Appendix E. Management Operations

Retrieve resources, add resources, delete resources and merge spaces are four basic management operations of the resource space.

E.1 Establish Relations Between Resource Space and its Resources

Assume that the set of resources for generating its resource space is stored in a directory on a file system without losing generality. A resource is retrieved by its name and the path of the directory, denoted as *path*/*name*. The resources belonging to a class n (i.e., $\{path_1/name_1, \ldots, path_k/name_k\} \in n$) is obtained by the function $B(d, n)$. The efficiency of determining the classes of the set of resources is $O(1)$.

E.2 Retrieve Operations

The retrieval operation enables users or application systems to use the following SQL-like language to accurately obtain the interested resources from a resource space $RS = \{X_1, X_2, ..., X_N\}$ where $X_i = \langle V_i, SR_i \rangle$ represents a class tree and $i \in [1, N]$. For example, the 2-dimensional resource space extracted from the GitHub dataset (1000 README text files) is represented as $RS = \langle network, model \rangle$, where the *network* is a class tree consisting of 1428 classes and 1885 subclass relations between the classes <*network*, *neural network*, *attention network*, *deep network*, ..., (*network*, *neural network*), (*network*, *attention network*), (*network*, *deep network*), ...>, and the *model* is a class tree consisting of 1431 classes and 1722 subclass relations between the classes <*model*, *classification model*, *ImageNet model*, *language model*, ..., (*model*, *classification model*), (*model*, *ImageNet model*), (*model*, *language model*), ...>.

1. SELECT RESOURCE FROM *RS* WHERE $X_i = m$. It searches the dimension $X_i = \langle V_i, SR_i \rangle$ in a resource space $RS = \{X_1, X_2, ..., X_N\}$ according to the given representation m (noun or noun phrase) of the expected class (i.e., find class n of the dimension according to the given m) and then returns $R(n)$, where $n \in V_i$, $n = m$. For example, the statement for retrieving projects from $RS = \langle network, model \rangle$ is represented as SELECT RESOURCE FROM *RS* WHERE *network*="neural network". It returns 400 projects from 1000 projects of the resource space, each of which has a README file containing the noun phrase "neural network". As a project can be accessed from a class and all its super-classes, the number of classes is larger than the number of projects. As the search is on the class tree of a dimension, so it can quickly reduce the search space.
2. SELECT RESOURCE FROM *RS* WHERE $X_i = (m_1, m_2, ..., m_P)$. It searches the dimension $X_i = \langle V_i, SR_i \rangle$ in a resource space $RS = \{X_1, X_2, ..., X_N\}$ according to the given representations $m_1, m_2, ..., m_P$ of the expected classes (i.e., find the classes $n_1, n_2, ..., n_P$ of the dimension according to the given $m_1, m_2, ..., m_P$) and then returns the set of texts accessed by these classes, i.e., $\cup_{j=1}^{P} R(n_j)$, where $n_j \in V_i$, $n_j = m_j$ and $j \in [1, P]$. For example, the following statement is for retrieving texts from $RS = \langle network, model \rangle$: SELECT RESOURCE FROM *RS* WHERE *network*=("recurrent neural network", "adversarial network"). It returns 106 projects consisting of the class "recurrent neural network" that contains 58 projects and "adversarial network" that contains 48 projects. As the RS is a 2-NF resource space, a project can only belong to one classes of a dimension. This statement supports users or application systems to retrieve resources from multiple classes of a dimension at one time.
3. SELECT RESOURCE FROM *RS* WHERE $X_1 = m_1$ AND $X_2 = m_2$ AND ... AND $X_N = m_N$. It searches dimensions $X_1 = \langle V_1, SR_1 \rangle$, $X_2 = \langle V_2, SR_2 \rangle$, ..., and $X_N = \langle V_N, SR_N \rangle$ in a resource space $RS = \{X_1, X_2, ..., X_N\}$ according to a set of given representations $m_1, m_2, ..., m_N$ of the expected classes (i.e., find the classes $n_1, n_2, ..., n_N$ of the multiple dimensions according to the given $m_1, m_2, ..., m_N$ and n_1 is a class of X_1, n_2 is a class of X_2, ..., n_N is a class of X_N) and then returns $\cap_{i=1}^{N} R(n_i)$, where $n_i \in V_i$, $n_i = m_i$ and $i \in [1, N]$. For example, the statement for retrieving the $RS = \langle network, model \rangle$ is represented as: SELECT RESOURCE FROM *RS* WHERE *network*="neural network" AND *model* = "segmentation model", which returns 4 common projects of the class "neural network" of *network* dimension that contains 400 projects and the class "segmentation model" of the *model* dimension that contains 14 projects.
4. SHOW COORDINATE OF d ON RS AT $\{X_1, X_2, ..., X_N\}$. It shows the coordinate $(p[X_1], p[X_2], ..., p[X_N])$ of a resource d (the representation of d includes the name and the path of the directory of the resource, i.e., *path*/*name*) in the resource set D accessed by resource space $RS = \{X_1, X_2, ..., X_N\}$, i.e., $d \in D = \cup_{i=1}^{N} R(n_i) = \{d_1, d_2, ..., d_{|D|}\}$, $R(X_i)$ is the set of resources that can be accessed from the dimension X_i. The $p[X_i]$ is a path started from the root of X_i (denoted as $<n_0, n_1, ..., n_{|p[X_i]|}>$) representing the projection of a resource d on X_i, and $d \in R(n)$ if and only if $n \in p[X_i]$. For the $RS = \langle network, model \rangle$ extracted from the 1000 GitHub README texts $D = \{d_1, d_2, ..., d_{1000}\}$, the statement for showing the coordinate of a text such as $d_{423} \in D$ in $RS = \langle network, model \rangle$ is represented as: SHOW COORDINATE OF d_{423} ON RS AT $\{network, model\}$ (d_{423} is the *path*/*name* of the README text file of the project "Scene Representation Networks" at https://github.com/vsitzmann/scene-representation-networks). It returns the coordinate of d_{423} in RS: (<"network", "scene representation network">, <"model", "core model", "core SRNs model">) as the paths framed by the red dotted line shown in Figure 1. It shows that d_{423} can be accessed from both the *network* dimension and the *model* dimension. This statement supports users or application systems to easily obtain the coordinate of a resource in the resource space to clearly shows the classes and subclass relations on multiple dimensions for retrieving resources and analyzing resources efficiently.

As the resources in the resource space are classified with the

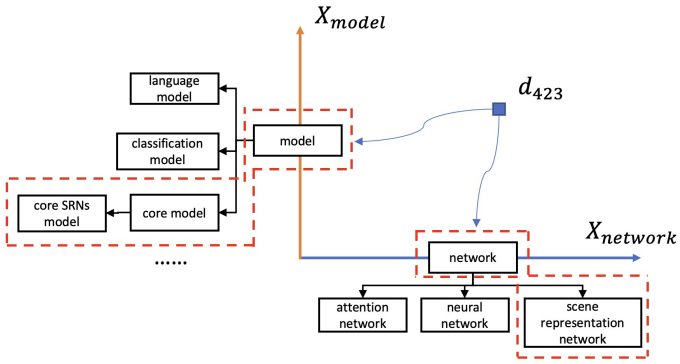

Figure 1: Projections of resource d_{423} onto two dimensions.

process of extracting class trees, retrieving a resource of a class carries out by searching the class on the class trees with an average time complexity of $O(logn)$ and then getting the resource with a time complexity of $O(1)$.

E.3 Add Resources

The add operation enables users or application systems to use the following SQL-like language to add a new text resource d (d is the *path*/*name* of the resource) to a resource space $RS = \{X_1, X_2, ..., X_N\}$, represented as: INSERT d INTO RS.

For the new text d and the resource space RS, and the add operation proceeds as follows, for each $X_i = \langle V_i, SR_i \rangle$ and d:

1. Extract the class tree with the root of X_i from the content of d, denoted as $T' = \langle V', SR' \rangle$. $T' = \langle \emptyset, \emptyset \rangle$ if the root node of X_i doesn't appear in the content of d.
2. For each class n in V' but not in V_i, add it to V_i (i.e., $V_i = V_i \cup \{n\}$) and add the corresponding subclass relations to SR_i. If there exists a subclass relation (n_i, n_j) in V_i so that n is the subclass of n_i and n_j is the subclass of n, then delete the subclass relation (n_i, n_j) from SR_i, add the subclass relations (n_i, n) and (n, n_j) to SR_i. Otherwise, for each n' in V_i, if n is the subclass of n', then add the subclass relation (n', n) to SR_i.
3. For each class n in V_i, calculate the $B(d, n)$, get the projection of the resource d on X_i, denoted as $p[X_i] = \langle n_0, n_1, ..., n_{|p[X_i]|} \rangle$, where $n_{i+1} = \arg\max_{n, n \in GetSubclass(n_i)} B(d, n)$, $i \in [0, |p[X_i]| - 1]$ and $GetSubclass(n_i)$ returns the set of subclasses of the class n_i.
4. For each class n in $p[X_i]$, add d to the set of the texts represented by class n, i.e., $R(n) = R(n) \cup \{d\}$.

For example, the statement for adding a resource to a 2-dimensional resource space $RS = \langle network, model \rangle$ is represented as INSERT d_{1001} INTO RS (d_{1001} is the *path*/*name* of the README text file of the project "Cedgan Interpolation" at https://github.com/dizhu-gis/cedgan-interpolation), where d_{1001} is a text that has not been managed by RS. A part of the contents of d_{1001} is shown in Figure 2 (a). A part structure of $RS = \langle network, model \rangle$ is shown in Figure 2 (b). Figure 2 (c) shows the abstraction trees extracted from d_{1001} for both the *network* and the *model* dimensions of RS. The RS after the add operation is shown in Figure 2 (d) (i.e., a new class "conditional generative adversarial neural network" is added to the *network* dimension, and no new class is added to *model* dimension). The part framed by the red dotted line is the projection of d_{1001} on two dimensions.

The add operation can automatically add new classes and subclass relations according to the added text and extract the classes to which the added resource belongs on each dimension and their abstractions.

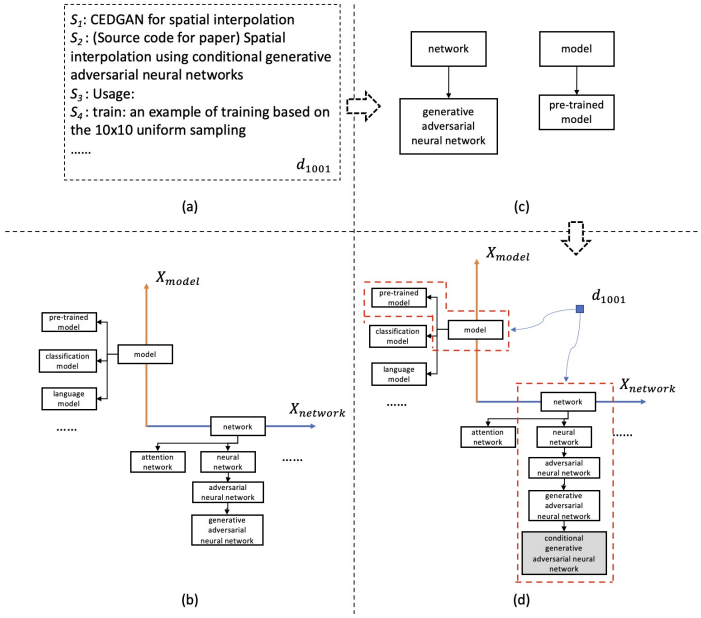

Figure 2: Adding resource d_{1001} to RS.

E.4 Delete Resources

The delete operation enables users or application systems to use the following SQL-like language to delete a resource d (the *path*/*name* of the resource) from the resource space $RS = \{X_1, X_2, ..., X_N\}$, represented as DELETE d FROM RS.

For a text d (including *path*/*name*) and the resource space $RS = \{X_1, X_2, ..., X_N\}$, the operation for deleting d takes the following steps:

1. Show the coordinate $(p[X_1], p[X_2], ..., p[X_N])$ of resource d in RS, where $p[X_i]$ is a path started from the root of X_i (denoted as $<n_0, n_1, ..., n_{|p[X_i]|}>$).
2. For each dimension X_i and each class n in the $p[X_i]$, delete d from $R(n)$, i.e., $R(n) = R(n) - \{d\}$.
3. Delete d from the file system.

For example, the statement for deleting a resource from the 2-dimensional resource space $RS = \langle network, model \rangle$ is represented as DELETE d_{467} FROM RS (the project "FFN" at https://github.com/google/ffn). It deletes a text d_{467} from the $RS = \langle network, model \rangle$, the process is depicted in Figure 3 .

Users or application systems can also use the following statement to delete the resources from the resource space $RS = \{X_1, X_2, ..., X_N\}$ by specifying the classes of the dimensions.

1. DELETE RESOURCE FROM RS WHERE $X_i = m$. It deletes the resources accessed by the expected class n of the given representation m from the dimension $X_i = \langle V_i, SR_i \rangle$ of the RS, i.e., deletes the resources in $R(n)$ where $n \in V_i$, $n = m$.
2. DELETE RESOURCE FROM RS WHERE $X_i = (m_1, m_2, ..., m_P)$. It deletes the resources in classes $n_1, n_2, ..., n_P$ according to the given representations $m_1, m_2, ..., m_P$ from the dimension $X_i = \langle V_i, SR_i \rangle$ of the RS, i.e., delete the resources in $\cup_{j=1}^{P} R(n_j)$, where $n_j \in V_i$, $n_j = m_j$ and $j \in [1, P]$.
3. DELETE RESOURCE FROM RS WHERE $X_1 = m_1$ AND $X_2 = m_2$ AND ... AND $X_N = m_N$. It deletes the resources accessed by the expected classes $n_1, n_2, ..., n_N$ according to the given representations $m_1, m_2, ..., m_N$ from the dimensions $X_1 = \langle V_1, SR_1 \rangle$, ..., and $X_N = \langle V_N, SR_N \rangle$ of the RS, i.e., delete the resources in $\cap_{i=1}^{N} R(n_i)$, where $n_i \in V_i$, $n_i = m_i$ and $i \in [1, N]$.

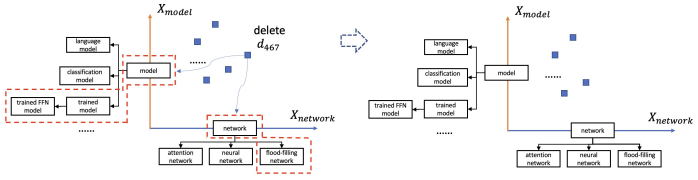

Figure 3: Delete resource d_{467} from RS.

E.5 Merge Space

The merge operation enables users or application systems to use the following statement to merge two spaces $RS = \{X_1, X_2, ..., X_N\}$ and $RS' = \{X'_1, X'_2, ..., X'_N\}$by merging their dimensions representing the same class (i.e., with the same root nodes), where $RS^* = \{X_1 \cup X'_1, X_2 \cup X'_2, ..., X_N \cup X'_N\}$, X_i and X'_i represent the same class, and $X_i \cup X'_i$ represents the merge of the dimensions, $i \in [1, N]$: MERGE RS AND RS' INTO RS^*.

Two dimensions $X_i = \langle V_i, SR_i \rangle$ and $X'_i = \langle V'_i, SR'_i \rangle$ of two resource spaces are merged into $X^*_i = \langle V^*_i, SR^*_i \rangle$ by two steps: (1) Merge node sets (i.e., $V_i \cup V'_i$) by combining the same nodes (i.e., nouns/noun phrases), denoted as V^*_i; and (2) Merge subclass relations (i.e., $SR_i \cup SR'_i$) by removing the redundant subclass relations and duplicating the child tree with two or more parent nodes to construct trees, denoted as SR^*_i.

To verify the merge operation in the GitHub application, the README dataset D (1000 texts) is split into two sub-datasets D' (468 texts) and D'' (532 texts). Two dimensions (*network* and *model*) can be extracted from D' and D'' respectively, denoted as $RS' = \langle network', model' \rangle$ and $RS'' = \langle network'', model'' \rangle$, where the $network'$ is a class tree consisting of 821 classes and 1088 subclass relations between the classes, the $model'$ is a class tree consisting of 875 classes and 1080 subclass relations, the $network''$ is a class tree consisting of 712 classes and 922 subclass relations, and the $model''$ is a class tree consisting of 670 classes and 822 subclass relations.

The statements for merging the two spaces RS' and RS'' can be represented as MERGE RS' AND RS'' INTO RS^*= $\{network^*, model^*\}$, where $network^*$ is the class tree consisting of 1428 classes and 1885 subclass relations obtained by merging dimensions $network'$ and $network''$, and $model^*$ is the class tree consisting of 1431 classes and 1722 subclass relations obtained by merging dimension $model'$ and dimension $model''$.

The number of nodes and subclass relations before and after the merge operations are shown in Table 1. We can see that *network* and $network^*$ have the same 1428 classes and 1885 subclass relations, and *model* and $network^*$ have the same 1431 classes and 1722 subclass relations. This verifies the correctness of the merge operation. The dimensions in the resource spaces discovered by different datasets may have duplicate classes and subclass relations. The merge operation is to merge the same classes and subclass relations to generate a concise resource space without losing semantics.

The merge operation provides the basis for accessing multiple resource spaces and distributed construction of resource spaces on large-scale dataset.

Table 1: The number of nodes and subclass relations for the original and merged dimensions.

Resource Space	Dimensions	Number of Nodes	Number of Subclass Relations
RS'	$network'$	821	1088
	$model'$	875	1080
RS''	$network''$	712	922
	$model''$	670	822
RS^*	$network^*$	1428	1885
	model*	1431	1722
RS	*network*	1428	1885
	model	1431	1722