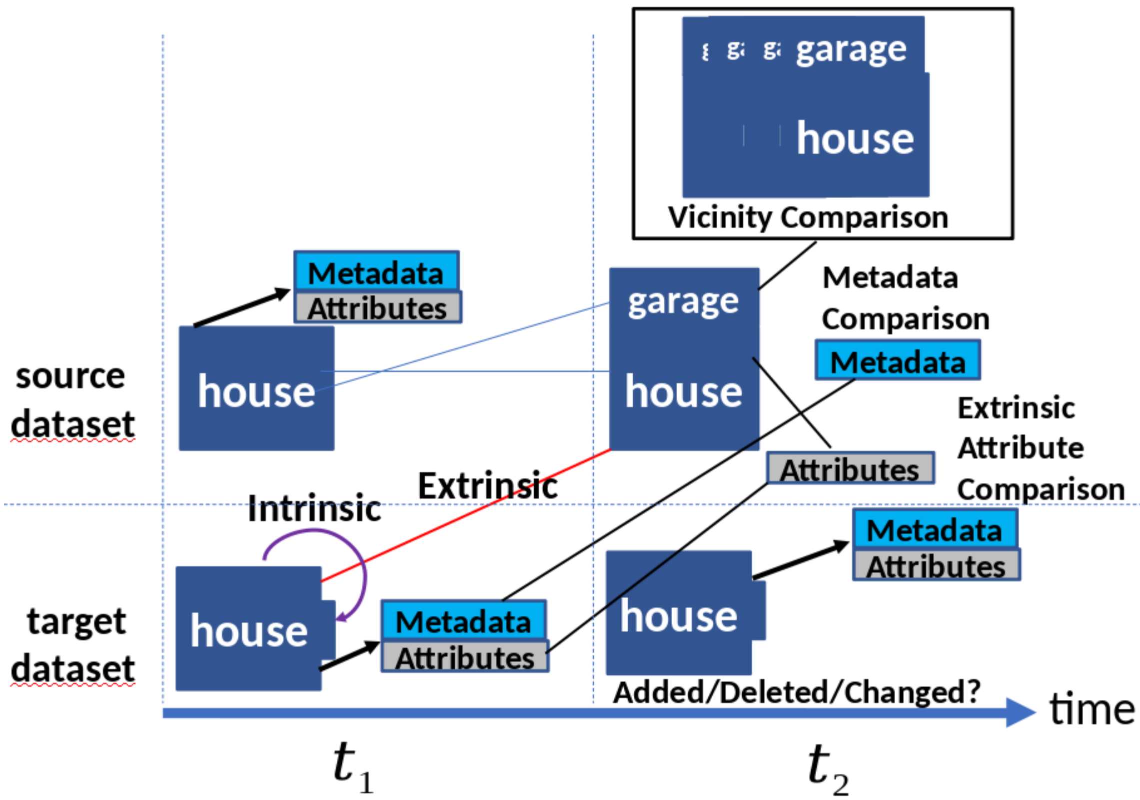


Fig. 13. Data Quality Metrics: Input for Machine Learning: Visualizes between which geospatial objects data quality metrics can be created in the timeframes given

5 EXPERIMENTAL SETUP

The experimental setup was designed to evaluate the performance of a *QPredict-train* classifier for predicting additions, deletions, and change predictions, to test the false positive rate of the classifier (cf. e.g., Figure 16) and to which extent intrinsic and extrinsic data quality metrics contribute to correct classification. To that end, we firstly introduce the data sets that were used for the experiment in Section 5.1, explain the machine learning setup in Section 5.2, show how we evaluate the machine learning results in Section 5.3, before presenting and discussing the results.

5.1 Data sets

We now describe the administrative geospatial data set (target data set) and corresponding data sets from OSM (source data sets) and discuss their change rate to determine if they change sufficiently often to justify a classification. OpenStreetMap represents one of the richest sources of VGI data currently available. The administrative geospatial data is collected by a mapping agency and, therefore, classified as MGI data. We remark that VGI is in certain areas managed by official authorities[4] or collected by professional mappers yielding arguably a comparative or even more detailed

[4]e.g., Cologne, Germany https://www.stadt-koeln.de/basisdienste/stadtplan/osm/index.html

quality map than official data sources. Our study focuses on test areas for which we know that the local mapping agency does not contribute to OSM to avoid this problem.

Target Data Set (Administrative Data). Our target data consists of geospatial data in the ALKIS standard [52] provided by the mapping agency of Thuringia, Germany[5] (ALKIS NAS) data[6] combined with building footprints data (Hausumringe HU) data[7] for the area of Thuringia in yearly revisions from 2013 to 2018 respectively. The data set includes building footprints of Thuringia extended with address data, nationwide geometry IDs, geometry type, and length. As geometry classifications are only provided since 2016, we assume the building type before 2016 to be the same as in 2016. We selected two experimental areas: Jena and Erfurt, similar-sized cities in Thuringia. We assume that most map changes in urban areas as rural areas contain fewer geometries in general, provide more possibilities for geometry changes, and arguably have a more diverse and active VGI community.

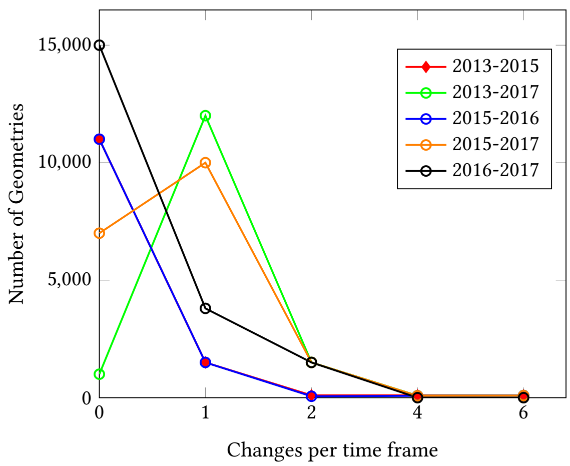
Source Data Set (OSM). The OSM data set consists of geometries we matched using the corresponding geometry matching Algorithm 1. The procedure is repeated for identified, added, and deleted geometries in the target data set to create a set of instances representing all classification targets we want to classify. The data set is enriched with metadata about the users and edit history extracted from the OpenStreetMap database. Therefore, we gain a set of changesets with the respective geometry at the time of the changeset. We take the geospatial object nearest to timepoint t_2 out of this set of changesets.

Preliminary analysis of map changes. We tracked map changes in target and source data sets as a preliminary analysis. We need to ensure that changes are sufficiently frequent to train a machine learning approach, i.e., that about half of the geometries have been edited at least once during our analysis time frame. For the target data set, the Feature Manipulation Engine [60] UpdateDetector Plugin[8] was used to track geometry changes, additions, and deletions, respectively. OSM changes were tracked using the OSM history of each geometry, using changesets[9] present in OpenStreetMap, which reflects when a geometry has been changed by whom and in which way. Figure 14a shows that for the area of Jena, a significant proportion of the geometries have been edited at least once in the time frame of 2013 to 2017, for which we possess target data sets. Erfurt (Figure 14b) exhibits similar behavior with slightly more edits on average.
In both Jena and Erfurt, very few geometries have been edited more than 3 times, and about the same amount of geometries have not been edited within four years. Within one year, one to two edits are commonly performed per geometry for less than 50% of all geospatial objects. Considering the edit frequency observed, we determine that the tracked changes provide sufficient distinct changes to train our classifier for QPredict, as we can expect about half of the geometries to have been edited at least once in OpenStreetMap in the time frame we investigate.

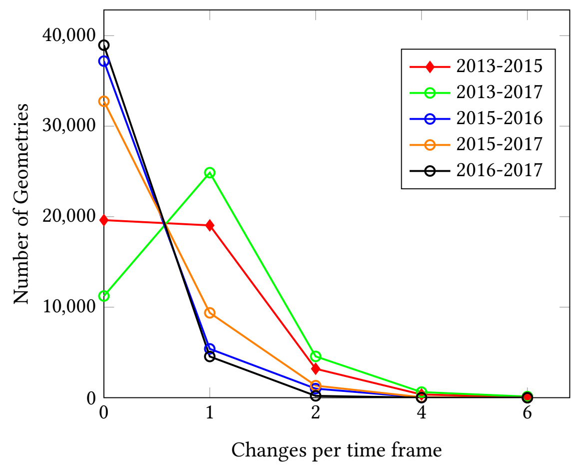
In the target data sets, we can observe the following behaviors:

In Jena, about 3000 to 5000 buildings were changed within one year, creating a building change rate of about 7-12%, respectively. On average, 2% of new geometries appeared in the target data sets annually. It becomes apparent that way more geometries have been edited in OSM than in the target data set, which is unsurprising considering everyone can edit the map at any time. A higher edit rate also gives the user data quality metrics a better chance to give a non-biased result. In Erfurt, we can observe a change rate of 4-7% of the geospatial objects, less than in Jena. On average, 2% of

[5] https://www.thueringen.de/th9/tlvermgeo/
[6] https://tlbg.thueringen.de/online-shop-vertrieb/testdaten
[7] https://tlbg.thueringen.de/online-shop-vertrieb/testdaten
[8] https://hub.safe.com/transformers/updatedetector
[9] https://wiki.openstreetmap.org/wiki/Changeset



(a) OSM Geometry Editing Frequency in Jena



(b) OSM Geometry Editing Frequency in Erfurt

Fig. 14. OSM geometry editing frequencies in Jena/Erfurt

new geometries appeared in the target data sets annually. Again, the geometry change rate in OSM is higher than in the target data set.

Lastly, we investigated the change rate of the federal state of Thuringia in which the two cities are located. With a growth rate of 0.6% and a change rate of 0.1% for the whole state of Thuringia, we are confident to have picked areas where a representative amount of changes should have occurred over the time points we tested.

Expectations. Besides real-world changes being adapted in map data, we expect error corrections in target data sets, updates due to new regulations by authorities and laws (some data should or should not be collected or differently labeled), or mistakes being introduced in official data. OSM changes are typically more diverse, not always relevant for a mapping agency, and often unrelated to the official data changes. Not every machine learning feature type is tracked in the official data set. Especially malicious attempts of data set modifications in the source dataset (i.e., vandalism [45]), if any, should stand out and are expected to be recognized by the algorithm. These natures of map changes represent possible reasons for the results we attempt to achieve in our classification.

5.2 Machine Learning Setup

Using the classification specifications defined in Section 3, we introduce the machine learning feature sets and machine learning configuration. Given the different categories of machine learning features introduced in Section 4.1, we define three machine learning feature sets. The *Baseline* feature set consists only of intrinsic data quality metrics. The *Extrinsic* feature set consists of only extrinsic metrics, and the *Combined* feature set consists of all features of *Baseline*, *Extrinsic*, and metadata metrics. Appendix B describes the complete list of features. To compare the performance of the different classifiers for our task, we chose the following machine learning algorithms: RandomForest Classification [56] and IBk Classification [20], thereby covering two different areas of machine learning, instance-based learning, and decision trees as a comparison. We used the Weka Machine Learning Toolkit [29] version 3.8 to pursue the classifications and chose the default settings for the classification approaches. The hyperparameters used in the classifications are shown in Table 2 While we tried different configurations of machine learning parameters, we did not aim to optimize them,

RandomForest	Hyperparameter	Value
	BagSize	100%
	Number of trees	100
	mTry	log(M+1) (M=inputs)
	maxDepth	unlimited
IBk	Hyperparameter	Value
	KNN Parameter	1
	Distance weighting	No
	Distance measure	LinearNNSearch

Table 2. Hyperparameters for the given machine learning algorithms

as the focus of this research was to find out whether a classification in this way is feasible at all based on our given test data. The parameters described in Table 2 are the parameters we used after the initial test. In this particular case, changing the parameters further did not significantly improve classification performance. We see the optimization and the testing of further machine learning algorithms as future work, as this would require testing the approach on geospatial data of different origins and configurations.

5.3 Evaluation

To evaluate QPredict's success, we use *QPredict-train* to train a classifier on the time slice given by the two available time points t_1 and t_2 as shown in Figure 15 - and possibly on further time slices depending on their availability. As described in Section 3.4, more training data can improve the accuracy of the classifier. Using the classifier, we then predict a geospatial object change at $t_3 - \epsilon$, the current time when the classification is conducted (assumed to be before t_3 to be of use). The classification is verified using a target data set at t_3, which is available for verification purposes. In this experiment, we trained on two time slices, 2015-2016 and 2016-2017, and evaluated on the time slice 2017-2018. Referring to our running example, we would evaluate the prediction of our classifier, which we trained using several iterations of training data from previous time slices, e.g., time slice t_1-t_2 (e.g., 2016-2017). We choose another time slice available to us, e.g. (t_3), to evaluate the success of our classifier, e.g., 2017-2018. Using this approach, we evaluate our classifier on a *future revision* of the target data set as a test set on which the classifier has not been trained. We evaluate the accuracy of the classifier using precision, recall, and f-score.

5.4 Results

The machine learning classification results are highlighted in Figures 16 to 18 and tables 3 and 4 for the three feature sets for Jena and Erfurt, respectively. The tables included the percentage ranges for all three classifications (Intrinsic, extrinsic, and combined). Detailed results for all classifications are included in the annex of this publication. Our results include the precision, recall, and f-score of the successful classifications of geospatial object changes, additions, deletions, and non-changes, as we consider this to be the classification challenge. We provide the average score of the four classifications in the "Overall" classification.

6 ANALYSIS AND DISCUSSION

We first judge the classifier's performance to understand its advantages and disadvantages. Then, we discuss the classifier's most influential features, limitations, and applications in a use case.

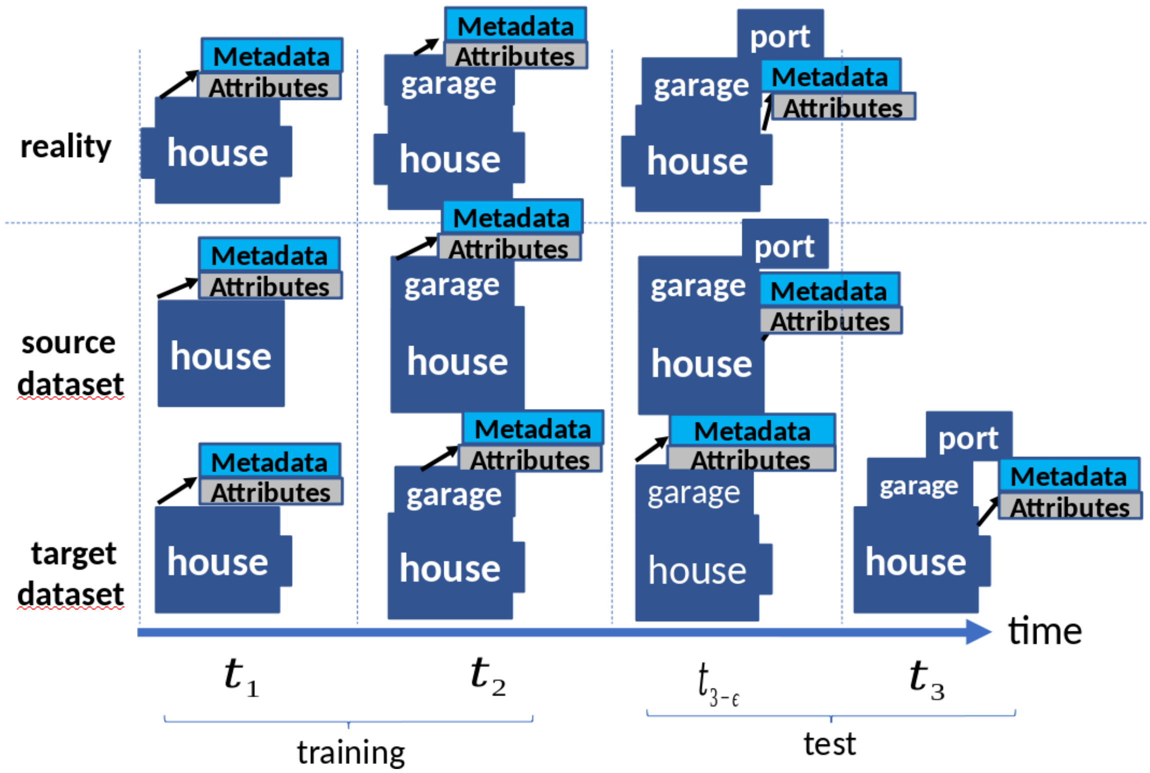


Fig. 15. Data Quality Prediction Model: The classifier trained between t_1 and t_2 predicts a geospatial object change at $t_3 - \epsilon$. The quality of this prediction is evaluated using the geospatial object available at t_3 to measure the accuracy of the classifier. Here, ϵ represents the delta time between the object change appearing in authoritative data at time point t_3 and the time the object changed in reality.

Area	Overall	NoChange	Change	Added	Deleted
Jena	83%-95%	87%-96%	59%-72%	40%-80%	49%-96%
Erfurt	90%-96%	94%-97%	53%-95%	40%-78%	4-80%

Table 3. Precision ranges over all three test sets. Best precision values were achieved using the *Combined* feature set

Area	Overall	NoChange	Change	Added	Deleted
Jena	92%-96%	87%-96%	30%-58%	16%-57%	13%-81%
Erfurt	92%-96%	91%-99%	31%-54%	16%-21%	4-89%

Table 4. Recall ranges over all three test sets. The highest recall values were achieved using the *Combined* feature set

6.1 Classifier Performance and Discussion

One might expect that a machine learning classification using only intrinsic features (*Baseline* feature set) would not produce a good enough classification, as intrinsic data quality metrics are insufficient to create absolute statements

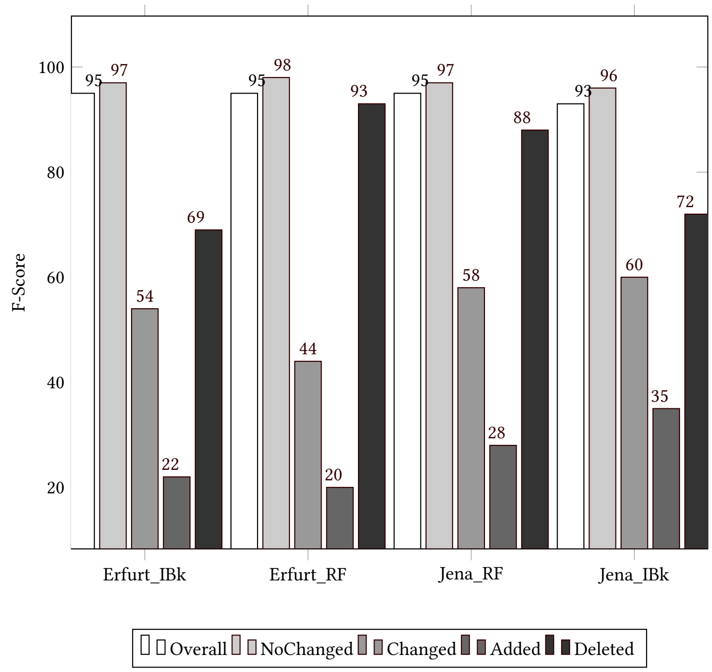


Fig. 16. *Combined* Featureset Results (F-Scores) for Erfurt and Jena with IBk and RandomForest (RF) classifications

about data quality. On the other hand, intrinsic features may point out changes over time that provide valid geometry results, which could be judged as a valid modification by the machine learning algorithm.

The *Combined* feature set performs best among all classifications (which was also expected), while the *Extrinsic* and *Baseline* feature sets usually performed 10%-20% worse. Therefore, our initial assumption was only partly true as the intrinsic feature set performed comparably well versus an extrinsic feature set in detecting map changes. The fact that the *Combined* feature set performed best overall shows the relevance of all introduced metric types for the classification, confirming our assumption that metadata and especially user experience classifications have an impact.

Non-changing geospatial objects could be identified with precision, recall, and f-score greater than 90%. Changes were notably harder to predict (f-score <=60%). This may be due to the lower availability of training examples for geospatial object changes.

However, despite achieving a lower f-score for change classifications, the precision of the classifier for the class ”Change” is comparably high (up to 95% in Erfurt) (cf. Tables 3 and 4). This observation is also true for the classes ”Added” and ”Deleted”. High precision and low recall indicate that although not all changes can be identified correctly, the ones classified as such can be transferred as valid results to the target dataset.

In an intended application case, this can be a satisfying result when the classifier should signify map changes to a mapping agency. If instances are classified wrongly, they are in the majority classified as non-changing, as indicated by a low false-positive rate, as exemplified using the example in Figure 19. These are mistakes that a mapping agency is more likely to accept than a signified change that has no relevance to the target data set. Despite a low recall, the mapping agency could use the classifier to hint at uncertain areas in their map data to investigate map changes.

Differences in performance between algorithms

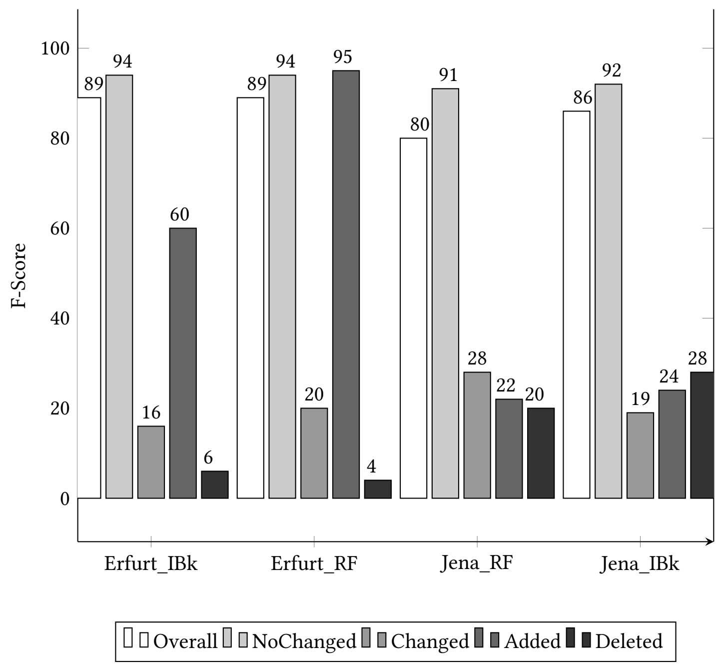


Fig. 17. *Extrinsic* Featureset Results (F-Scores) for Erfurt and Jena with IBk and RandomForest (RF) classifications

The algorithms used for the machine learning training IBk and RandomForest showed minor differences in outcome. The IBk classifier performed slightly better on the *Baseline* feature set, which can be observed in both test areas. Also, the IBk classifier yielded significantly better results for the class "Deleted". We observed that the IBk algorithm performs better when fewer training examples are given, while the RandomForest classification performs better when more training data is used.

Between the areas of Erfurt and Jena, it can be observed that the classification in Jena yielded a slightly lower f-score overall but performed better on the changed classification. We attribute this to the fact that more changes, in general, took place in Erfurt, and therefore, its classifier produced more accurate results.

Differences between feature sets

We found that intrinsic features were quite influential for the classification of deletions and changes, as is evident when comparing the results in Figure 18 vs. Figure 16. Despite improving the classification results in the *Combined* feature set, extrinsic features were less influential in contributing to the result. This might be due to differences between the source and target data set, shown in Figure 1. Suppose geospatial objects in the source data set are modeled less precisely on a wider scale. In that case, extrinsic data quality metrics like HausdorffDistance or OverlappingDegree may generally give a higher distance. Extrinsic geometry changes may not be as influential for a correct classification.

Comparison to related work At this point, asking how QPredict compares against previously conducted work is natural. QPredict is, to the authors' knowledge, the first machine learning classification task that uses data quality metrics to classify changes, including not only geometry data but also attributes and metadata for change detection. In addition, QPredict is, according to the related work shown in Section 2, the second publication to attempt to classify

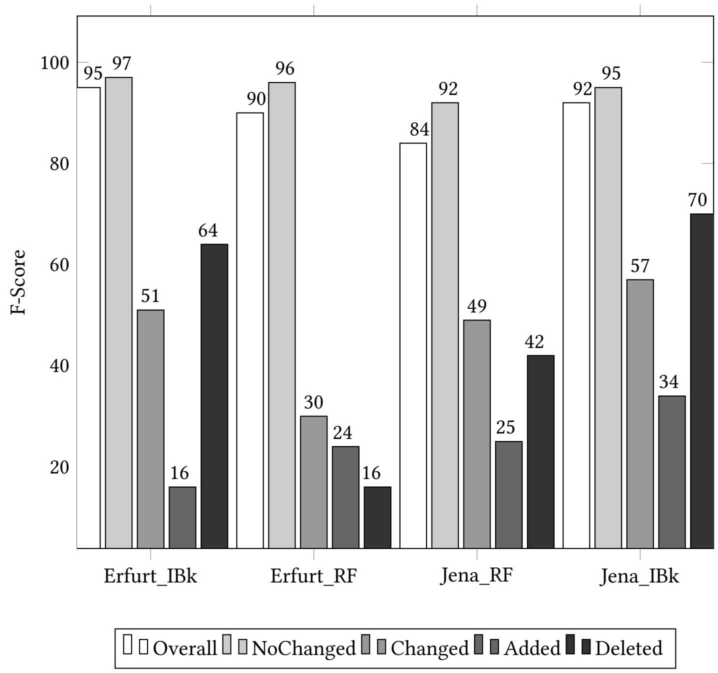


Fig. 18. Baseline Featureset Results (F-Scores) for Erfurt and Jena using the IBk and RandomForest (RF) classifications

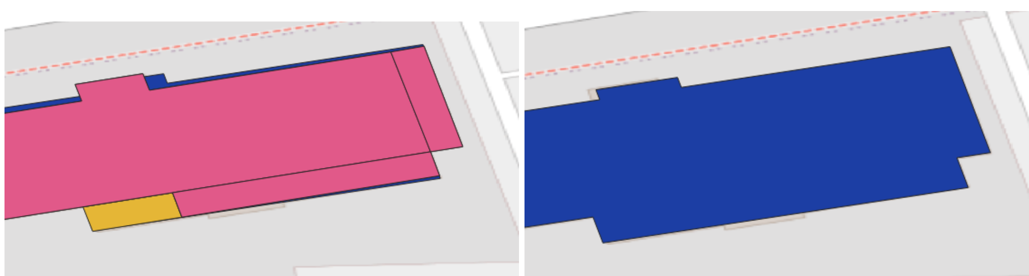


Fig. 19. Example for a change which QPredict has missed. The OpenStreetMap geometry (in blue) is already modeled in such a way that QPredict did not detect a geometry change from the geometry in the year 2017 (in pink) to the geometry in the year 2018 (in yellow)

whether updates from VGI data should be applied to MGI data. Thus, we can only set the results of our work into perspective by comparing it to the non-machine learning approach by [63], which attempted to classify only changes and not additions and deletions at the same time. When comparing the results of QPredict to this related work, we can see that the precision achieved by QPredict is 10% higher than the baseline approach attempted in [63]. Compared to their more sophisticated approaches, QPredict shows in some cases (Erfurt Combined) a better precision and up to 20% worse precision in other cases. One can also observe that QPredicts recall values are about 20% lower. We believe that the lower recall score of QPredict as compared to [63] stems from the fact that the classification task differs in its goals to detect not only changes but also additions and deletions and by the fact that more features about metadata and attributes have been included. In future work, these insights might help create better feature sets for QPredict and improve its classification performance.

6.2 Most influential features

A significance analysis yielded the most significant features for the combined feature set.

- *Extrinsic:* Distance, HausdorffDistance, Overlapping Degree
- *Intrinsic:* Positional Accuracy, Area
- *Metadata:* (Neighbourhood) Freshness, User Experience Score

Besides, further metadata metrics such as the number of days the user has been active in OSM, the size of the history of the geometry, and the number of edits of a user showed up in the significance result. In the extrinsic feature set, Distance and HausdorffDistance were the most influential features. The positional accuracy and the geometry area were most influential in the baseline feature set. We conclude that metrics commonly used in the ground-truth analysis are also effective when included in a change prediction approach.

6.3 Limitations of the approach

Our approach is limited by changes not reflected in the real world, e.g., changes in the data structure introduced by governmental authorities. Between 2013 and 2015, new national regulations forced structural changes in the target data set (reclassification of geometries). Such changes are hardly foreseeable from the source data set, as they are not reflected in, e.g., OSM geometries. However, we would argue that such changes are not frequent and previously announced, e.g., by the government. Therefore, The changes should already be known to occur by an educated user, especially by a mapping agency. Besides, some wrong classifications might be rooted in a lack of training examples for these changes. Geospatial changes can be very diverse and stem from factors that may not be reflected sufficiently often in the features of the investigated areas. Also, it should be noted that this experiment was conducted in a European country with a relatively high contributor activity in OpenStreetMap. As the change prediction depends on the user activity of the VGI data source, results may vary depending on the area's user contribution. Nonetheless, it may be fair to assume that data quality patterns observed in the European dataset may also be prevalent in other parts of the world, so a definite evaluation of this approach will need further studies. Finally, our approach cannot detect changes not documented in VGI data.

Figure 20 shows that sometimes authorities become aware of building changes that were not tracked by the OpenStreetMap community. QPredict cannot find classifications for these changes, as they are not documented in the source dataset used for classification.

6.4 A possible application case at a mapping agency

The evaluation of our algorithm showed the reliability of such an approach to a certain extent. In particular, we could show that predictions have a high precision score, i.e., not many false positive predictions. The classifier could be used in two contexts in the workflow of a governmental authority. Firstly, predictions could be visualized as an additional map layer to indicate areas currently investigated by the mapping agency. Secondly, the layer might be exposed to users of the official map to indicate uncertain areas. This concept could be extended to a situation-specific data quality framework as described in [30, 31]. The prediction could be used as one of many data quality parameters to indicate the

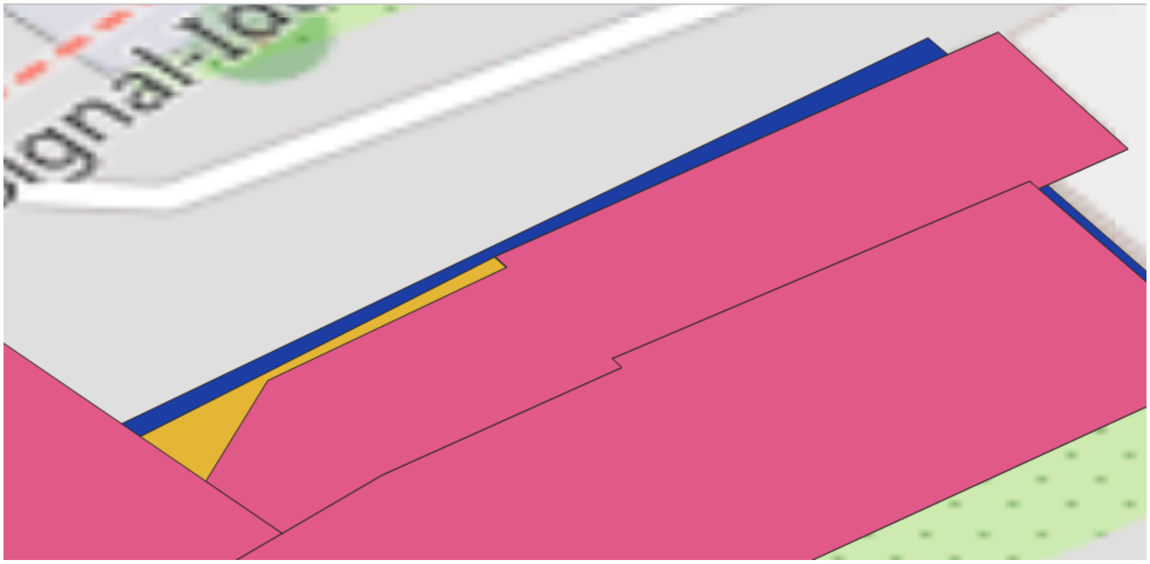


Fig. 20. A change of building size has happened between revision 2017 (pink) of the administrative data set and revision 2018 (yellow) of the administrative dataset. However, OpenStreetMap did not record this change in the year 2017 (indicated by blue in the graphic)

feasibility of application cases whose effectiveness could be affected by a change in the data set. For example, uncertain building and/or road information might affect a fire brigade mission. The fire brigade mission briefing could then notify firefighters of uncertain map quality for their area of operation. This information could lead to changing plans for the rescue mission or exercising caution in certain areas when navigating to the target.

However, one would also need to sketch its anticipated usage in a spatial data infrastructure to be usable in a mapping agency.

Geographical authorities provide data through spatial data infrastructures, which store, quality-assure, and provide geospatial data using, e.g., OGC geospatial web services. A sophisticated spatial data infrastructure would also provide a history of its geodata. In that sense, QPredict can be incorporated in this updated workflow as shown in Figure 21. Firstly, QPredict-train should be executed whenever a new dataset is integrated/updated to the spatial data infrastructure to increase the number of training examples on which QPredict-train operates. This approach would be done per geospatial feature type. Still, it may be extended to cover many geospatial feature types, resulting in machine learning models for various deemed practical machine learning feature sets. Throughout the ordinary revision cycle, QPredict-classify may be executed periodically to uncover potential transfers of VGI data to MGI data for the respective data set.

The results can be used twofold:

(1) Staff of the mapping agency gets recommendations about spatial objects that may need to be revised. Their feedback may be used to improve the accuracy of QPredict through reinforcement learning
(2) Researchers or experts at the mapping agency may use the wealth of geospatial information provided to improve the algorithm by testing different combinations of features, possibly yielding better or worse results for different geometry types

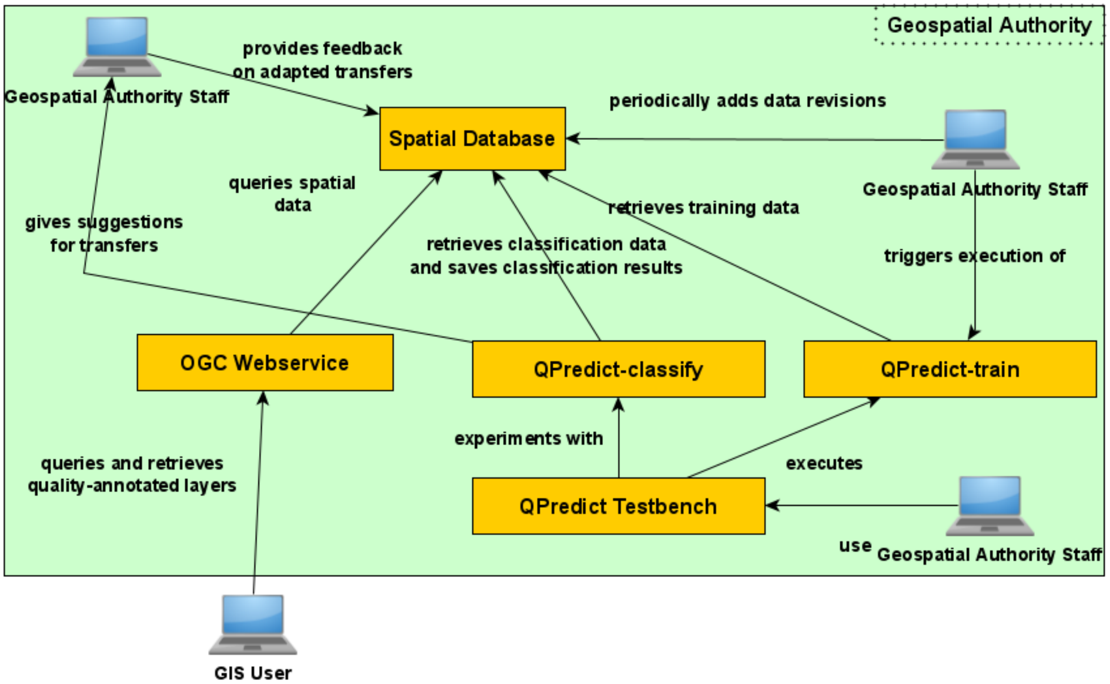


Fig. 21. QPredict in the update process of a mapping agency: QPredict-train is executed when the staff of the geographic authority performs an update to the spatial database. A testbench for selecting features and managing training and test sets allows for the targeted execution of QPredict. The staff of the geographic authority audits suggested transfers by QPredict, and feedback on meaningful transfers is fed into the spatial database as a feature for further classifications to improve accuracy.

The benefit for the user of the geospatial data from the geographic authority is, therefore, increased awareness about possible uncertain data, better planning for their respective use cases, and the possibility of including these results in case-specific data quality assessments as described in [30, 31].

7 CONCLUSIONS

We presented QPredict, a machine learning approach to indicate if changes present in a perceived lower quality data source should be applied to a perceived higher quality data source. We applied our algorithm to areas of two German cities and evaluated its effectiveness. We found that the algorithm shows a high precision for most classes and performs best on non-changing geospatial objects. This leaves us with a classifier that can classify changes that will be reflected in the next revision of a geospatial data set with a high probability. Despite yielding an average recall for geometry changes, the classifier does not yield the same results for additions and deletions. All in all, we have shown three aspects in this article. Firstly, classifying changes in geospatial objects using a machine learning approach is feasible. The classifier is useful enough as an indicator for mapping agencies to improve local datasets, and we give a baseline for other researchers to improve the precision and recall of our classifications. Secondly, the classification benefits from the inclusion of not only extrinsic data quality metrics but also intrinsic and metadata quality metrics. Thirdly, we have illustrated how QPredict could be used as a central component for data inspection in the spatial data infrastructure of a mapping agency.

Future work could deal with this improvement or conduct similar experiments on test cases we have neglected so far, such as comparing one geometry to many other corresponding geometries. We also assume that the QPredict approach can, without much effort, be applied to geometry types other than building footprints. This might require a change in the feature set but could be explored in a future publication.

Also, we would like to try the classification method for different kinds of areas. Rural areas might show different editing errors than metropolitan areas, and other mapping communities in OSM might be more or less accurate depending on the location. Another approach could be to predict which OSM geometries pick up changes found in future revisions of target data and, in this context, how to predict a good geometry edit in OSM. Finally, we are interested in exploring how long inconsistencies persist in the official government data and finding reasons why inconsistencies have not been resolved in a more timely manner.

REFERENCES

[1] Helmut Alt and Michael Godau. 1995. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications* 5, 01n02 (1995), 75–91.

[2] AdV-Arbeitsgruppe ATKIS. 2010. Amtliches Topographisch-Kartographisches Informationssystem.

[3] Adrian Baer, C Eastman, and M_ Henrion. 1979. Geometric modelling: a survey. *Computer-Aided Design* 11, 5 (1979), 253–272.

[4] Christopher Barron, Pascal Neis, and Alexander Zipf. 2014. A comprehensive framework for intrinsic OpenStreetMap quality analysis. *Transactions in GIS* (2014), 877–895.

[5] Christopher Barron, Pascal Neis, and Alexander Zipf. 2014. iOSMAnalyzer-ein Werkzeug für intrinsische OSM-Qualitätsuntersuchungen. *a* (2014).

[6] James Biagioni and Jakob Eriksson. 2012. Inferring road maps from global positioning system traces: Survey and comparative evaluation. *Transportation research record* 2291, 1 (2012), 61–71.

[7] Loreto Bravo and M. Andrea Rodríguez. 2009. Semantic Integrity Constraints for Spatial Databases. In *Proceedings of the 3rd Alberto Mendelzon International Workshop on Foundations of Data Management, Arequipa, Peru, May 12-15, 2009 (CEUR Workshop Proceedings, Vol. 450)*, Marcelo Arenas and Leopoldo E. Bertossi (Eds.). CEUR-WS.org. http://ceur-ws.org/Vol-450/paper8.pdf

[8] Gargi Chaudhuri and Keith C. Clarke. 2014. Temporal Accuracy in Urban Growth Forecasting: A Study Using the SLEUTH Model. *Transactions in GIS* 18, 2 (2014), 302–320. https://doi.org/10.1111/tgis.12047 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.12047

[9] Rodolphe Devillers, Robert Jeansoulin, and Michael F Goodchild. 2006. *Fundamentals of spatial data quality.* Number ISBN: 1905209568. ISTE London.

[10] Helen Dorn, Tobias Törnros, and Alexander Zipf. 2015. Quality evaluation of VGI using authoritative data—A comparison with land use data in Southern Germany. *ISPRS Journal of Geo-Information* 4, 3 (2015), 1657–1671.

[11] Jane Drummond. 1995. Positional accuracy. *Elements of spatial data quality* (1995), 31–58.

[12] Melanie Eckle and João de Albuquerque. 2015. Quality Assessment of Remote Mapping in OpenStreetMap for Disaster Management Purposes. In *ISCRAM*.

[13] Max Egenhofer and Robert Franzosa. 1991. Point-set topological spatial relations. *International Journal of Geographical Information System* 5, 2 (1991), 161–174.

[14] Thomas Eiter and Heikki Mannila. 1994. Computing discrete Fréchet distance.

[15] OR Abd El-Kawy, JK Rød, HA Ismail, and AS Suliman. 2011. Land use and land cover change detection in the western Nile delta of Egypt using remote sensing data. *Applied Geography* 31, 2 (2011), 483–494.

[16] Ahmed El-Rabbany. 2002. *Introduction to GPS: the global positioning system.*

[17] Chové et. al. 2017. Osmose. Data consistency checks for OSM.

[18] Goodchild et. al. 1997. A simple positional accuracy measure for linear features. *International journal of geographical information science* 11, 3 (1997), 299–306.

[19] Neis et. al. 2010. Empirische Untersuchungen zur Datenqualität von OpenStreetMap-Erfahrungen aus zwei Jahren Betrieb mehrerer OSM-Online-Dienste. *Angewandte Geoinformatik 2010* (2010).

[20] Rodriguez-Galiano et. al. 2012. An assessment of the effectiveness of a random forest classifier for land-cover classification. *ISPRS Journal of Photogrammetry and Remote Sensing* 67 (2012), 93–104.

[21] Sjahputera et. al. 2008. GeoCDX: An automated change detection & exploitation system for high resolution satelite imagery. In *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*, Vol. 5. IEEE, V–467.

[22] Schoof et. al. 2012. ATKIS-Basis-DLM und OpenStreetMap–Ein Datenvergleich anhand aus-gewählter Gebiete in Niedersachsen. *Untersuchung der Nutzung von OpenStreetMap Daten zur Darstellung von TMC Verkehrsmeldeinformatio* (2012).

[23] Michael F Goodchild. 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal* 69, 4 (2007), 211–221.

[24] G Gösseln and M Sester. 2005. Change detection and integration of topographic updates from ATKIS to geoscientific data sets. *Next Generation Geospatial Information; CRC Press: Boca Raton, FL, USA* (2005), 85–100.

[25] John C. Gower. 1982. Euclidean distance geometry. *Math. Sci* 7 (1982), 1–14.

[26] Li et. al GUO. 2009. Study for Area Feature Matching Technique Based on Area Similarity [J]. *Hydrographic Surveying and Charting* 3 (2009), 006.

[27] Timothy C Hart and Paul A Zandbergen. 2013. Reference data and geocoding quality: Examining completeness and positional accuracy of street geocoded crime incidents. *Policing: An International Journal of Police Strategies & Management* 36, 2 (2013), 263–294.

[28] Robert Hecht, Carola Kunze, and Stefan Hahmann. 2013. Measuring completeness of building footprints in OpenStreetMap over space and time. *ISPRS Journal of Geo-Information* 2, 4 (2013), 1066–1091.

[29] Geoffrey Holmes, Andrew Donkin, and Ian H Witten. 1994. Weka: A machine learning workbench. In *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*. IEEE, 357–361.

[30] Timo Homburg. 2020. Connecting Semantic Situation Descriptions with Data Quality Evaluations—Towards a Framework of Automatic Thematic Map Evaluation. *Information* (2020). https://doi.org/10.3390/info11110532

[31] Timo Homburg and Frank Boochs. 2018. Situation-dependent data quality analysis for geospatial data using Semantic technologies. In *International Conference on Business Information Systems*. Springer, 566–578.

[32] Daniel P. Huttenlocher, Gregory A. Klanderman, and William J Rucklidge. 1993. Comparing images using the Hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence* 15, 9 (1993), 850–863.

[33] ISO. 2013. ISO 19157. Geographic information–data quality.

[34] ISO 8000-8 2015. *Data quality – Part 8: Information and data quality: Concepts and measuring*. Standard. Geneva, CH.

[35] ISO19113. 2002. Geographic Information - Quality principles.

[36] ISO19114. 2003. Geographic Information - Quality evaluation procedures.

[37] Suming Jin, Limin Yang, Patrick Danielson, Collin Homer, Joyce Fry, and George Xian. 2013. A comprehensive change detection method for updating the National Land Cover Database to circa 2011. *Remote Sensing of Environment* 132 (2013), 159–175.

[38] Wolfgang Kainz. 1995. Logical consistency. *Elements of spatial data quality* 202 (1995), 109–137.

[39] Hamidreza Keshtkar, Winfried Voigt, and Esmaeil Alizadeh. 2017. Land-cover classification and analysis of change using ML classifiers and multi-temporal remote sensing imagery. *Arabian Journal of Geosciences* 10, 6 (2017), 154.

[40] Harald Kleiner. 2017. Keep right. Data consistency checks for OSM.

[41] Dengsheng Lu, P Mausel, E Brondizio, and Emilio Moran. 2004. Change detection techniques. *International journal of remote sensing* 25, 12 (2004), 2365–2401.

[42] Ross S Lunetta and John G Lyon. 2004. *Remote sensing and GIS accuracy assessment*. CRC press.

[43] Peter Mooney and Padraig Corcoran. 2011. Accessing the history of objects in OpenStreetMap. In *Proceedings AGILE*. 155.

[44] Peter Mooney, Padraig Corcoran, and Adam C Winstanley. 2010. Towards quality metrics for OpenStreetMap. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*. ACM, 514–517.

[45] Pascal Neis, Marcus Goetz, and Alexander Zipf. 2012. Towards automatic vandalism detection in OpenStreetMap. *ISPRS Journal of Geo-Information* 1, 3 (2012).

[46] Pascal Neis and Alexander Zipf. 2012. Analyzing the contributor activity of a volunteered geographic information project—The case of OpenStreetMap. *ISPRS Journal of Geo-Information* 1, 2 (2012).

[47] Ana-Maria Olteanu-Raimond, Glen Hart, Giles M Foody, Guillaume Touya, Tobias Kellenberger, and Demetris Demetriou. 2017. The scale of VGI in map production: A perspective on European National Mapping Agencies. *Transactions in GIS* 21, 1 (2017), 74–90.

[48] Veronika Peralta. 2006. Data freshness and data accuracy: A state of the art. *Instituto de Computacion, Facultad de Ingenieria, Universidad de la Republica* (2006).

[49] Claire Prudhomme, Timo Homburg, Ponciano Jean-Jacques, Frank Boochs, Ana Roxin, and Christophe Cruz. 2017. Automatic Integration of Spatial Data into the Semantic Web. In *Proceedings of the 13th International Conference on Web Information Systems and Technologies - Volume 1: WEBIST,* (Porto, Portugal). INSTICC, SciTePress, 107–115. https://doi.org/10.5220/0006306601070115 Best Student Paper Award.

[50] HB Qi, ZL Li, and Junnan Chen. 2010. Automated change detection for updating settlements at smaller-scale maps from updated larger-scale maps. *Journal of Spatial Science* 55, 1 (2010), 133–146.

[51] Frederik Ramm, Jochen Topf, and Steve Chilton. 2011. *OpenStreetMap: using and enhancing the free map of the world*. UIT Cambridge Cambridge.

[52] Vermessung RLP. 2010. Der neue Standard für Geobasisinformationen: Leitfaden zur Umstellung auf AFIS-ALKIS-ATKIS. 18 (2010).

[53] François Salgé. 1995. Semantic accuracy. *Elements of spatial data quality* (1995), 139–151.

[54] Angela Schwering. 2008. Approaches to Semantic Similarity Measurement for Geo-Spatial Data: A Survey. *Transactions in GIS* 12, 1 (2008), 5–29.

[55] Rade Stanojevic, Sofiane Abbar, Saravanan Thirumuruganathan, Sanjay Chawla, Fethi Filali, and Ahid Aleimat. 2018. Robust road map inference through network alignment of trajectories. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 135–143.

[56] Ingo Steinwart and Andreas Christmann. 2008. *Support vector machines*. Springer Science & Business Media.

[57] Daniel Sui, Sarah Elwood, and Michael Goodchild. 2012. *Crowdsourcing geographic knowledge: volunteered geographic information (VGI) in theory and practice.* Springer Science & Business Media.

[58] M Van Exel, E Dias, and S Fruijtier. 2010. The impact of crowdsourcing on spatial data quality indicators. In *Proceedings of the GIScience 2010 Doctoral Colloquium, Zurich, Switzerland.* 14–17.

[59] Pepijn AJ van Oort. 2006. *Spatial data quality: from description to application.* Wageningen Universiteit.

[60] Chang-Rong Xu, Jing Shen, and Kui-Yuan Shi. 2006. Research on GIS interoperability based on FME. *Journal of Southern Institute of Metallurgy* (2006).

[61] Peng Yue and Lianlian He. 2009. Geospatial data provenance in cyberinfrastructure. In *Geoinformatics, 2009 17th International Conference on.* IEEE, 1–4.

[62] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Soeren Auer. 2016. Quality assessment for linked data: A survey. *Semantic Web* 7, 1 (2016), 63–93.

[63] Xiaodong Zhou, Zhe Chen, Xiang Zhang, and Tinghua Ai. 2018. Change Detection for Building Footprints with Different Levels of Detail Using Combined Shape and Pattern Analysis. *ISPRS Journal of Geo-Information* 7, 10 (2018), 406.

A MORE DETAILED RESULTS

Algorithm	Feature set	Class	Precision	Recall	F-Score
RandForest	Baseline	Overall	90%	92%	90%
RandForest	Baseline	Unchanged	94%	97%	96%
RandForest	Baseline	Changed	37%	26%	30%
RandForest	Baseline	Added	**36%**	**18%**	**24%**
RandForest	Baseline	Deleted	23%	12%	16%
IBk	Baseline	Overall	**94.8%**	**94.9%**	**94%**
IBk	Baseline	Unchanged	**97%**	**97.6%**	**97.3%**
IBk	Baseline	Changed	**95.0%**	**52%**	**51%**
IBk	Baseline	Added	17%	15.3%	16%
IBk	Baseline	Deleted	**74%**	**57%**	**64%**
RandForest	Extrinsic	Overall	89%	91%	90%
RandForest	Extrinsic	Unchanged	94%	97%	95%
RandForest	Extrinsic	Changed	20%	11%	15%
RandForest	Extrinsic	Added	95%	51%	66%
RandForest	Extrinsic	Deleted	0.04%	0.02%	0.03%
IBk	Extrinsic	Overall	89%	88%	88%
IBk	Extrinsic	Unchanged	94%	93%	94%
IBk	Extrinsic	Changed	16%	16%	16%
IBk	Extrinsic	Added	60%	50%	54%
IBk	Extrinsic	Deleted	6%	8%	7%
RandForest	Combined	Overall	**96%**	**96%**	**95%**
RandForest	Combined	Unchanged	**96%**	**99%**	**98%**
RandForest	Combined	Changed	**74%**	31%	44%
RandForest	Combined	Added	**78%**	10%	20%
RandForest	Combined	Deleted	**98%**	**89%**	**93%**
IBk	Combined	Overall	95%	95%	95%
IBk	Combined	Unchanged	97%	97%	97%
IBk	Combined	Changed	53%	**54%**	**54%**
IBk	Combined	Added	24%	**21%**	**22%**
IBk	Combined	Deleted	80%	61%	69%

Table 5. Classification Results Area I - Erfurt

Algorithm	Feature set	Class	Precision	Recall	F-Score
RandForest	Baseline	Overall	83.0%	85.0%	84.0%
RandForest	Baseline	Unchanged	89.4%	95.5%	92.0%
RandForest	Baseline	Changed	**59%**	40.0%	48.8%
RandForest	Baseline	Added	35%	19%	25%
RandForest	Baseline	Deleted	51%	36%	42%
IBk	Baseline	Overall	**92%**	**92%**	**92%**
IBk	Baseline	Unchanged	**95%**	**96%**	**95%**
IBk	Baseline	Changed	57%	**57%**	**57%**
IBk	Baseline	Added	**38%**	**31%**	**34%**
IBk	Baseline	Deleted	**83%**	**60%**	**70%**
RandForest	Extrinsic	Overall	80%	84%	80%
RandForest	Extrinsic	Unchanged	87%	**96%**	91%
RandForest	Extrinsic	Changed	**50%**	**30%**	**38%**
RandForest	Extrinsic	Added	**50%**	14%	22%
RandForest	Extrinsic	Deleted	**49%**	13%	20%
IBk	Extrinsic	Overall	**86.2%**	**85%**	**86%**
IBk	Extrinsic	Unchanged	**92.4%**	92.2%	**92.3%**
IBk	Extrinsic	Changed	19%	20%	19.4%
IBk	Extrinsic	Added	40%	**18%**	**24%**
IBk	Extrinsic	Deleted	33%	**25%**	**28%**
RandForest	Combined	Overall	**95%**	**95.5%**	**95.1%**
RandForest	Combined	Unchanged	**96%**	**98%**	**97%**
RandForest	Combined	Changed	**72%**	48%	58%
RandForest	Combined	Added	**80%**	16%	28%
RandForest	Combined	Deleted	**96%**	**81%**	**88%**
IBk	Combined	Overall	92.7%	92.9%	92.8%
IBk	Combined	Unchanged	96.0%	96.3%	96.1%
IBk	Combined	Changed	59%	**58%**	**60%**
IBk	Combined	Added	39%	**32%**	**35%**
IBk	Combined	Deleted	84.2%	61.8%	71.3%

Table 6. Classification Results Area II - Jena

B LIST OF FEATURES

This annex introduces the feature sets that were used throughout this publication.

B.1 Baseline feature set

The baseline feature set includes intrinsic features that were used for the classification.

Featurename	Category	Applied on	Application	Description	VariableType	Domain	Reference
Amount Of Attributes	Intrinsic Metric	Attribute	Source and Target	Returns the amount of attributes	Integer	>=0	
Area	Intrinsic Metric	Attribute	Source and Target	Returns the amount of attributes	Double	>=0	
Geometry Closedness	Intrinsic Metric	Geometry	Target and Source	Indicates if the geometry is closed	Binary	TRUE/FALSE	
Geometry Emptyness	Intrinsic Metric	Geometry	Target and Source	Indicates if the geometry is empty i.e. contains no points	Binary	TRUE/FALSE	
Geometry ID	Intrinsic Metric	Geometry	Target and Source	Unique ID of a geometry	Binary	TRUE/FALSE	
Geometry Length	Intrinsic Metric	Geometry	Target and Source	The length of the geometry	Discrete	TRUE/FALSE	
Geometry Number Of Nodes	Intrinsic Metric	Geometry	Target and Source	The number of nodes per geometry	Discrete	TRUE/FALSE	
Geometry Number Of Geometries	Intrinsic Metric	Geometry	Target and Source	The number of geometries per geometry	Discrete	1-n	
Geometry Rectangularity	Intrinsic Metric	Geometry	Target and Source	Indicates if the geometry is rectangular	Binary	TRUE/FALSE	
Geometry Resolution	Intrinsic Metric	Geometry	Target and Source	The resolution of the geometry	Double	>=0	
Geometry Scale	Intrinsic Metric	Geometry	Target and Source	The scale of the geometry	Double	>=0	
Geometry Simplicity	Intrinsic Metric	Geometry	Target and Source	Indicates if the geometry is simple	Binary	TRUE/FALSE	
Geometry Validity	Intrinsic Metric	Geometry	Target and Source	Indicates if the geometry is valid	Binary	TRUE/FALSE	
AmountOfAttributes	Intrinsic Metric	Attribute	Source and Target	Returns the amount of attributes of the geometrical feature	Integer	>=0	
HistorySize	Intrinsic Metric	Attribute	Source and Target	Returns the amount of edits of this geometrical feature	Integer	>=0	
Freshness	Intrinsic Metric	Metadata	Source	Returns the amount of days since the geometry has been last modified	Double	>=0.00	[48]
Average Attribute Freshness	Intrinsic Metric	Attribute	Source and Target	Returns the average freshness of attributes associated with the data set	Double	>=0.00	[48]

B.2 Extrinsic Comparison feature set

The extrinsic feature set includes only extrinsic features that were used for the classification.

Featurename	Category	Applied on	Application	Description	VariableType	Domain	Reference
AreaSimilarity	Extrinsic Metric	Geometry	Target vs. Source	Calculates the AreaSimilarity between the target and source geometry	Discrete	> 0.0	[26]
Hausdorff distance	Extrinsic Metric	Geometry	Target vs. Source	Calculates the Hausdorff distance between the target and source geometry	Discrete	> 0.0	[32]
ContainsReference	Extrinsic Metric	Geometry	Target vs. Source	Calculates if the target geometry is contained by the target data set	Binary	TRUE/FALSE	[13]
DisjointWithReference	Extrinsic Metric	Geometry	Target vs. Source	Calculates if the target geometry is disjoint with the target data set	Binary	TRUE/FALSE	[13]
EuclideanDistance	Extrinsic Metric	Geometry	Target vs. Source	Calculates the distance of the center points of the target vs. source geometry	Discrete	>=0.0	[25]
GeoCodingCompleteness	Extrinsic Metric	Attribute	Target vs. Source	Checks for the completeness of attributes identifying geocoding attributes	Binary	TRUE/FALSE	[27]
IntersectionPercentage	Extrinsic Metric	Geometry	Target vs. Source	Calculates the percentage of intersection of the target vs. source geometry	Discrete	>=0.0	
EqualsExact	Extrinsic Metric	Geometry	Target vs. Source	Calculates if the two geometries are topologically the same	Binary	TRUE/FALSE	[13]
Positional Accuracy Difference	Extrinsic Metric	Geometry	Target and Source	Compares the positional accuracy	Double	>0.00	[18]
FrechetDistance	Extrinsic Metric	Geometry	Target vs.Source	Calculates the FrechetDistance between the target and source geometry	Discrete	> 0.0	[14]
SuperiorRepresentation	Extrinsic Metric	Geometry	Target vs.Source	Indicates if the target data set includes a more detailed representation of a geometry	Binary	TRUE/FALSE	
WithinReference	Extrinsic Metric	Geometry	Target vs. Source	Calculates if the target geometry is within the target data set	Binary	TRUE/FALSE	[13]
AttributeDifference	Extrinsic Metric	Geometry	Target vs. Source	Calculates the amount of different attributes among the two revisions	Discrete	>=0	

B.3 Combined feature set

The combined feature set includes all features defined in the baseline feature set and in the extrinsic feature set. Besides, it includes the following metadata features.

Featurename	Category	Applied on	Application	Description	VariableType	Domain	Reference
Attribute Difference per Feature	Extrinsic Metric	Attribute	Target vs. Source	Returns the number of conflicts between matching and attributes of corresponding geometries	Double	>=0.00	[48]
Amount of Users	User Metric	Metadata	Source	Retrieves the amount of users editing this geometry	Double	>0.00	[46]
Average User Experience	User Metric	Metadata	Source	Retrieves the average user experience	Double	>0.00	[46]
Average User Mapping Days	User Metric	Metadata	Source	The average amount of mapping days per user	Double	>0.00	[46]
ChangeRate from History	User Metric	Metadata	Source	Retrieves the average user experience	Double	>0.00	[58]
HasUniqueRecognizedClass	Metadata Metric	Metadata	Source	Checks if the geometry can be assigned a unique semantic web class	Binary Metric	TRUE/FALSE	[58]
History Size	Metadata Metric	Metadata	Source	Retrieves the history size of the source geometry	Double	>0.00	[43]
IsInUsersMainAreaOfEdit	User Metric	Metadata	Source	Indicates if the users focus is on editing the area in which the current geometry is situated	Boolean	TRUE/FALSE	[58]
OverlapsWithNeighbourGeometry	Intrinsic Metric	Geometry	Target and Source	Indicates if a geometry overlaps with another geometry in the same data set	Boolean	TRUE/FALSE	[13]
User Mapper Type	User Metric	Metadata	Source	Indicates the average activity of a user overall	Integer	0-1E5	[46]
UserLastModifierOf	User Metric	Metadata	Source	Checks if the amount of last modifications the last editing user made	Double	>0.00	