RESEARCH

Curami: An Assisted Attribute Curation Tool for the BioSamples Database.

Matthew Green*

Correspondence:
hewgreen@ebi.co.uk
European Bioinformatics Institute, European Molecular Biology Laboratory (EMBL), Wellcome Trust Genome Campus, CB10 1SD Cambridge, UK
Full list of author information is available at the end of the article
*Equal contributor

Abstract

First part title: Text for this section.

Second part title: Text for this section.

Keywords: metadata; biosamples; assisted curation; data quality

Background

Architects of biological databases are forced to decide which trade offs to make in order to best balance the data consumer's downstream requirements; a data contributor's input effort versus the value-added benefits and the project's available resources. Whilst inexplicit validation and pliable data schema lower the barrier to initial data submission, these practices result in unstructured data that is difficult to reuse without incurring high curation costs. Many tools have been built which aim to improve data quality whilst minimising incurred financial and time costs. These tools broadly operate either pre-data submission (validation) or post-data submission (curation). Within these two strategies, the tools are spread on a scale between fully automated or fully manual. Often, increased manual effort correlates with increased data value and diminishing throughput [1]. Therefore, a tool should be designed to suit the specific requirements of the resource's stakeholders. Thankfully, there are applications that successfully leverage the benefits of automated data assessment and combine this with expert manual curation to measurably increase the efficiency of manual curation efforts [2, 3, 4]. Curami, the tool described herein, demonstrates how such a tool can increase the productivity of domain expert curators (who could also be data contributors or consumers) by asking them the right questions and maximising their insight.

The BioSamples database at the European Bioinformatics Institute (EBI) is a metadata repository for all biological samples used to generate datasets across the EBI [5, 6]. The database consists of more than 4.5 million sample records which each contain a unique BioSamples identifier. Many records also have links to the sample's derived data in external databases, relationships to other samples (derived from, child of etc.) and sample metadata expressed as key value pairs (note that keys are hereinafter referred to as attributes). As the database provides a single point of entry to metadata from a wide range of scientific domains and technologies, BioSamples allows users to create ad hoc fields that are not predefined. Whilst this increases utility and flexibility for submitters, the dataset suffers from redundancy and inconsistency. For example, there are 22 attributes that contain longitudes.

Although BioSamples mitigates this problem by providing elastic search with ontology expansion, the samples returned may not necessarily be findable, accessible, interoperable and reusable, the universal targets defined as the FAIR principles [7]. A snapshot of the dataset containing 4,790,415 sample records (taken on the 5th January 2018) was used for initial analysis. These samples contained 29,751 unique attributes used conjointly 45,275,314 times, giving a mean average of 9.45 attributes per sample record. The redundancy and inconsistency within the large number of attributes in the BioSamples database prompted the creation of this assisted-curation tool.

The overarching goal was to provide a tool that facilitates the reduction of redundancy and inconsistency of the attributes in the BioSamples database, ultimately to reduce the total number of attributes in the database without loosing information. In order to achieve this, we aimed to identify pairs of attributes that were semantically similar enough to merge and to further identify the correct polarity of the merging. Although these aims differ from recent work to identify more generally topically related attributes in the Gene Expression Omnibus (GEO) [8, 9], the strategy these authors employ highlights some key challenges to solving this problem [10]. Whilst GEO encourages data submitters to conform to the Minimum Information About a Microarray Experiment (MIAME) guidelines [11], similarly to BioSamples, the lack of a controlled vocabulary leads to the usage of different terms to represent the same concept. The authors developed a clustering methodology to group attributes that covered similar concepts and in doing so aim to separate curation activities into more manageable chunks. Unfortunately, applying clustering methodologies to data with heavily biased distributions leads to biased results, which impedes semantic understanding. Furthermore, as is the case for BioSamples, the majority of metadata submissions come from relatively few pipelines which often impose their own validation, recommendations and guidelines leading to further clustering bias and rendering frequency of term usage a less definitive predictive metric.

Methods

Source code and application documentation is available for inspection and reuse at *https://github.com/EBIBioSamples/curami*.

Overview of Data and Work Flow

During initialization, each sample record is requested via the BioSamples Application Programming Interface (API) and parsed into four input files which represent a snapshot of the BioSamples database (see figure 1). i.e. A list of unique attributes and their frequency of usage, a list of sample records (including the BioSamples ID and the attributes the sample contains), a list of attribute pairs that co-occur within a sample record at least once with the frequency of the co-occurrence (all comma separated format) and a JSON file with unique values and their frequency of use for each unique attribute. The second step of initialisation uses these four files to create a network using the Python package networkX [12], which is exported as a .gexf for later use by the co-occurrences analysis script. This graph contains nodes representing individual attributes and edges weighted by co-occurrence and can also be

visualised using Gephi [13] and explored using various inbuilt force-directed layout algorithms [14] (see Figure [**?**]).

Attributes are paired (as later described) and various pairwise features are calculated and stored in a graph database using Neo4J. This database serves the frontend flask application by providing the statistics for each attribute pair and it also stores the user's curation decisions taken thereafter. Attribute co-occurrences are also stored within the graph, which provides recommendation engine functions through various cypher queries (as later described).

The workflow thus far has been automated and can therefore be triggered at regular intervals to keep up to date. The user interface is a flask application capable of navigating the attribute pairs, reviewing analysis and making curation decisions. For each pair the user can choose not to merge, merge the least frequently used attribute into the more popular of the pair or merge in the reverse direction. This simple interface takes inspiration from gamification concept in web app design by facilitating quick decision making and showing individual curators their comparative impact statistics. These decisions are also captured in the graph database and form the basis of both export of curation objects directly to the BioSamples API and training data for later machine learning and tuning. Curation objects can be generated directly from Curami's graph database and submitted to the BioSamples API to apply the curation to the attribute in the original BioSamples database. It is worth noting that in the current BioSamples implementation, if two or more conflicting curation objects are invoked on the same sample record, BioSamples users will see the original unedited record. Nevertheless, the curation objects will persist and can be later filtered to give users a customised view of sample records.

Attribute Pairing

Curami aims to both identify erroneous attributes in the dataset and find a suitable replacement. Both of these functions are required in order to successfully curate a sample record. In order to identify an erroneous attribute's replacement, we assume that the dataset contains a more suitable attribute most of the time. Hence, we are trying to identify an attribute's synonyms within the dataset and elect a preferred term to represent the group. This strategy aims to reduce the total number of attributes whilst maintaining meaningful semantic differentiation.

Curami has a tiered modular design to aid fungibility of the code, allowing new methods to be added at each tier as required. Figure 3 shows how initial high throughput attribute pairing is followed by secondary pairwise analysis, which calculate statistics used to populate the curation interface. The 29,751 attributes expand into 885,092,250 potential unique pairs, excluding self-matches. Therefore, the initial tier (high throughput pairing) filters this high number of potential pairs into a smaller number that is able to undergo more rigorous similarity computation. This layer is designed to be optimised for throughput and therefore may only rely on a low computational burden for pair selection. As such, the selected pairs at this stage are most likely not mergeable. To initially reduce the number of pairs, a Levenshtein distance [15] threshold of 0.8 was invoked resulting in 35,699 lexically matched pairs for further analysis. These pairs cover 17,376 out of 29,751 total attributes (58.4% coverage). This arbitrary threshold should be revised once a reasonable amount of pairs have been reviewed.

Once attribute pairs have been filtered to a more manageable number the second tier (pairwise analysis) can be performed. In this tier a range of analysis modules calculate various similarity features for each pair. The modular design is critical to encourage future development of new orthogonal approaches. One example of an additional module could be pairwise ontology comparisons. At the time of publication, Curami has four modules for feature calculation. These are mismatch typing, lexical similarity, inter-sample co-occurrence and value similarity. These modules calculate the 13 features outlined in table 1.

Co-occurrence Network

Co-occurrence refers to attributes that are used together within the same sample. By counting co-occurrence, we can quantify a relationship between attributes and calculate a relationship weighting. A high frequency of co-occurrence may indicate that the attributes provide distinct information and therefore should not be merged. However due to the skewed distribution of attribute usage, highly popular attributes will co-occur more frequently than less popular attributes. To account for frequency of use when calculating co-occurrence weighting, the following normalisation is performed.

If $C(a_1)$ is the number of occurrences of the first attribute, $C(a_2)$ the number of occurrences of the second and $C(a_t)$ the total occurrences of all attributes in the dataset, the probability of $C(a_1)$ occurrence ($P_{(a1)}$) is:

$$P_{(a1)} = \frac{C(a_1)}{C(a_t)}$$

Therefore, we can calculate the expected co-occurrence ($E[x]$) if attributes were randomly paired:

$$E[x] = P_{(a1)} \cdot P_{(a2)} \cdot P_{(t)}$$

The observed co-occurrence ($O[x]$) for each pair is transformed into a normalised weight ($W_{(a1,a2)}$) like so:

$$W_{(a1,a2)} = \frac{O[x] - E[x]}{\sum(O[x] - E[x])}$$

These weights are used to create a NetworkX co-occurrence graph. If the frequency of co-occurrence is zero, the weight will also be zero although these edges are not explicitly calculated to reduce overhead. Equally, the co-occurrence JSON only contains counts greater than zero. The theoretical weight range is +1 to -1 and all weights in the graph sum to 1. As the majority of BioSamples metadata records come from relatively few sources, downstream validation biases these weightings.

Data Features

Levenshtein Distance [15] and Jaro Score were calculated using Jellyfish 0.7. Edge Weight, Jaccard coefficient, break number and degree [16] were calculated using the co-occurrence graph and algorithms from NetworkX 2.1 [17]. Other features were calculated using python algorithms developed herein.

Results

State of Dataset

The 45,275,314 values over 5,172,345 samples in the Biosamples database is largely navigated via the attributes. These 29,753 attributes represent the only labels on

the data and further structuring of the dataset requires high quality in those labels. The reason for the strikingly high number of attributes is largely due to slight naming variations whereby many attributes are used to represent the same semantic definition. Therefore, merging these synonyms would have a high curation benefit for the database to make data more finable and interoperable.

The distribution of attribute usage is highly positively skewed, which indicates many attributes are rarely required to describe the data. Figure 4 A shows that relatively few attributes are heavily used whilst the majority of attributes rarely occur. 2459 attributes are only used once and the most frequently used attribute is 'Organism' which is used 4,727,867 times in 91.4% of samples. Although it would be reasonable to presume that the more popular attributes were more likely to be regarded as the better standard, high throughput programatic submission breaks this assertion. A clear example of this is the attribute 'collection_date' (the 3529th most popular attribute with 429 uses), which was more frequently mis-spelled 'colection_date' (the 7th most popular attribute with 928,381 uses). This is likely to be a error introduced in a high throughput programatic submission from ENA. However, in some instances attribute diversity may correspond to genuine data diversity, for example the informative attribute 'Ni (ppm)' only occurs once.

This example demonstrates that popularity of an attribute and it's quality are not always correlated. However, if this mostly holds true, there is a trade off between focusing on the curation of the more popular attributes that are less likely to be incorrect but have a greater impact on samples versus targeting less popular attributes that are more likely to be better represented by a better term but curation will have a low impact on the dataset. Put simply, when 60% of the attributes only cover 5% of the values is it worth while curating these? **Matt: We should use the app in a random mode (not available right now but could be quickly added or we could use 'smart' for now) to test the assumption that more popular attributes are 'better' than less popular ones.**

Figure 4 B again shows the trend that a few attributes occur frequently. The outliers coloured red are examples of attribute signatures representing potential clusters of attributes that are used frequently in the same samples. These attributes occur at exactly the same frequency, which suggests they may originate from the same source. For example, 2 attributes ('env feature' and 'env biome') both occur 102,528 times and indeed they co-occur 102,528 times. These are standardised attributes required by ENA. 255 attributes are used precisely 8,058 times. These are attributes defined by VioScreen, a standardised clinical dietary survey. These outliers are also clearly seen as tight clusters in the co-occurrence graph (see figure 2). These signatures display the submitter bias towards their preferred attribute terms. In some instances it would be useful to break up these clusters by homogenising (or mapping) equivalent terms although it is clear that in some cases submitter bias also represents unique submitter requirements that should be retained after curation.

Taking 'vioscreen zinc', 'zinc', 'Zinc', 'zinc concentration' and 'Zinc concentration' as a set of highly related attributes, merging may not be appropriate for all pairings. In this case, 'Zinc concentration', 'zinc' and 'vioscreen zinc' values are all continuous numeric ranges without units that differ by 10 fold. As range units are missing but seem consistent within attributes, merging would make it more difficult to

retrospectively add units. 'zinc concentration' and 'Zinc' are non-continuous values (e.g. '¡ 10 mg/l', 'high zinc' and 'low zinc') so merging these values may be more appropriate. Before a curator could make these decisions they would need to inspect the values associated to the attributes to consider the contributor's requirements as the values are not necessarily self describing.

Assessment of curation effectiveness

One curator using Curami for 20 minutes targeting frequently used attributes ('major attribute' setting) was able to review 200 attribute pairs. 40 of these pairs were merged or 'reverse merged' (where the application did not correctly determine the polarity of the merge) which ultimately edited 9M key value pairs in the dataset. This demonstrates the speed potential but also highlights the risk of false positives.

TODO: I suggest we do more tests with different people operating the app in the same mode to assess if different curators make the same decisions and operate at the same speed. Without this work I'm not sure what else we can put into this section. This could be compared to a gold standard for these pair merges that we would carefully assess.

Pair Sorting Settings

Curami supports several different user requirements via its sort settings. Users can select a setting to influence the order in which attribute pairs will be displayed to them for curation decisions. There are currently six sort settings. 1. The 'max sample impact' setting sorts pairs by the attribute that is least frequently used within the pair. This allows the curator to tackle frequently-used attributes and have a big impact on the dataset as this will ultimately result in creating the most curation objects for the Biosamples database, which are made at a per sample level. 2. The 'major attribute' setting sorts pairs by the attribute that is most frequently used within the pair. This setting allows curators to tackle popular erroneous attributes. As frequency of usage is not a perfect indicator of quality, it is often useful to closely interrogate these frequently-used attributes, which have a large overall impact on the dataset. 3. The 'smart sort' is a placeholder for the latest available prediction that an attribute will result in a merge decision. Prior to the availability of more accurate predictive features, this method sorts by match type favouring 'S Discrepancy', 'Just Specials Discrepancy' and 'Only Space Discrepancy', which would often suggest a merge is likely (see table 2). Later, this sorting method could use a prediction score that weights features from previous user decisions to predict which pairs should be merged.

The other three sort methods are designed for domain experts who only wish to curate a portion of the dataset that relates to their work. Topical sorting requires an attribute recommendation engine supported by graph queries across the co-occurrence network. The weighted co-occurrence graph of attributes facilitates attribute recommendation. The nearest neighbour attribute to the query attribute is considered to be the one which most frequently co-occurs within the same sample. High positive weightings imply that two attributes are used in the same types of samples and are often both required. It is also possible to find the nearest attribute to a list of query nodes using a sum of weightings.

The recommendation engine first requires a user to input some attributes they would be interested in reviewing. After free text input, the UI shows the user which attributes are in the dataset and which are not, and makes suggestions to find lexically similar matches. Once a user has selected one or more attributes that either they are interested in working on directly or they feel defines their area of interest, these attributes are saved to the user's profile. 4. The 'Review My Attributes' sort method limits the pairs to those that contain their chosen attributes. This setting allows users to work directly with the attributes they are interested in. This same sort method can be used by curators that spot mistakes in the dataset elsewhere. For example when using Biosamples directly, and wish to submit a curation suggestion. 5. The 'Related to My Attributes' sort option will act in the same way, allowing domain experts to review the pairs that contain the attributes they pre selected. It then goes further suggesting pairs that contain attributes closely related to their pre defined set. This distance is a sum of the weighted co-occurrence between their preselected attributes and attributes in the rest of the graph. This allows domain experts to move through the dataset inspecting attributes that are related to their domain of interest by co-occurrence within the same samples. 6. The final sort setting 'Dynamic Specialism' is similar to the last, other than it takes into account the attributes the user has previously made decisions about. If a user skips pairs, these are not used in the recommendation ranking. However, if a user makes positive or negative merge decisions those pairs are factored into the nearest neighbour calculation when traversing the graph. Early implementations of this method have significant speed impediments that need to be overcome before this method would be useful in a production environment.

Attribute Recommendation

Due to the diverse nature of the BioSamples data set, slicing attributes by specialism is a very important curator requirement. Ultimately, this should connect the curators to the relevant attribute pairs to make decisions where their expertise counts. Rather than predefining these slices, the application allows the user to select attributes that are commonly associated with the dataset they are interested in and uses this to suggest pairs that are relevant by looking at attributes that are closest according to the co-occurrence weighting.

TODO: an example of how a specialist may use the app, which pairs are predicted. I am blocked on this until I get the app running locally again!

The recommendation engine relies on the co-occurrence weightings. Table 3 a. shows the 5 strongest associated attributes which are all required in European Nucleotide Archive (ENA) checklists while 3 b. shows the weakest attribute associations. Nevertheless, weights are often intuitively relevant. For example, table 3 c. highlights three pairs that intuitively belong together (depth & elevation, serovar & strain and disease state & individual) and two pairs that occupy contrary domains which intuitively don't overlap (organism part & serovar and organism part & environment biome).

TODO: This section is missing some real user assessment of this functionality.

TODO: Make a figure showing screenshots of the recommendation functionality.I am blocked on this until I get the app running locally again!

Discussion

Curami is designed to be used from two perspectives. As an assisted curation application it can be used to navigate the user through their domains of interest within the dataset and provide auxiliary information to the curator, which may increase their confidence and allow them to make a curation decision. From this perspective the application is complete and its utility has been demonstrated. From a second perspective, decision capture is incredibly useful as a training set to later discover in which circumstances the calculated features become predictive indicators of a required curation operation. Although this work has not explored analysis of the captured features, this second utility was a major contributing factor to the design of Curami. Later this will allow identification of curation patterns, which can be used to make smarter curation predictions to leverage manual effort. The lack of gold standard training data dictated that as a first step, Curami had to provide immediate utility whilst generating training data for later correlation analysis.

The pairing rationale shown in figure 3 carries the assumption that erroneous attribute labels (for example mis-spelled attributes) can be paired with a correct version of the attribute (the elected representative) that must exist within the BioSamples dataset. This corrected attribute may not always be present but anecdotal exploration of the data suggested that it normally did. The pairing rationale also failed to appreciate that more than two synonymous attributes were often present. This may add to the required workload but layered pairwise decisions provide extra information to arbitrate final merge decisions.

The diversity of the BioSamples database added the requirement to slice the dataset by topic so that curators could target specific areas of interest. Curami's recommendation engine drives this feature displaying the targeted pairs to the curator for investigation. Further user tests are required to drive future feature development.

Future work should expand Curami's pairing capability with an new all-by-all pairing method orthogonal to lexical similarity. Discovery of this will require feature analysis of the training dataset generated by the first users of the tool. An ontology module should also be added to expand the number of pairwise feature calculations to further improve similarity detection.

TODO: Sample level curation is not discussed. This was not fully implemented but explored. The only clear example of it's use was with the 'smoker' 'smoke' mere where the clustering suggested only a few attributes should be merged. I'm not sure if this should be included.

Competing interests
The authors declare that they have no competing interests.

Author's contributions
Text for this section ...

Acknowledgements
Text for this section ...

References

1. Goble, C., Stevens, R., Hull, D., Wolstencroft, K., Lopez, R.: Data curation+ process curation= data integration+ science. Briefings in bioinformatics **9**(6), 506–517 (2008)
2. Salgado, D., Krallinger, M., Depaule, M., Drula, E., Tendulkar, A.V., Leitner, F., Valencia, A., Marcelle, C.: Myminer: a web application for computer-assisted biocuration and text annotation. Bioinformatics **28**(17), 2285–2287 (2012)
3. Salimi, N., Vita, R.: The biocurator: connecting and enhancing scientific data. PLoS computational biology **2**(10), 125 (2006)
4. Szostak, J., Ansari, S., Madan, S., Fluck, J., Talikka, M., Iskandar, A., De Leon, H., Hofmann-Apitius, M., Peitsch, M.C., Hoeng, J.: Construction of biological networks from unstructured information based on a semi-automated curation workflow. Database **2015** (2015)
5. Gostev, M., Faulconbridge, A., Brandizi, M., Fernandez-Banet, J., Sarkans, U., Brazma, A., Parkinson, H.: The biosample database (biosd) at the european bioinformatics institute. Nucleic acids research **40**(D1), 64–70 (2011)
6. Faulconbridge, A., Burdett, T., Brandizi, M., Gostev, M., Pereira, R., Vasant, D., Sarkans, U., Brazma, A., Parkinson, H.: Updates to biosamples database at european bioinformatics institute. Nucleic acids research **42**(D1), 50–52 (2013)
7. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L.B., Bourne, P.E., et al.: The fair guiding principles for scientific data management and stewardship. Scientific data **3** (2016)
8. Edgar, R., Domrachev, M., Lash, A.E.: Gene expression omnibus: Ncbi gene expression and hybridization array data repository. Nucleic acids research **30**(1), 207–210 (2002)
9. Barrett, T., Wilhite, S.E., Ledoux, P., Evangelista, C., Kim, I.F., Tomashevsky, M., Marshall, K.A., Phillippy, K.H., Sherman, P.M., Holko, M., *et al.*: Ncbi geo: archive for functional genomics data setsóupdate. Nucleic acids research **41**(D1), 991–995 (2012)
10. Hu, W., Zaveri, A., Qiu, H., Dumontier, M.: Cleaning by clustering: methodology for addressing data quality issues in biomedical metadata. BMC bioinformatics **18**(1), 415 (2017)
11. Brazma, A., Hingamp, P., Quackenbush, J., Sherlock, G., Spellman, P., Stoeckert, C., Aach, J., Ansorge, W., Ball, C.A., Causton, H.C., *et al.*: Minimum information about a microarray experiment (miame)ótoward standards for microarray data. Nature genetics **29**(4), 365 (2001)
12. Hagberg, A., Schult, D., Swart, P.: NetworkX. https://github.com/networkx/networkx Accessed 13.4.18
13. Bastian, M., Heymann, S., Jacomy, M., *et al.*: Gephi: an open source software for exploring and manipulating networks. Icwsm **8**, 361–362 (2009)
14. Jacomy, M., Venturini, T., Heymann, S., Bastian, M.: Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. PloS one **9**(6), 98679 (2014)
15. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady, vol. 10, pp. 707–710 (1966)
16. Jaccard, P.: Distribution of the alpine flora in the dranseís basin and some neighbouring regions. Bulletin de la Societe vaudoise des Sciences Naturelles **37**, 241–272 (1901)
17. Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (2008)
18. Fruchterman, T.M., Reingold, E.M.: Graph drawing by force-directed placement. Software: Practice and experience **21**(11), 1129–1164 (1991)
19. Newman, M.E.: Modularity and community structure in networks. Proceedings of the national academy of sciences **103**(23), 8577–8582 (2006)

Figures

Figure 1 Data Flow During initialization Curami crawls BioSamples using the API and generates four input files. The input files are used to populate a Neo4J database which is further populated with co-occurrence weighting and other pairwise measurements shown in table 1. The flask app Curami uses cypher queries to find attribute pairs to display to the user. As the user takes curation decisions these are stored in the Neo4J DB which can then be used to periodically create curation objects that can be posted back to the BioSamples API.

Figure 2 Gephi snapshot Snapshot from the interactive network layout tool Gephi [13]. Each attribute is a node and the node size represents the approximate frequency of attribute usage (size is binned rather than scaled). The Fruchterman-Reingold layout [18] uses cooccurance weighting captured on the edges which are represented as grey lines between nodes. Colour assignment was done using the inbuilt Newman's modularity function [19] to loosely highlight node clusters. Some specific attributes are labelled along with some large clusters that show many attributes related to 'human metagenome' samples and a commonly used 'medical screen'.

Tables

Figure 3 Attribute Pairing Overview of the modular design of Curami showing separation of modules and general dataflow. Initialization and analysis modules 1-5 should be executed in series to pull information from the BioSamples API to generate input files, pair lexically similar attributes, extract pairwise and individual features and store the results in the Neo4J database. The Data Analysis Store has three node types; User, Pair and Attribute, with corresponding relationships as shown. Edges *r1* represent curation decisions made by the user about a pair of attributes and can be one of four types; merge, merge with reversed polarity, don't merge and skip. Edged *r2* has only one type, 'pair contains', and indicates which individual attributes are in the pair. The Curation Interface allows curators to view the attribute pairs sorted by applying one of the available settings. Curation decisions are then stored in the Neo4J database which can then be directly converted into curation objects, a format that can be directly submitted to the BioSamples API.

Figure 4 frequency of attribute occurrence A, B

Table 1 A table of analysis features calculated for each attribute pair.

Feature	Description
Levenshtein Distance	[15]. N.B. pairs that score ¡ 0.8 are not included.
Attribute 1 frequency	The frequency of usage of attribute 1. This is used to predict the polarity of the merge. This assumes that the most popular attribute is most likely to be correct.
Attribute 2 frequency	The frequency of usage of attribute 2.
Discrepancy Type	The result of several lexical tests (see table 2)
Camel Case	Boolean indicating lexical detection of potential camel case usage.
Edge Weight	placeholder
Jaccard Coefficient [16]	Calculated using the co-occurrence NetworkX graph. Quantifies the overlap of usage between attributes which may not directly co-occur (have a weighted edge in the graph).
Break Number	placeholder
Degree	placeholder
Edge Total	placeholder
Value Match Type	Can be either numeric, date or string match if more than 90% of the values adhere to one data type. Alternatively, the field will show a mixed match or no match.
Order of Magnitude	placeholder
Jaro Score	placeholder

Table 2 These tests aim to catagorize the lexical difference between the lexically similar attributes. ‡ are strong candidates for automated curation.

Lexical Catagory	Description of Difference Test
Number Discrepancy	A numerical character differentiates the attributes.
Case Discrepancy	If case stripping improves the Levenshtein score, case is identified as at least one differential factor.
Space Discrepancy	Indicates if a space is at least one part of the discrepancy.
Only Space Discrepancy‡	If a space difference is the only difference between the attributes.
Specials Discrepancy	If special characters are partially responsible for the difference.
Just Specials Discrepancy‡	If special characters are the only difference between the attributes.
Word Number Discrepancy	If the number of tokens in the attributes differ.
Stop Word Discrepancy	If the difference is due to the presence of stopwords (from the NLTK corpus imported from nltk.corpus import stopwords).
Dictionary Matching	Triggered if there is a difference in spelling between the attribute's tokens (so ignoring equally misspelled tokens) that fail a dictionary test against enchants US dictionary after stripping numbers, case and special characters.
Lemmatisation Matching	If lemmatisation or stemming produces an identical match.
S discrepancy‡	If an additional 's' character is the only difference between attributes.

Table 3 a. Most strongly associated and b. weakly associated pairs ranked by co-occurrence weighting. c. Highlights pairs that demonstrate intuitive validation of the weighted ranking.

a. Strongest attribute associations

Rank	Attribute Name	Second Attribute Name	$W_{(a1,a2)}$
1	package	synonym	6.94×10^{-4}
2	model	synonym	6.94×10^{-4}
3	colection date	geographic location	4.45×10^{-4}
4	geographic location	package	4.43×10^{-4}
5	geographic location	model	4.43×10^{-4}

b. Weakest attribute associations

Rank	Attribute Name	Second Attribute Name	$W_{(a1,a2)}$
710293	Sample source name	synonym	-2.54×10^{-4}
710294	Sample_source_name	model	-2.87×10^{-4}
710295	description	model	-4.06×10^{-4}
710296	description	package	-4.07×10^{-4}
710297	Sample_source_name	synonym	-4.17×10^{-4}

c. Highlighted pairs

Rank	Attribute Name	Second Attribute Name	$W_{(a1,a2)}$
107	depth	elevation	8.72×10^{-5}
251	serovar	strain	5.90×10^{-5}
363	disease state	individual	4.84×10^{-5}
710117	organism part	serovar	-2.38×10^{-5}
710223	environment biome	organism part	-5.24×10^{-5}