Supplementary materials for *MetaSRA: normalized human sample-specific metadata for the Sequence Read Archive*

April 2017

Contents

1 Biologically significant ontology terms 1

2 Preprocessing of the Experimental Factor Ontology 2

3 Detailed description of the ontology mapping pipeline 2

4 Configurations for running existing methods 11
   4.1 SORTA .............................................. 11
   4.2 ZOOMA .............................................. 11
   4.3 BioPortal Annotator ................................. 12

5 Performance on most commonly mapped terms 12

6 Defining sample-type categories 13

7 Detailed description of sample-type prediction procedure 13
   7.1 Feature selection ...................................... 13
   7.2 Prediction procedure .................................. 14

8 Evaluation of sample-type prediction on the training set 15

1 Biologically significant ontology terms

We only map samples to “biologically significant terms.” A term is biologically significant if the presence of the term offers information regarding the biochemical processes occurring in the living sample. For example, disease terms are biologically significant. Similarly, terms for ethnic backgrounds such as the Experimental Factor Ontology (EFO) terms for “Caucasian” and “African American” are biologically significant due to the fact that certain genotypes may be more or less common in different ethnic populations.
We manually searched the ontologies for biologically significant terms that are near the roots of the ontologies’ directed acyclic graphs. We then assume that all children of a biologically significant term are also biologically significant and retrieve all children of the manually selected nodes.

2 Preprocessing of the Experimental Factor Ontology

There are certain idiosyncrasies unique to the EFO that caused errors in the ontology mapping process. To address these issues, we modified the EFO during a preprocessing procedure. This procedure involves the following steps:

1. Add cell line synonyms to the EFO: We use the Cellosaurus to add synonyms to cell line terms in the EFO. More specifically, each entry in the Cellosaurus includes a set of links to external references of that cell line. Such references for a given cell line may include the EFO’s definition of that cell line. We add to the EFO’s definitions all of the synonyms that appear in the Cellosaurus’s entry for the respective cell line. For example, the EFO’s term for the MCF-7 cell line does not include the synonym “MCF.7”; however, this is included as a synonym for the Cellosaurus’s entry. Thus, we add the synonym “MCF.7” to the EFO’s entry for MCF-7.

2. Remove incorrect synonyms from the EFO: Many of the EFO’s cancer-type terms are associated with synonyms that represent a more general concept than the term. Moreover, these synonyms are usually labelled as “exact synonyms” rather than “broad synonyms.” For example, the term “hepatocellular carcinoma” has the exact synonym “Liver Cancer.” Given this synonym, a metadata entry that includes the substring “Liver Cancer” will map to the term “hepatocellular carcinoma.” We assert that this is an incorrect mapping. To avoid these erroneous mappings, we manually removed synonyms from the EFO’s cancer-terms that represent a more general concept than the term.

3. Convert EFO synonyms to lower-case: Many of the synonyms in the EFO have a first character that is upper-case. For example, the term “breast carcinoma” has the synonym “Carcinoma of breast.” For these synonyms, we convert the first character to lower-case. For example, the synonym “Carcinoma of breast” is converted to “carcinoma of breast.”

3 Detailed description of the ontology mapping pipeline

The version of the pipeline used to generate the results in this paper consists of the stages listed below. We note that the design of our software is modular and
allows for easy implementation of new stages. In the future, we plan to develop and refine the pipeline to further increase the accuracy of the mapped ontology terms.

1. **Filtering key-value pairs:** We created a blacklist of keys and a blacklist of values such that if a key appears in the blacklist of keys or a value appears in the blacklist of values, the key-value pair will be removed from the ontology mapping process.

   We remove all keys that describe a property that likely does not describe the sample. For example, we remove key-value pairs with keys “biomaterial provider”, “study name”, and “submitter handle.” The blacklist of values include values that negate the key. Such values include “normal”, “unknown”, “none”, and “no.”

2. **Initializing the Text Reasoning Graph (TRG):** The initial TRG consists of a set of nodes that represent the raw set of key-value pairs describing the sample. First, a node is created for each key-value pair. From each of these “start nodes”, we draw two edges to two artifact nodes – one artifact representing the key and the other artifact representing the value. Figure 1 depicts the initial TRG for the following set of key value pairs:

   cell line: Parkin-expressing MRC5 fibroblasts
   cell state: Proliferating
   source name: Prolif

3. **Generating n-grams:** From each artifact node, we generate all n-grams for $n = 1, \ldots, 8$. We use the Python Natural Language Toolkit (nltk) to tokenize the text before constructing n-grams. For each n-gram generated from an artifact, we draw an edge from the original artifact to the derived artifact. Figure 2A illustrates an artifact node from the graph of Figure 1 with derived artifacts representing n-grams.

4. **Lowercase:** From each artifact node that represent artifacts with uppercase characters, we draw an edge to a new artifact node that has all lowercase characters. Figure 2B demonstrates this process.

5. **Delimiters:** The NLTK’s tokenizer does not split on the characters “+”, “-”, “/”, and “.”. We therefore, split all artifact strings by these delimiters as shown in Figure 2C.

6. **Inflectional variants:** We derive the inflectional variants of all artifacts by consulting the SPECIALIST Lexicon. This is demonstrated in Figure 2D.

7. **Spelling variants:** We derive the spelling variants of all artifacts by consulting the SPECIALIST Lexicon. This is demonstrated in Figure 2E.
Figure 1: The initial TRG created from a set of key-value pairs describing a sample.
Figure 2: (A) An artifact node with derived n-grams. (B) An artifact with a derived lowercase artifact. (C) Delimiting artifacts on special characters. (D) Deriving inflectional variants. (E) Deriving spelling variants. (F) Deriving custom synonyms. (G) Expanding acronyms.
8. **Manually annotated synonyms:** There are certain words that are very common in the metadata, but that are not included in the ontologies. For example, the word “tumor” is extremely common in the metadata, but is not present in the Disease Ontology. We filled such gaps by creating a small, custom thesaurus. In our thesaurus, “tumor” is given the synonym “neoplasm.” The word “neoplasm” is a term in the Disease Ontology that is semantically equivalent to “tumor.” This process is demonstrated in Figure 2F.

9. **Custom acronym expansion:** There are certain acronyms that are common in the metadata, but are not included in the ontologies. For example, the acronym “hESC” is very common in the metadata, but is not present in the Cell Ontology. We filled such gaps by expanding common acronyms. For example, we expand “hESC” to “human embryonic stem cell.” This process is demonstrated in Figure 2G.

10. **Exact string matching:** We perform a preliminary mapping step in which we map artifacts to ontology terms by searching for exact matches between the artifact strings and term names and synonyms in the ontologies. This preliminary mapping stage is performed quickly using a trie data structure.

11. **Context-specific synonyms:** We create a list of “context-specific synonyms” and derive synonyms for artifacts when that artifact was derived from a value that is associated with a specific key. For example, a common key-value pair is sex: F. Here, the string “F” is an abbreviation for “female”; however, this is only known because the key maps to the EFO term for “sex.” “F” in another context may not be an abbreviation for “female.” This process is illustrated in Figure 3.

12. **Fuzzy string matching** We perform fuzzy string matching between the artifacts and the ontology terms. Let $a$ and $b$ be two strings and let $d(a, b)$ be their Levenshtein edit distance. Let $l(x)$ be the length of a string $x$. An artifact $a$ matches with an ontology term name or synonym $b$ if the following conditions hold: $l(a) > 2$, $d(a, b) \leq 2$, and $d(a, b) \leq \max\{l(a), l(b)\}/10$. We do not match artifacts that are less than 3 characters long in order to avoid false positive mappings. If the artifact is greater than 2 characters long, a match is called if the edit distance is less than or equal to 2 and less than or equal to 0.1 times the length of the longer string. For example, the misspelled artifact “forskin fibroblast” would match with the ontology term name “foreskin fibroblast.” In contrast, the string “year” would not match with the ontology term “ear” because the edit distance of 1 is greater than 0.1 of the length of the longer string. When an artifact matches an ontology term, we create a node representing the ontology term and draw an edge from the artifact to the new ontology term node.

To more efficiently perform fuzzy string matching we store all ontology term names and synonyms in a Burkhard-Keller metric tree with the
bag-distance metric defined in [2]. When performing fuzzy string matching between a query string $s$ and the strings in the metric tree, we retrieve all strings in the metric tree that are within a distance of 2 from $s$ using bag-distance. Since bag-distance is a lower-bound on edit-distance, this process filters out all strings in the ontologies whose lower bound on the edit distance is greater than the threshold of 2 that we impose on fuzzy string matching. We then explicitly compute edit distance between $s$ and the retrieved strings.

13. **Matching to custom terms:** There are several noun-phrases that are common in the metadata and that are superstrings of ontology terms, but that do not imply that the sample maps to the contained ontology term. For example, the phrase “blood type” does not imply that the sample was derived from blood. Similarly, “tissue bank” describes the organization that provided the sample, but does not necessarily imply that the sample is a tissue sample. To differentiate the larger noun-phrase from the ontology term it contains, we maintain a custom list of misleading noun-phrases and remove ontology term mappings if those mappings were derived from a substring of a misleading noun-phrase. For example, given the string “blood type”, the artifact “blood” will be blocked from mapping to the ontology term for blood because it is a substring of the noun-phrase “blood type.” Currently, we have 27 noun-phrases in our index and we will continue to build this index as we find more misleading noun-phrases that contain ontology terms.

14. **Remove extraneous cell-line matches:** Many cell lines have short
names that oftentimes resemble acronyms or gene names. For example, “SRF” is a gene as well as a cell line in the Cellosaurus. Similarly, “MDS” is often used as an acronym for Myelodysplastic Syndromes and also happens to be the name of a cell line in the Cellosaurus.

We remove extraneous mappings to cell line terms by searching the graph emanating from the key for a lexical match to ontology terms such as those for “cell line” and “cell type”. If such a match is not found, we search the graph emanating from the value for artifacts that have a lexical match to a cell line ontology term and remove all such ontology term nodes. This process is illustrated in Figure 4.

Figure 4: (A) The term “cell line” was found in the graph emanating from the key. Thus, we keep the cell line term for “MRC5” in the graph emanating from the value. (B) No ontology term for “cell line” or “cell type” was found in the graph emanating from the key. We therefore remove the cell line term for “ASO” in the graph emanating from the value.

15. Map to linked-superterms: The domain covered by the EFO overlaps with many of the other ontologies because it includes cell types, anatomi-
cal entities, diseases, and cell lines. In many cases, the EFO is inconsistent with other ontologies in how it draws edges between terms. For example, the term “lung adenocarcinoma” and “adenocarcinoma” are present in both the Disease Ontology and the EFO; however “adenocarcinoma” is a parent of “lung adenocarcinoma” only in the Disease Ontology and not in the EFO. These inconsistencies pose a problem when we filter for the maximal phrase-length in the metadata. For example, when a sample maps to “lung adenocarcinoma” and “adenocarcinoma”, we remove “adenocarcinoma” because it is a substring of “lung adenocarcinoma”. This is valid for the Disease Ontology because the term for “adenocarcinoma” is implied by “lung adenocarcinoma” by its position in the ontology. However, this results in a false negative for the EFO version of this term.

To counteract this problem, we link the terms in the EFO to terms in the other ontologies. Two terms are linked when they share the same term-name or exact-synonym. Then, when an artifact maps to a term, we traverse the term’s ancestors and map to any terms that are linked to those ancestors. In the case of “lung adenocarcinoma”, we would traverse the ancestors of this term in the Disease Ontology and map to the EFO’s “adenocarcinoma” because it is linked to the Disease Ontology version of this term. Figure 5 illustrates this process.

Figure 5: An example of linked terms between the Disease Ontology and the EFO. If a sample maps to “lung adenocarcinoma” in the Disease Ontology, we follow all ancestors and also map to linked terms of those ancestors. In this case, the EFO’s “adenocarcinoma” term will also be mapped.

16. **Cell line disease implications:** The EFO is missing edges between disease cell line terms and the corresponding disease terms. For example,
the term “cancer cell line” does not have an edge to “cancer.” To fill this gap, if a sample maps to a cell line category term, we also map to the corresponding disease terms.

17. **Block superterm mapping:** It is a common occurrence for disease ontology terms to include anatomical entities in their name. For example, “breast cancer” includes “breast” as a substring. It would be incorrect to map “breast” to the sample because this word localizes the cancer, but does not localize the origin of the sample. We note that it is possible that the sample was indeed derived from breast tissue; however, it is also possible the sample originated from other tissue such as a malignant site. We maintain a conservative approach and avoid mapping to “breast.” We implement this process by designing each artifact node to keep track of the original character indices in the metadata from which it was derived. After mapping all artifacts to the ontologies, we remove all ontology terms that were lexically matched with an artifact node that is subsumed by another artifact node that matches with another ontology term.

18. **Custom consequent mappings:** We maintain a small list of 6 common terms that imply other terms. For example, if a cell maps to a the EFO term for “cell line”, we consequently map the sample to the Cell Ontology’s term for “cultured cell.”

19. **Real-value property extraction:** We maintain a list of ontology terms that define real-value properties. Currently, we use 6 terms: “age”, “passage number”, “timepoint”, “age at diagnosis”, “body mass index”, and “age at death.” Future work will entail expanding this list. To extract a real-value property from a key-value pair, we search the graph emanating from the key for a match to a property ontology term. If such a property is found, we search the graph emanating from the value for an artifact representing a numerical value and a unit ontology term node (e.g., “46” and “year”). From this process, we extract the triple (property, value, unit). For example, given the key-value pair age: 46 years old, we extract (“age”, 46, “year”).

20. **Filtering mapped ontology terms by semantic similarity:** The ontologies are structured so that each synonym of an ontology term is given a synonym-type. These types include “exact”, “broad”, and “narrow.” These synonym-types describe the relationship between the synonym string and the term name. An “exact” synonym indicates that the string is semantically closer to the ontology term name than a “broad” synonym. If an artifact matches to multiple ontology terms, the ontology term with the semantically nearest matched target is likely to be the best match with the artifact. Thus, given an artifact with multiple matched terms, we examine the targets within the matched terms to which the artifact matched and rank these targets according to the semantic similarity with the ontology term name. We then keep the match with the highest similarity and discard the rest.
For example, given the artifact “skin”, we find several terms in the Uberon ontology that have a synonym “skin”: “zone of skin” (exact synonym), “skin epidermis” (broad synonym), “skin of body” (related synonym), and “integument” (related synonym). Of these terms, “skin” is semantically most similar to the term “zone of skin” because it is an exact synonym of this term. We therefore keep this mapping and discard the rest.

21. **Consequent cell line mappings:** Our pipeline draws edges between cell line ontology term nodes and the ontology terms that describe the cell line. For example, if the TRG contains the node for the cell line “HeLa”, we draw an edge to the ontology terms for “adenocarcinoma” and “female” because this cell line was derived from a woman with cervical adenocarcinoma. We consider such mappings to be consequent mappings because they are retrieved using an external knowledge base. This knowledge base was created from data we scraped from the ATCC website at [https://www.atcc.org](https://www.atcc.org). To construct mappings between cell lines and ontology terms, we ran a variant of our pipeline on the scraped cell line data. We scraped cell line metadata for all cell lines that are present in the Cellosaurus.

22. **Consequent developmental stage mappings:** If the sample maps to a real-value property with property “age” and unit “year”, we check whether the value is greater than 18. If so, we consequently map the sample to the EFO and Uberon terms for “adult.”

4 Configurations for running existing methods

4.1 SORTA

We ran SORTA through its web interface ([https://molgenis19.gcc.rug.nl/](https://molgenis19.gcc.rug.nl/)). The SORTA tool accepts as input a set of strings. It then attempts to find a match between each string and a term in the target ontology. To run SORTA on the key-value SRA metadata, we let each key and each value be a separate string for SORTA’s input. Since SORTA does not allow any string to be longer than 255 characters, we truncated any input string longer than 255 characters to meet this threshold.

4.2 ZOOMA

We ran ZOOMA using its REST API ([https://www.ebi.ac.uk/spot/zooma/docs/api.html](https://www.ebi.ac.uk/spot/zooma/docs/api.html)). More specifically, we used ZOOMA’s /services/annotate API. We queried ZOOMA on each key and each value separately by supplying these strings to the `propertyValue` argument.
Another approach to running ZOOMA on key-value data is to supply the `propertyValue` argument with the value and to supply the `propertyType` argument with the key. However, we found that ZOOMA will not map the key to the ontologies when used in this manner. Thus, ZOOMA will miss mapping key value pairs like “disease : yes” to the ontology term for “disease.” For this reason, we queried ZOOMA on keys and values separately.

### 4.3 BioPortal Annotator

We ran BioPortal’s Annotator through its REST API ([http://data.bioontology.org/documentation](http://data.bioontology.org/documentation)). More specifically, we used the Annotator’s `/annotate` API. We ran the tool on keys and values separately by supplying these strings to the `text` argument. We further set the `longest_only` argument to “true.” This tells the tool to only retrieve the longest matched phrase in the target ontology.

### 5 Performance on most commonly mapped terms

We evaluated performance on the top 10 most commonly mapped terms. For this analysis, we count a term as being mapped from a sample if that term is a most-specifically-mapped term for that sample. That is, no children of the term were mapped from the sample.

For each term in the top 10 most commonly mapped terms, we sampled 100 samples at random from the entire set of samples in the MetaSRA. For this analysis, we allowed multiple samples from the same study. We then evaluate the precision for the target term over these 100 samples. The results are displayed in Table 1. These statistics provide an unbiased estimate of the precision for each term over the entire set of samples in the MetaSRA.

| Term          | Name            | Precision |
|---------------|-----------------|-----------|
| CL:0000010    | cultured cell   | 0.98      |
| UBERON:0003100| female organism | 1.00      |
| UBERON:0003101| male organism   | 1.00      |
| UBERON:000955 | brain           | 0.99      |
| EFO:0000727   | treatment       | 0.90      |
| UBERON:000178 | blood           | 0.86      |
| EFO:0003156   | Caucasian       | 1.00      |
| EFO:0000322   | cell line       | 1.00      |
| EFO:0001272   | adult           | 1.00      |
| UBERON:0007023| adult organism  | 1.00      |

Table 1: Precision of the most commonly mapped ontology terms.
6 Defining sample-type categories

Figure 6 illustrates how we define each sample-type category based on the methods by which the sample was obtained. We note that we call an isolated cell sample a “stem cell” if the targeted cell type has the ability to differentiate. Thus, the “stem cell” category includes any cell type that is pluripotent, multipotent, or oligopotent.

Figure 6: A graph illustrating how sample-type categories are defined. Each node in the graph represents a biological sample. Arrows between nodes represent procedures carried out on the sample. Nodes are colored according to their sample-type category.

7 Detailed description of sample-type prediction procedure

7.1 Feature selection

We consider two types of features for representing each sample: \( n \)-gram features and ontology term features. For \( n \)-gram features, we consider all uni-grams and bi-grams appearing in the training samples’ raw metadata. For ontology term features, we consider the set of all ontology terms that were mapped to the training samples by our automated pipeline.

Let \( S \) be this aforementioned set of all \( n \)-gram features and ontology term features. We initially prune \( S \) by removing all stop word \( n \)-grams as well as
all features that appear in fewer than 2 samples. From the remaining features in \( S \), we filter features using the mutual information between the feature and the class labels. Let \( y_1, y_2, \ldots, y_n \) be the sample-types of the training samples. Furthermore, let \( Y \) be the set of possible sample-types. Our feature selection process starts with an empty set of features \( F := \emptyset \) and iteratively adds features from \( S \) to \( F \) according to the following strategy: for each sample-type \( j \in Y \), we convert the training labels into binary labels \( y_{1j}, y_{2j}, \ldots, y_{nj} \) such that \( y_{ij} := 1 \) if \( y_i = j \) and \( y_{ij} := 0 \) otherwise. We then computed the mutual information between each feature \( f \in S \) and the binarized labels. We then add to \( F \) the 75 top scoring features in \( S \). When this process is complete, each feature \( f \in F \) should be indicative of at least one of the target sample-types.

7.2 Prediction procedure

Although we train a one-vs-rest classifier using logistic regression binary classifiers, we ultimately use a custom decision procedure for making a sample-type prediction. This procedure entails limiting the possible predicted sample-types based on ontology terms that were mapped by our computational pipeline. The algorithm chooses among the remaining possible sample-types by selecting the sample-type with highest confidence according to the one-vs-rest classifier.

To provide an example, if the ontology term “stem cell” was mapped to the sample, we set \( p(y = j|x) = 0 \) for \( j \in \{ \text{tissue, cell line, primary cells} \} \). We then compute the probabilities \( p_i := \frac{p(y = i|x)}{\sum_k p(y = k|x)} \) for each \( i \). Our final prediction is then \( \hat{y} = \arg\max_i p_i \). In summary, this process asserts that if “stem cell” mapped to the sample, then the sample must be either a stem cell sample, in vitro differentiated cell sample, or induced pluripotent stem cell sample. We let the classifier decide which is the most likely label among these possible labels.

More specifically, we follow the following steps for making a prediction:

1. If the sample maps to “xenograft” (EFO:0003942), then we predict tissue with confidence 1.0.
2. If the sample was passaged (i.e. maps to a real-value property tuple with property “passage number” and unit “count”), then we assert the sample cannot be tissue. If the number of passages is greater than 0, then we assert the sample cannot be primary cell.
3. If the sample maps to a cell line, then we check the Cellosaurus for the cell line category. We map the Cellosaurus cell-line category to a set of possible sample types as follows:
   - Induced_pluripotent_stem_cell: in vitro differentiated cells, induced pluripotent stem cell line
   - Cancer_cell_line: cell line
   - Transformed_cell_line: cell line

14
• Finite_cell_line: cell line
• Spontaneously_cell_line: cell line
• Embryonic_stem_cell: stem cells, in vitro differentiated cells
• Telomerase_cell_line: cell line
• Conditionally_cell_line: cell line
• Hybridoma: cell line

4. If the sample maps to the term “stem cell”, then we remove the possibility that the sample type is cell line, tissue, or primary cells.

5. If the sample maps to a specific cell-type term (i.e. any term that is a child of “somatic cell”), then we remove the possibility that the sample-type is tissue. This follows from our observation that when the metadata describes a specific cell-type, the sample consists of homogenous cells that have been isolated and filtered. The sample no longer consists of cells positioned in their original three dimensional structure and is thus not a tissue sample.

8 Evaluation of sample-type prediction on the training set

We note that the test data set was small compared to the training data set. To provide an estimate of the performance of the classifier on a larger data set, we ran the algorithm using leave-one-out cross validation on the training set. The algorithm achieved 0.845 accuracy on this data set. Figure 7A shows the row-normalized confusion matrix and Figure 7B shows the calibration of the model.

References

[1] W.A. Burkhard and R.M. Keller. Some approaches to best-match file searching. Communications of the ACM, 16(4):230–236, April 1973.

[2] I. Bartolini, P. Ciaccia, and M. Patella. String matching with metric trees using an approximate distance. In Proceedings of the 9th International Symposium on String Processing and Information Retrieval, SPIRE 2002, pages 271–283, London, UK, UK, September 2002. Springer-Verlag.
Figure 7: (A) The confusion matrix of the algorithm on the training set evaluated using leave-one-out cross validation. The bar graph above the matrix displays the distribution of classes within the training set. (B) Plotting the calibration of the classifier.