

Chapter 7

Taxonomy Design

Wisam Dakka, Panagiotis Ipeirotis,
and Giovanni Maria Sacco

*“Think simple” as my old master used to say—
meaning reduce the whole of its parts into the
simplest terms, getting back to first principles.*

Frank Lloyd Wright, 1867–1959

This chapter discusses the design of taxonomies to be used in dynamic taxonomy systems. Although the only actual requirement of dynamic taxonomies is a multidimensional classification, i.e., objects are classified under more than one concept, an organization by facets is normally used.

In the first section, we provide guidelines for the design of DT taxonomies, which include the automatic construction from structured data, and the retrofitting of traditional monodimensional taxonomies. In the second section, we show how a faceted taxonomy can be automatically extracted from the infobase itself when objects are textual or are described by textual captions or tags.

7.1 General Guidelines for Taxonomy Design

The extensional inference rule of dynamic taxonomies has important implications on conceptual modeling. Inference simplifies taxonomy creation and maintenance because concept relationships are dynamically inferred, whereas all the possible relationships need to be described in traditional taxonomies. In addition, the inference rule coupled with conceptual summaries makes the relationships between different concepts immediately visible to the user. Finally, dynamic taxonomies can synthesize compound concepts so that they do not need to be explicitly represented in the taxonomy. This removes the principal cause of the combinatorial growth of traditional taxonomies.

These properties of dynamic taxonomies suggest to break down the conceptual taxonomy to a minimal set of basic constituent concepts or *fundamental facets*, whose combinations can fully describe the entire universe of discourse. This operation closely resembles normalization in relational database systems, because its goal is the reduction of (conceptual) redundancy, and the factoring of common properties. The reduction of redundancy minimizes the complexity of the resulting taxonomy.

At the same time, factoring common properties improves the exploration expressivity of the navigation, because it substitutes complex concepts with relationships among primitive concepts.

The principal guideline [236] is to organize the taxonomy as a set of independent, ‘orthogonal’ subtaxonomies (facets or perspectives), which behave as coordinates in a multidimensional space. Taxonomies produced in this way are ‘minimal’ without a decrease in expressivity. Such an organization, it must be stressed, is a design guideline, and not a requirement of the model, because dynamic taxonomies only require a multidimensional classification.

We can make the notion of ‘orthogonal’ facets clearer by recalling the analysis in Sect. 3.2. In a faceted organization, we have the fastest convergence when each facet F is independent of any other facet F' , $F' \neq F$, i.e., when given any two concepts $A < F$ and $B < F'$, $p(A|B) = p(B|A) = p(A)p(B)$. Such an independence can only be verified on the extension, and this criterion is therefore *a posteriori*. However, it can be used in design by checking if the underlying independence hypothesis is likely to hold in practice.¹

The practical criterion in arriving at a set of facets is to identify a set of single criteria which can be used to subdivide a given concept (initially, the universe of discourse) [276].² For example, a digital camera can be characterized by a set of single criteria such as ‘Price’, ‘Weight’, and ‘Resolution’.

In addition to facets, we can usually define primitive concepts which partition the corpus into disjoint sets. For example, a broad-scope corpus like an encyclopaedia can have primitive concepts like ‘art’, ‘science’, and ‘history’. These concepts, which are *partitioning aspects*, are different from facets, such as ‘location’ or ‘time’. Facets, which are rather *cross aspects* and tend to cover the entire corpus, rather than partitioning it.

Once the fundamental concepts of the universe of discourse have been identified, each will be described by a subtaxonomy, i.e., a hierarchy of subsumptions, typically specializations. Faceting and IS-A hierarchies are two independent conceptual devices, though both can be represented by subsumptions. In an IS-A hierarchy, the extension of a child of a concept C is generally a subset of the extension of C ; the union of the extensions of the children of C is usually equal to the extension of C . In a subdivision of C by facets, the extension of each facet (e.g., Price) is usually equal to the extension of C , as the digital camera example above shows.

The ‘faceting’ process can be repeated again at each level, but doing so constructs a static decision tree and consequently decreases the exploratory flexibility of DT

¹Statistical facet independence substitutes Ranganathan’s Canon of Concomitance [226] and the Classification Research Group (CRG) Principle of Mutual Exclusion [125, 276] in traditional facet analysis, which state that no two facets can overlap in content. Content overlap is determined on the basis of concept labels, and is therefore imprecise and not easily applicable to DTs where concepts are abstract.

²This is CRG’s Principle of Division [125].

access.³ In our context, faceting is generally used once for determining the root nodes of each subtaxonomy. There are however cases in which subsequent faceting can be used to simplify the taxonomy, and consequently user access. Assume we want to create an e-catalog for heterogeneous product types. Each of these product types⁴ (i.e., cameras, refrigerators, tv sets) will have a set of disjoint features, with Price being possibly the only feature common to them all.⁵ In this case, we can have a first subdivision by product type, and a second one by specific features (e.g., resolution, energy class).

Traditional facet analysis uses multiple principles to guide the faceting process. The most important ones, the Principle of Division and the Principle of Mutual Exclusion, have already been discussed. The Principle of Relevance, i.e., that division is performed according to criteria that are useful for access, clearly applies.

It is important to stress that the other two principles used by traditional facet analysis, the Principle of Ascertainability and the Principle of Permanence, do not apply in our present context. The Principle of Ascertainability requires that the division criterion must be *always* ascertainable: Ranganathan suggests that ‘date of death’ should not be used as a facet, because it is impossible to know when people will die [276]. This is obviously related to the problem of null values, and we can apply common solutions such as using special facet values like ‘unknown’, ‘not applicable’, etc., or simply not classifying under a facet if no child value applies.

The Principle of Permanence, i.e., facets should represent characteristics of division which represent permanent qualities of the concept being divided, is the legacy of the static physical medium underlying facet theory. In a traditional library, the classification of a book cannot change. In our present context, we have no problem in changing it, and this can be extremely useful in certain application areas such as e-auctions. In e-auctions, a facet ‘Time to completion’, which is not permanent, can be a fundamental access path.⁶

The extensional inference rule allows to represent *multiple inheritance* either explicitly (by a specific concept) or implicitly (by extensional inference). Consider the classic ‘working student’ example. A working student has a multiple inheritance because he inherits his properties from both ‘worker’ and ‘student’. We can represent this explicitly in the intension, by having a specific ‘working student’ concept in the taxonomy, and make it a child of both ‘student’ and ‘worker’. The visual interface remains the same, because the user will see ‘working student’ if he opens ‘student’, and ‘working student’ if he opens ‘worker’. Although ‘working student’ appears in different places in the taxonomy, it is actually the same concept, represented by the same internal id, and consequently identifies the same extension.

³This is actually prescribed by facet analysis where the fixed, predefined order used for division is called the ‘citation order’ and actually defines a static decision tree. This is one of the most evident differences between traditional faceted approaches and DTs.

⁴These are partitioning aspects.

⁵Price is a cross aspect.

⁶See also Sect. 8.1.7.2.

As an alternative, this multiple inheritance can be represented implicitly, by simply classifying every working student under ‘worker’ and ‘student’. This assumes that the existence of a multiple inheritance is captured by the classification system. In the case of multiple inheritance (and of any compound concept), the avoidance of compound concepts is an opportunity rather than a rule. Although the need for a compound concept often indicates that the taxonomy was not correctly defined, we advise to use an explicit representation (specific concepts) when the topic is specific, well-known, or deserving user attention [236]. For example, Internet is best described explicitly by a specific concept, rather than implicitly by the intersection of ‘computers’ and ‘communication networks’.

With respect to the ‘shape’ of the taxonomy, human factors dictate some restrictions on the number of levels and on the average number of children for each node. We believe that a number of children larger than approximately 10 objects makes the selection of the appropriate child(ren) to zoom on too difficult [236]. At the same time, an average number of levels larger than 3 or 4 is likely to result in taxonomies that require too much effort in order to access the terminal level and are difficult to understand. These two constraints place the number of terminal concepts for effective DTs roughly between 1,000 and 10,000.

Concepts with a large cardinality can cause an inordinate growth of the taxonomy. Such concepts include concepts with numeric values (prices, weights, etc.), with dates (start date, birth date, etc.), and concepts such as Persons, Companies, etc. Numeric and date concepts can be represented by ranges rather than by specific values, but this causes a loss of information.

If the corresponding data is available in a database, the designer can opt for an asymmetric approach and use an external query method (e.g., an SQL query) to focus on specific values, and summarize the result through a reduced taxonomy. Obviously, this approach does not allow to summarize focus sets through these concepts, and should be consequently avoided if such summaries are important. For example, in an infobase describing classical music recordings, it would probably be a bad idea not to represent composers in the dynamic taxonomy, but only in an external database. A focus on ‘String Quartets’ cannot be summarized by composer, which is an important navigational dimension in this application.

An alternate approach, which is often more appropriate and is discussed in Sect. 5.3, is the use of *virtual concepts* to virtualize part of the taxonomy. Virtual concepts appear to the user as *bona fide* concepts, but they are synthesized on demand from external concepts.

Although DTs taxonomies are usually balanced trees, it is sometimes beneficial to allow unbalanced subtrees in order to have terminal concepts with roughly the same selectivity, and consequently less variance in the cardinality of concept intersections [236]. Since the selectivity of a terminal concept depends on the extension, rather than on the intension, of the infobase, it is useful to periodically monitor the extension. A high number of objects classified under a terminal concept *C* usually indicates that a further refinement of *C* (increasing the abstraction level) is advisable. Conversely, a very small number of object classified under *C* indicates that *C*’s specializations may be discarded and documents directly classified under *C*, thereby decreasing the abstraction level.

However, increasing the abstraction level is

1. not always possible, because concepts are not infinitely specializable; and
2. not strictly required, because concept intersections considerably reduce the problems deriving from skewed distributions.⁷

A decrease in the abstraction level is useful to simplify the taxonomy and to reduce its storage requirements. The actual impact on user interactions is generally negligible because, especially if related counters are shown, the user will not expand the concept.

These considerations and our practical experience, suggest that dynamic taxonomies tend to be stable, and that schema update and maintenance is a relatively rare event, in practice, after the initial design and test phase. Base, fundamental concepts evolve slowly in time. The real dynamic part of a DT is given by the relationships among concepts, which often change very rapidly. However, these relationships are dynamically computed through the extensional rule, and do not require any change in the taxonomy.

Concept labels should be clearly understandable and unambiguous, and taxonomic abstractions should be clearly perceivable and consistent. Since the taxonomies designed according our guidelines minimize the number of required concepts, this task is easier than in traditional approaches. In addition, in DTs the meaning of any concept can be made clear by examples not only in the form of sample objects, but also in terms of summaries which highlight related concepts [236].

Finally, children of concepts must be arranged in a clear, consistent way. In this context, we can refer to traditional facet analysis [276] which proposes the Principle of Relevant Succession⁸ for child order. This principle identifies the following ordering strategies:

1. *Chronological Order*;
2. *Alphabetical Order*;
3. *Spatial/Geometric Order*, which orders children by contiguity according to seven possible ways: Bottom Upwards, Top Downwards, Left to Right, Clockwise Direction, Counter-Clockwise Direction, Periphery to Center, and Center to Periphery;
4. *Simple to Complex Order vs. Complex to Simple Order*;
5. *Canonical Order*, which is an established, traditional order for the subdivision of a concept. For example, the canonical order for the children of Philosophy is Logic, Epistemology, Metaphysics, Ethics, Aesthetics, etc.;
6. *Increasing Quantity vs. Decreasing Quantity*

Ranganathan also includes ordering by ‘literary warrant’, which lists children by decreasing number of objects listed under each concept. A similar, but more general, notion was used in Sect. 5.7.3. This is a dynamic ordering scheme, and its variation over time may disorient users.

⁷See Sect. 3.2.6.

⁸Also known as Ranganathan’s principles for helpful sequence.

The guidelines discussed above are generally valid for all DT applications. Taxonomy design for structured data is clearly a design ‘in the small’, i.e., a design for a limited, well-defined application domain. For example, the catalog of an e-shop is usually already available as a relation in a RDBMS. The focus here is on the automatic creation of taxonomies from structured data.

Taxonomy design for unstructured data (e.g., free text, images) is, at least potentially, a design ‘in the large’, in that it can be applied to describe the entire present (and future) knowledge. Often, these applications already use a monodimensional taxonomy, which can be transformed to a faceted multidimensional taxonomy as we discuss in the following.

7.1.1 Design ‘in the Small’

7.1.1.1 Automatic Construction for Views

To illustrate the basic design principles, we will consider the currently most frequent application of dynamic taxonomies: the intelligent exploration of a relation, or, more generally, of a relational view V , which can be derived from base relations and can also obviously represent the temporary result of a query. This structure is inherently multidimensional: each attribute in the view is an independent indexing dimension. Each tuple in the view is a DT object, which is classified in the taxonomy according to the values of its attributes.

The creation of a bare-bones shallow dynamic taxonomy from a relational view V can be accomplished as follows:

```
for each attribute  $a$  in  $V$ ,
    create a facet  $f(a)$  in the taxonomy
for each value  $v$  in the domain of  $a$ 
    create a son  $v$  of  $f(a)$  in the taxonomy
```

Each record r in V is then indexed by taking, for each attribute a in V , its value $r.a$ and indexing r under the corresponding son of $f(a)$. Since a concept that has no object classified under it is automatically pruned from the initial taxonomy, we only need to create for each facet $f(a)$ as many sons as the unique values in the attribute a in V .

This algorithm establishes a fundamental correspondence between facets and attributes of a relational view, which helps to put the guidelines described above in a more essential perspective. Interestingly, design techniques for relational databases, which are based on the Entity–Relationship model [64], consider the selection of attributes (facets) as an ‘intuitive’ initial step, and the design guidelines for it are likely to be “each property of interest of a specific entity is represented by an attribute”. This indicates that the fundamental principle for facet definition is really a principle of relevant division: each attribute inherently defines a division criterion, and only relevant attributes are defined.

In considering the mapping of a general relational view to a dynamic taxonomy, there are multiple considerations. First, the integration between dynamic taxonomies access and access through any other retrieval method (database or XML queries, in the present context) can be exploited, to reduce the number of facets. Not all the attributes in the view must necessarily have a counterpart in the dynamic taxonomy. However, attributes to be summarized must be explicitly described in the taxonomy.

Second, in the approach proposed above, attribute values are enumerated in the taxonomy as immediate children of a facet. While this is acceptable for ‘small’ enumerative domains, such as Brand, Country, etc. it becomes rapidly unwieldy as the number of different values increases. In addition, such a flat representation does not allow any systematic exploration. As an example, a global company would probably find a hierarchical grouping of locations into continents, nations, counties, etc., more useful than a flat list of locations. The same rationale holds for domains, such as numeric domains, whose potential number of different values is infinite, and are usually more manageable from the user point of view by structuring them in (multi-level) ranges of values. Consequently, in general, an independent subtaxonomy is defined for each attribute in order to structure existing values in a meaningful way.

Chakrabarti et al. [61] categorize results of relational queries by generating a summary multi-level taxonomy on the fly. The taxonomy generated is a plain taxonomy, so that their approach is a special case and considerably less powerful than dynamic taxonomies. The interesting point in their work is that the taxonomy is generated in such a way as to minimize a formally defined information overload cost for users. The adaptation of the higher level of the taxonomic tree is especially valuable as a way to define meaningful ranges of numeric values. As an example, consider the catalog of a megasite carrying everything from pins to elephants.⁹ Prices in such a store would exhibit such a large variance that it would be difficult to predefine value ranges which are meaningful for all users.

However, taxonomies generated in this way are not generally applicable because they do not necessarily capture the semantics of abstraction: e.g. the fact that Rome is in Italy and that Italy is in the European Union, which we contend is fundamental in most applications. An additional weakness of this method is that different query results produce different taxonomies, thereby disorienting the user.

These considerations suggest that a correct design for a general taxonomy should comprise both explicitly and implicitly defined concepts. Facets whose concepts can be taxonomically arranged in a hierarchical way, to model IS-A or PART-OF relationships, must be explicitly defined in the taxonomy. An example is a Location facet which can be organized by nations, continents, etc. We note here that more than one facet can correspond to a given attribute. For instance, Locations can be semantically structured in different ways: by nations and continents, or by organizations (NATO, OPEC), or by climate, etc. On the other hand, attributes with a very high number of different values and shallow facets (i.e., facets whose sons are ac-

⁹London’s Harrods motto.

tual attribute values) are best represented in an implicit way by queries on the view itself, as described in Sect. 8.1.

7.1.1.2 Dynamic Taxonomies for E–R Schemata

The mapping strategy outlined above is simple and effective, and has some affinities with star schemata in OLAP applications [62]. However, it consolidates all attributes together in a possibly large number of facets. This is undesirable because it makes user orientation difficult: a taxonomy with tens of facets would pose a cognitive challenge to most users.

A natural way of structuring the taxonomy in such a way as to make it easily understandable by users, is by using an Entity–Relationship schema¹⁰ as a starting point. An E–R schema structures the infobase as entities which represent real-world object types, and their relationships. Entities in the schema therefore provide natural top-level facets for the taxonomy, with the respective attributes appearing as immediate sons. The objects to be classified in the dynamic taxonomy are the tuples in the universal relation view constructed for the schema.

Regarding relationships, a first viable strategy is to represent them in the same way as entities, i.e., each relationship defines a top-level facet, with only the attributes of the relationship appearing as immediate sons. A relationship with no additional attributes is not explicitly represented in the schema. Participating entities need not be explicitly represented within the context of the relationship because the extensional inference rule establishes the right relationships between the top-level facets which represent the entities involved.

Consider the schema in Fig. 7.1 in which the primary keys of entities are assumed to be represented by surrogates (i.e. unique identifiers). This schema can be mapped into the following relational schema:

```
Part(Part#, Pname, Ptype)
Supplier(Supplier#, Sname, Slocation)
Plant(Plant#, PLname, PLlocation)
PlantUsesPart(Plant#, Part#)
SupplierSuppliesPart(Supplier#, Part#, Price)
```

First, we construct the universal relation view for this schema:

```
UR(Part#, Plant#, Supplier#, Pname, Ptype, PLname,
  PLlocation, Sname, Slocation, Price)
```

by outer-joining all the entities and relationships in the schema.

¹⁰The Enhanced Entity–Attribute model [97], or other extensions of the E–R model to conceptual hierarchies, are more adequate to the task at hand, but not as well known, and more complex.

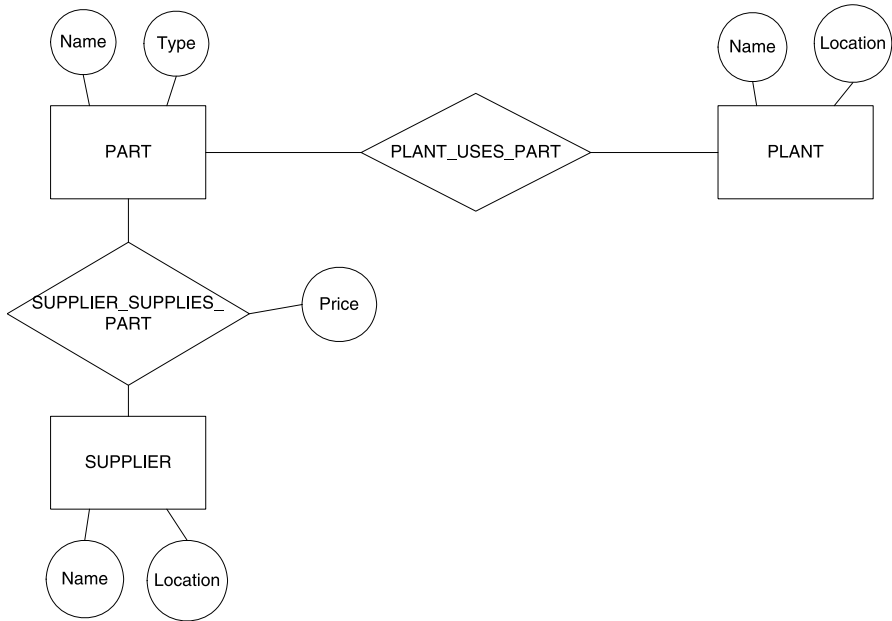


Fig. 7.1 Sample E–R schema; entities are implicitly identified by surrogates

This means that each object in the DT will be identified by a triplet (Part#, Plant#, Supplier#) such that Part# is supplied by Supplier# and/or Part# is used by Plant#. Both Plant# and Supplier# can be NULL.

Second, we decide which attributes will be described in the dynamic taxonomy, whether subtaxonomies are needed for selected attributes, and whether concrete or virtual concepts are to be used.

Third, we make entities and relationships explicit in the schema. A candidate dynamic taxonomy schema for the example is reported in Table 7.1.

As an example of use, assume that we zoom on a specific supplier *S*. The objects we are selecting are all those tuples in UR, such that Supplier# = *S*. The reduced taxonomy will report all the Parts supplied by *S*, and in addition all the Plants which are using such Parts (and their appropriate attribute values). In addition, also the Prices for Parts supplied by *S* will be reported.

The taxonomy reported in Table 7.2 presents an alternate strategy for representing relationships. Here, all the relationships are explicit top-level facets in the taxonomy and, in addition, all participating entities are explicitly represented in the context of each relationship as immediate sons.

This taxonomic schema shows a very important point: although the extensional inference rule infers unnamed relationships among concepts, the meaning of specific relationships can be made concrete and visible to the end-user. In this schema, relationship facets are used to disambiguate the unnamed relationships inferred by the extensional inference rule. When zooming on Part>Name>XYZ, the extensional inference rule establishes relationships among entity instances; the two facets repre-

Table 7.1 Taxonomy for sample E–R schema

Part
Name
Type
Mechanical
Electric
Electronic
Supplier
Name
Location
Africa
America
Asia
Plant
Name
Location
Africa
America
Asia
SupplierSuppliesPart
Price

senting relationships disambiguate the role of the part: whether it is a used-by-plant part or a supplied-by-supplier part or both. At the same time, if the user zooms on PlantUsesPart>Part>Name>XYZ, he specifies a specific role for XYZ. No role is specified by Part>Name>XYZ.

In summary, a top-level facet representing an entity represents such entity in any role, whereas a specific role is specified when this same facet is set as the son of another facet.

By converse, there might be attributes which are shared among different entities: in the current example, Location is an attribute both of Supplier and Plant. Instead of representing these attributes only as sons of the facets representing their entities, it is more convenient to add a top-level facet for each of them. This allows the user to zoom on a specific value of that attribute regardless of its role, i.e., of the entities to which the attribute is associated. In the examples in Tables 7.1 and 7.2, this means adding a top-level facet, Location: the user zooming on Location>Africa will select Plants and Suppliers in Africa.

Similar considerations apply to domains, and in particular to dates. A facet representing dates would allow the user to zoom on a specific date, and have a summary of all the entities and relationships related to that date. Obviously, this technique should be applied to attributes and domains only if focusing on them in a role-free way is useful for the user. Otherwise, the additional facets needed only make the taxonomy more complex and harder to understand.

Table 7.2 Alternate taxonomy for sample E–R schema

Part
Name
Type
Mechanical
Electric
Electronic
Supplier
Name
Location
Africa
America
Asia
Plant
Name
Location
Africa
America
Asia
PlantUsesPart
Plant
Name
Location
Africa
America
Asia
Part
Name
Type
Mechanical
Electric
Electronic
SupplierSuppliesPart
Supplier
Name
Location
Africa
America
Asia
Part
Name
Type
Mechanical
Electric
Electronic
Price

Dynamic taxonomies represent an intermediate model between traditional taxonomies and complex semantic models. Dynamic taxonomies are more powerful than plain taxonomies because traditional taxonomies only describe subsumptions, whereas dynamic taxonomies are able to represent, in a dynamic way, any kind of relationship that can be inferred from empirical evidence, that is from the classification itself.

Dynamic taxonomies are less powerful than general semantic networks or semantic data models, because these additional relationships are, in general, unnamed and therefore ambiguous. However, we have shown that the meaning of unnamed, inferred relationships can be made explicit by a careful design of the dynamic taxonomy. From the user point of view, both traditional and dynamic taxonomies are easily understood by end-users, whereas general semantic schemata are not. Whenever user access is important, the use of dynamic taxonomies which represent complex semantic schemata appears beneficial.

Dynamic taxonomies which are flexible and easily understandable by end-users can be derived from relational views and E–R conceptual schemata. If a database or a semantic information base already exists, the design methodology produces a dynamic taxonomy which captures the semantics of the information base and makes it easily available to end-user. Even if no schema exists, starting with traditional and well-understood data design techniques and applying the methodology we introduced will produce consistent and effective dynamic taxonomies that are at the same time exhaustive and easy to understand, even for demanding applications.

7.1.2 Design ‘in the Large’

The design of taxonomies ‘in the large’ is usually required for unstructured corpora (e.g., free text, images) with a very broad application domain. Examples include encyclopaedias, news feeds, very large image bases, catalogs of WWW resources, etc. In this context, design can be carried out through the general guidelines described above. Since the substantial equivalence between facets and E–R entities and attributes indicates that fundamental facets depend on the application domain, there are no predefined sets of facets for corpora with broad domains. Or, more precisely, no fixed set of facets is likely to be acceptable for all broad domains.¹¹

This does not necessarily mean that it is impossible to give indications for the selection of fundamental facets and partitioning aspects for very broad domains. Certainly, ‘Space’ (intended as location) and ‘Time’ (intended as Chronology) belong to the set of fundamental facets, as they are immediately relevant to most real-world objects. Ranganathan proposed five fundamental categories [226] to describe the entire universe of ideas:

- P (Personality, or Who): what the object is primarily ‘about’. This is the ‘main facet’;

¹¹This view is accepted also by many researchers in Information Sciences [276].

- M (Matter, or What): the material of the object;
- E (Energy, or How): the processes or activities which take place in relation to the object;
- S (Space, or Where): where the object happens or exists;
- T (Time, or When): when the object happens or exists.

The Bliss classification system [191] considerably extends the list of fundamental facets:

- thing
- kind
- part
- property
- material
- process
- operation
- patient
- product
- by-product
- agent
- space
- time

and defines the following partitioning aspects:

- Generalia, Phenomena, Knowledge, Information science & technology
- Philosophy & Logic
- Mathematics, Probability, Statistics
- General science, Physics
- Chemistry
- Astronomy and earth sciences
- Biological sciences
- Applied biological sciences: agriculture and ecology
- Physical Anthropology, Human biology, Health sciences
- Psychology & Psychiatry
- Education
- Society
- History
- Religion, Occult, Morals and ethics
- Social welfare & Criminology
- Politics & Public administration
- Law
- Economics & Management of economic enterprises
- Technology and useful arts
- The Arts
- Music
- Language and literature

Table 7.3 Dewey
classification fragment

800	Literature and Rethoric
	...
810	American literature in English
811	Poetry
812	Drama
813	Fiction
814	Essays
815	Speeches
816	Letters
817	Satire and humor
818	Miscellaneous writings
820	English literature
821	Poetry
822	Drama
823	Fiction
824	Essays
825	Speeches
826	Letters
827	Satire and humor
828	Miscellaneous writings
830	German literature
831	Poetry
832	Drama
833	Fiction
834	Essays
835	Speeches
836	Letters
837	Satire and humor
838	Miscellaneous writings
840	French literature
841	Poetry
842	Drama
843	Fiction
844	Essays
845	Speeches
846	Letters
847	Satire and humor
848	Miscellaneous writings

Other fundamental facets and partitioning aspects can be suggested by different sources, such as WordNet [100], Wikipedia at wikipedia.org, and the Open Directory Project at dmoz.org.

In many practical cases, the corpus might be already described by a taxonomy, but such a taxonomy is likely to be a traditional, monodimensional taxonomy. Monodimensional classification schemes are intuitively a bad idea. It is very difficult to find examples where only a single dimension or feature can be used to classify items. In fact, monodimensional schemes such as the Dewey classification for libraries [87] ‘linearize’ a multidimensional scheme into a monodimensional one. To do so, compound concepts are created and used. As an example, refer to the Dewey classification fragment in Table 7.3.

The reader will note that the entries are the cross product of two sets of label terms: {English, German, French, ... } and {poetry, drama, fiction, ... }. The first set represents the ‘Language’ facet, whereas the second set represents the ‘Literary Genre’ facet. Now the same fragment can be reorganized by facets as in Table 7.4. The retrofit of a monodimensional taxonomy to a faceted taxonomy is usually a fairly straightforward process, which mainly involves finding common terms in concept labels and factoring them out. The factoring process is usually simpler and more accurate if fundamental facets are preliminarily isolated. This allows one to disambiguate polysemic terms such as English, which means ‘written in English’ in ‘English poetry’, and ‘located in England’ in ‘English history’.

The advantages of the resulting faceted taxonomy are

1. the minimization of concepts, which decrease from 32 concepts in the Dewey fragment to 12 in the faceted fragment, i.e., from $n \cdot m$ to $n + m$;
2. an easy, symmetric correlation between features. If the user focuses on ‘drama’, she will find that there are dramas in different languages; if she focuses on ‘English’, she will find the different literary genres for English, including ‘drama’. In

Table 7.4 Faceted classification fragment

Language
American English
English
German
French
Literary Genre
Poetry
Drama
Fiction
Essays
Speeches
Letters
Satire and humor
Miscellaneous writings

the monodimensional taxonomy, access is asymmetric, and in the example only the second type of access is allowed;

3. the combinatorial complexity of compound concepts in a monodimensional taxonomy forces the taxonomy designer to a biased view of the universe. In the Dewey classification, each ‘major’ language has eight literary genre descriptors (poetry, drama, fiction, essays, speeches, letters, satire and humor, and miscellaneous writings). ‘Minor’ languages, such as Portuguese or Romanian have only one descriptor, and all the languages in East and Southeast Asia are grouped together into a single descriptor. It is easy to imagine that this will not be the perspective of a Portuguese or Thai classifier or user.

7.2 Automatic Construction from Text Information Bases

Faceted searching and browsing can be improved by utilizing various facets. However, in the presence of many facets, we have to choose which ones to present to the user. Presenting tens or hundreds of facets will make information access more difficult rather than easier. Hence, we have to select only the few that will be most useful for browsing purposes. For example, we would identify and assign video clips from a YouTube collection to the “Animals” or “Location” facets. Then, for each item in the collection, we would supply text-annotated keywords to describe the relationship between the item and the facet to which it has been assigned. Finally, we would use these text-annotated descriptions to construct faceted hierarchies for browsing the collection or lengthy search results. This chapter is dedicated to automation of this task to support wide deployment of faceted hierarchies over textual and text-annotated collections. One example of a text-annotated collection is the Corbis royalty-free image collection. Corbis has a large set of annotated images, in which each image has a title, free-text description, and a set of keywords associated with it. Each keyword is manually assigned by the Corbis annotators to one of the 38 facets that Corbis uses. The New York Times archive is an example of a large textual collection of news articles, dating to 1851.

In this chapter, we present methods to automatically discover facets and their useful browsing terms from a collection. In Sect. 7.2.1, we give a detailed overview of the problem of finding useful facet terms, and in Sects. 7.2.2 and 7.2.3 we describe supervised and unsupervised methods for extracting such facets from collections with textual and text-annotated data. In Sects. 7.2.4 and 7.2.5, we evaluate our approaches over two different textual and text-annotated collections. Finally, we elaborate on future work in Sect. 7.2.6 and conclude our chapter in Sect. 7.2.7.

7.2.1 Problem Overview

One of the bottlenecks in the deployment of faceted interfaces over collections of text or text-annotated documents is the need to *manually* identify useful dimen-



Fig. 7.2 A Flickr image shot by Sephirotty Fiesta

sions or facets for browsing a collection or lengthy search results. Once the facets are identified and assigned to the collection objects through descriptive keywords, a hierarchy is built and populated with the collection objects to enable the user to locate the objects of interest through the hierarchy. Static, predefined facets and their manually or semi-manually constructed hierarchies are usually used in commercial systems such as Amazon and eBay, with their faceted hierarchies for consumer products. The first step to automate the construction of faceted hierarchies is to identify the facets that are useful for browsing and assign them to the collection objects. In this chapter, we focus on this step.

We consider two types of collections in this chapter. A collection of the first type consists of objects that have some associated descriptive keywords or tags. One example of this type of collection is the Corbis royalty-free image collection. Corbis has a large set of annotated images, in which each image has a title, a free-text description, and a set of keywords associated with it. Each keyword is manually assigned by the Corbis annotators to one of the 38 facets that Corbis uses. Other examples include Flickr and YouTube. In contrast, a collection of the second type consists of free-text objects such as news articles in The New York Times archive or Newsblaster.

These two types of collections need to be processed differently for facet extraction, as the following example illustrates:

Example 7.1 Consider the Flickr image in Fig. 7.2 of a dog and a cat on a rocky beach. Typically, in Flickr-like collections each image (object) is tagged with several descriptive keywords. The image in Fig. 7.2 is associated with keywords “dog”, “cat”, “sea”, “beach”, “rock”, and “Tampico”. As we can see, these keywords describe several orthogonal aspects (facets) of the image: some describe the “Ani-

mals” in the image, some describe the “Topographic Features”, and others describe the “Location” where this image was taken. Knowing which keyword belongs to which facet is key for generating a meaningful browsing hierarchy for each facet. Unfortunately, we often do not have these keywords organized across facets and we resort to generating a single hierarchy that fits all keywords in a large collection such as Flickr or YouTube. A single hierarchy of this type is typically awkward and confusing for browsing. Collections such as The New York Times archive bring an additional challenge: news stories usually are neither associated with descriptive keywords nor organized across facets. Therefore, we have to resort to the story text to identify descriptive keywords and organize them across facets.

In this chapter, we investigate an extraction technique for each collection type. First, for the text-annotated collections (e.g., Flickr) that already have some keywords organized across different facets, we can use this facet data to train a machine learning algorithm to classify keywords in the appropriate facets (Sect. 7.2.2). The classifier can then be used to assign keywords of new objects to the right facets. For example, for a new image on Flickr that has the user-provided tags “sheep”, “fox”, “mountain”, and “fields”, our classifier will put the words “sheep” and “fox” under the “Animals” facet, while the words “mountain” and “fields” go under “Topographic Features”, even though the classifier may not have encountered some of the keywords beforehand. Second, for the collections with free-text objects such as The New York Time archive, we present an *unsupervised* technique that fully automates the extraction of useful facets from free-text objects (Sect. 7.2.3). In particular, we observe, through a pilot study, that facet terms rarely appear in text documents, which implies that we need external resources to identify useful facet terms. For this, we first identify important phrases in each document. Then, we expand each phrase with “context” phrases using external resources, such as WordNet [100] and Wikipedia,¹² causing facet terms to appear in the expanded collection. Finally, we compare the term distributions in the original collection and the expanded collection to identify the terms that can be used to construct browsing facets. Our extensive user studies, using the Amazon Mechanical Turk service, show that our techniques produce facets with high precision and recall, superior to existing approaches, and help users locate interesting objects fast.

7.2.2 *Supervised Facet Extraction for Collections of Text-Annotated Items*

Now, we describe our approach for extracting useful facets when we have access to some descriptive user-provided annotation, such as in the Corbis collection or on YouTube. One of the potential problems for constructing a concept hierarchy for

¹²<http://www.wikipedia.org>.

such a collection is that the same collection can be browsed in many different, orthogonal ways. Consider, for example, how a user can browse the schedule of TV programs. It is possible to browse by the time facet, by the TV channel facet, or by the title facet. It is also possible to browse by the actor facet, or by many other facets. Mixing facet-specific terms from multiple facets while constructing a single hierarchy can result in an awkward hierarchy. For example, an actor might be classified under the term “Monday” because he/she appears on a sitcom that is aired every Monday night, and therefore, the hierarchy would have the parent–child relation that assigns the actor’s name under the node “Monday” in the hierarchy. While it might be perfectly valid to assume this relation based on the co-occurrence of the term “Monday” and the name of the actor across the collection, this is not a structure that is useful for browsing. This type of relation contributes to the awkwardness of the resulting hierarchy. In short, having the items of a collection associated with descriptive keywords, such as with YouTube video clips, is not sufficient to produce useful hierarchies. Organizing these keywords across facets is a key step before proceeding to hierarchy construction.

While collections such as YouTube and Flickr lack such organization, a set of items in the Corbis collection has its keywords organized across predefined facets. For example, according to this Corbis set the words “cat” and “dog” are under the “Animals” facet, while the words “mountain” and “fields” are under “Topographic Features”. To be able to organize the keywords of a new item across the Corbis predefined facets, we use the data in this set to train a machine learning algorithm to classify keywords in the appropriate facets. In our approach, we treat the facet as a target classification class and the keywords as classification features. Unfortunately, such a straightforward approach does not generalize. A classifier trained in this way will correctly classify only words that have been assigned to facets before. A classifier might correctly classify the words “cat” and “dog” in the “Animals” facet, but a new word, such as “sheep”, that was not among the keywords of the training data will not be assigned to any facet.

To allow our technique to generalize for unseen keywords, we rely on the observation that keywords under different facets tend to have different “hypernyms”.¹³ Based on this observation, we expand each keyword using its hypernyms from a lexical corpus, such as WordNet. After the expansion, each keyword is represented as a set of words. For example, the word “cat” is represented as “cat, feline, carnivore, mammal, animal, living being, object, entity”. The new representation allows the classifier to generalize more easily and assign unseen words to the correct facets.

However, using hypernyms does not resolve the problem of sense disambiguation. Each word can have different meanings according to its context. Consider the word “kid”, which can mean either a young person or a young goat. Before assigning this word to a facet, we have to first decide the intended meaning of the word. To identify the correct meaning, we exploit the fact that keywords are associated with objects and each object is characterized by a set of other keywords, which provide

¹³ A hypernym is a linguistic term for a word whose meaning includes the meanings of other words, as the meaning of vehicle includes the meaning of car, truck, motorcycle, and so on.

valuable clues for the writer’s intended meaning of the word. (The use of context is the basis of many techniques [178] for sense disambiguation.¹⁴) For example, when the word “kid” appears together with the words “goat” and “grazing”, then “kid” is much more likely to refer to a young goat than to a child.

Based on the observations noted above, we treat facet classification as a text classification problem. In text classification [93, 168], we characterize each document using a set of words; based on the presence of these words across categories, we train a classifier to assign documents to the appropriate categories. In our case, we treat each keyword as *three sets* of words. The first set of words contains the keyword itself, the second set contains the hypernyms of all the senses of the keyword, and the third set contains the other keywords associated with the object.

Specifically, our algorithm for assigning keywords to facets performs the following steps:

1. Obtain a collection D of text-annotated objects. Each object $d_i \in D$ has a set of associated keywords k_{i1}, \dots, k_{in} and each keyword k_{ij} is assigned to a facet F_{ij} .
2. For each keyword–facet pair $k_{ij} - F_{ij}$:
 - (a) Define the facet F_{ij} as the target class.
 - (b) Add the keyword k_{ij} in the first set of words.
 - (c) Add the hypernyms of k_{ij} in the second set of words.
 - (d) Add the other keywords associated with d_i (and their hypernyms) in the third set of words.
3. Train a document classifier over the prepared training data.

After training the classifier, we can use it over a new set of annotated objects to identify the facets that appear in the collection. After running the classifier over the keywords of the new objects, we can examine which facets appear frequently in the new data and use these facets for browsing. Empirically, we observed that facets that appear in 5% of the data can be useful for locating content of interest. We gathered our training data from a set of annotated images from the Corbis collection, which contained a comprehensive set of facets.¹⁵ We describe the experimental settings and report the results in Sect. 7.2.4. One “disadvantage” of supervised learning techniques is that they cannot “discover” new types of facets. In the next section, we describe an approach to identify new, previously unknown dimensions for browsing.

¹⁴We should emphasize that disambiguation for facet extraction is easier than the general problem of sense disambiguation. First, the context keywords are of high quality, something that is not always the case in natural language sentences. Second, and most importantly, while a word might have multiple senses, the senses are often closely related (see, for example, the WordNet senses for “gear” and “battle”). While sense disambiguation is hard for such words, closely related senses typically correspond to the same facet (“Generic Thing” for “gear” and “Action, Process, or Activity” for “battle”), eliminating the need for disambiguation for facet extraction.

¹⁵The whole collection contains more than 3 million images and 38 facets.

7.2.3 *Unsupervised Facet Extraction for Collections of Text Documents*

So far, the identification of the facets was either a manual procedure, or relied on a priori knowledge of the facets that can potentially appear in the underlying collection. Now, we describe our approach for extracting useful facets when descriptive keywords are not available for the collection items. In particular, we focus on the important family of collection of text documents. A key characteristic of documents in such collections is that they contain a relatively large number of words (and, correspondingly, of potential facets for interaction). This is in contrast to the text-annotated collections of the previous section, where each item in the collection generally has a much lower number of (often highly descriptive) keywords in the user-provided annotations, and a portion of the items, as in Corbis, has its keywords organized across a predefined set of facets.

To examine the largest hurdles for generating faceted hierarchies on top of news collections, we ran a small pilot study (Sect. 7.2.3.1) to examine what navigational structures would be useful for people who are browsing a news archive and to find clues on how to discover useful facets accompanied with descriptive keywords for each text document in Newsblaster. Our conclusions from the pilot study helped shape our approach for extracting useful facets and descriptive keywords from news articles (Sect. 7.2.3.2). Our approach assumes that high-level facet terms rarely appear in the documents. For example, consider the named entity “*Jacques Chirac*”. This term would appear under the facet “*People* → *Political Leaders*”. Furthermore, this named entity also implies that the document can be potentially classified under the facet “*Regional* → *Europe* → *France*”. Unfortunately, these (facet) terms are not guaranteed to appear in the original text document. However, if we expand the named entity “*Jacques Chirac*” using an external resource, such as Wikipedia, we can expect to encounter these important *context terms* with greater frequency. Our hypothesis is that facet terms emerge after the expansion, and their frequency rank increases in the new, expanded collection. In particular, we take advantage of this property of facet terms to automatically discover, in an unsupervised manner, a set of candidate facet terms from the expanded news articles. We then automatically group together facet terms that belong to the same facet using a hierarchy construction algorithm [274] and build the appropriate browsing structure for each facet using our algorithm for the construction of faceted interfaces.

7.2.3.1 A Pilot User Study

For our initial pilot study, we recruited 12 students studying either journalism or art history. We randomly chose a thousand stories from *The New York Times* archive, and we asked the student annotators to manually assign each story to several facets that they considered appropriate and useful for browsing. The most common facets

Table 7.5 Facets identified by human annotators in a small collection of 1,000 news articles from The New York Times

Facets
Location
Institutes
History
People
↔ Leaders
Social Phenomenon
Markets
↔ Corporations
Nature
Event

identified by the annotators were “*Location*”, “*Institutes*”, “*History*”, “*People*”, “*Social Phenomenon*”, “*Markets*”, “*Nature*”, and “*Event*”. For these facets, the annotators also identified other “sub-facets” such as “*Leaders*” under “*People*” and “*Corporations*” under “*Markets*”.

From the results of the pilot study, we observed that the terms for the useful facets do not usually appear in the news stories. (In our study, this phenomenon surfaced in 65% of the user-identified facet terms.) Typically, journalists do not use general terms, such as those used to describe facets, in their stories. For example, a journalist writing a story about *Jacques Chirac* will not necessarily use the term “*Political Leader*” or the term “*Europe*” or even “*France*”. Such (missing) *context terms* are useful for identifying the appropriate facets for the story.

This pilot experiment demonstrated that a tool for the automatic discovery of useful facet terms should exploit external resources that could return the appropriate facet terms. Such an external resource should provide the appropriate context for each of the terms that we extract from the collection. As a result, a key step of our approach corresponds to an expansion procedure, in which the *important terms* from each news story are expanded with *context terms* derived from external resources. The expanded documents then contain many of the terms that can be used as facets. Next, we describe our algorithm in detail, showing how to identify these *important* and *context* terms.

7.2.3.2 Automatic Facet Discovery

The results of our pilot study from Sect. 7.2.3.1 indicate that general facet terms rarely occur in news articles. To annotate a given story with a set of facets, we normally skim through the story to identify important terms and associate these terms with other more general terms, based on our accumulated knowledge. For example, if we conclude that the phrase “Steve Jobs” is an important aspect of a news story, we can associate this story with general terms such as “personal computer”, “en-

Input: Original collection D , term extractors E_1, \dots, E_k

Output: Annotated collection $I(D)$

Step 1: Extract *all* terms from each document d in collection D and compute for each term t its term frequency $Freq_O(t)$.

Step 2: Execute *all* extractors E_1, \dots, E_k on each document d in collection D to identify d 's important terms $E_i(d)$ based on extractor E_i , and compute $I(d)$ to be the union $E_1(d) \cup \dots \cup E_k(d)$.

Algorithm 7.1: Identifying important terms within each document

entertainment industry”, or “technology leaders”. Our techniques operate in a similar way. In particular, our algorithm follows these steps:

1. For each document in the collection, identify the *important* terms *within* the document that are useful for characterizing the contents of the document.
2. For each important term in the original document, query one or more external resources and retrieve the *context* terms that appear in the results. Add the retrieved terms to the original document, in order to create an expanded, “context-aware” document.
3. Analyze the frequency of the terms, both in the original collection and the expanded collection, and identify the candidate facet terms.

Identifying Important Terms The first step of our approach (see Algorithm 7.1) identifies important terms¹⁶ in the text of each document. We consider the terms that carry information about the different aspects of a document to be important. For example, consider a document d that discusses the actions of *Jacques Chirac* during the *2005 G8 summit*. In this case, the set of important terms $I(d)$ may contain two terms, as follows:

$$I(d) = \{\text{Jacques Chirac}, \text{2005 G8 summit}\}$$

We use the next three techniques in this term selection step:

- **Named Entities (LPNE):** We use a named-entity tagger to identify terms that provide important clues about the topic of the document. Our choice is reinforced by existing research (e.g., [107, 129]) that shows that the use of named entities increases the quality of clustering and of news event detection. We build on these ideas and use the named entities extracted from each news story as important terms that capture the important aspects of the document. In our work, we use the named-entity tagger provided by the LingPipe¹⁷ toolkit.
- **Yahoo Terms (YTERM):** We use the “Yahoo Term Extraction”¹⁸ web service, which takes as input a text document and returns a list of significant words or

¹⁶By *term*, we mean single words and multi-word phrases.

¹⁷<http://www.alias-i.com/lingpipe/>.

¹⁸<http://developer.yahoo.com/>.

phrases extracted from the document.¹⁹ We use this service as a second tool for identifying important terms in the document.

- **Wikipedia Terms (WTERM):** We developed our own tool to identify important aspects of a document based on Wikipedia entities. Our tool is based on the idea that an entity is typically described in its own Wikipedia page. To implement the tool, we downloaded the contents of Wikipedia and built a relational database that contains (among other things) the titles of all the Wikipedia pages. Whenever a term in a document matches a title of a Wikipedia entry, we mark the term as important. If there are multiple candidate titles, we pick the longest title to identify the important term.

Furthermore, we exploit the link structure of Wikipedia to improve the detection of important terms. First, we exploit the “*redirect*” pages, to improve the coverage of the extractor. For example, the entries “Hillary Clinton”, “Hillary R. Clinton”, “Clinton, Hillary Rodham”, “Hillary Diane Rodham Clinton”, and others redirect to the page with title “Hillary Rodham Clinton”. By exploiting the redirect pages, we can capture multiple variations of the same term, even if the term does not appear in the document in the same format as in the Wikipedia page title. (We will also use this characteristic in Step 2, to derive context terms.) In a similar manner, we also exploit the *anchor text* from other Wikipedia entries to find different descriptions of the same concept. Even though the anchor text has been used extensively in the web context [50], we observed that the anchor text works even better within Wikipedia, where each page has a specific topic.

Beyond the three techniques described above, we can also follow alternative approaches in order to identify important terms. For instance, we can use domain-specific vocabularies and ontologies (e.g., from the *Taxonomy Warehouse*²⁰ by Dow Jones) to identify important terms for a domain. Here, due to the lack of appropriate text collections that could benefit from such resources, we do not consider this alternative. Still, we believe that exploiting domain-specific resources for identifying important terms can be useful in practice.

The next step of the algorithm uses important document terms to identify additional context terms, relevant to the documents.

Deriving Context Using External Resources In Step 2 of our approach, we use the identified important terms to expand each document with relevant context (see Algorithm 7.2). As we discussed in Sect. 7.2.3.1, in order to build facets for browsing a text collection, we need more terms than the ones that appear in the collection. To discover the additional terms, we use a set of *external resources* that can provide the additional context terms when queried appropriately.

For example, assume that we use Wikipedia as the external resource, trying to extract context terms for a document d with a set of important terms

¹⁹We have observed empirically that the quality of the returned terms is high. Unfortunately, we could not locate any documentation about the internal design of the web service.

²⁰<http://www.taxonomywarehouse.com/>.

Input: Annotated collection $I(D)$, external resources R_1, \dots, R_m

Output: Contextualized collection $C(D)$

Step 1: Query each external resource R_i to retrieve the context terms $R_i(t)$ for each important term t in $I(d)$ of each document d .

Step 2: Create for each document d the context terms $C(d)$ as the union of all context terms $R_i(t)$ of all terms t in $I(d)$ and all external resources R_1, \dots, R_m .

Step 3: Augment document d with context terms $C(d)$.

Algorithm 7.2: Deriving context terms using external resources (Sect. 7.2.3.2)

$I(d) = \{\text{Jacques Chirac, 2005 G8 summit}\}$. We query Wikipedia with the two terms in $I(d)$, and we analyze the returned results. From the documents returned by Wikipedia, we identify additional context terms for the two terms in the original $I(d)$: the term *President of France* for the original term *Jacques Chirac* and the terms *Africa debt cancellation* and *global warming* for the original term *2005 G8 summit*. Therefore, the set $C(d)$ contains three additional context terms, namely, *president of France*, *Africa debt cancellation*, and *global warming*.

In our work, we use four external resources, and our framework can be naturally expanded to use more resources, if necessary. We used two existing applications (WordNet and Google) that have proved useful in the past and developed two new resources (Wikipedia Graph and Wikipedia Synonyms). Specifically, the resources that we use are the following:

- **Google (GOOG):** The web can be used to identify terms that tend to co-occur frequently. Therefore, as one of the expansion strategies, we query Google with a given term, and then retrieve as context terms the most frequent words and phrases that appear in the returned snippets.
- **WordNet Hypernyms (WORDNET):** Previous studies in the area of automatic generation of facet hierarchies [79, 282] observed that WordNet *hypernyms* are good terms for building facet hierarchies. Based on our previous experience [79], hypernyms are useful and high-precision terms, but they tend to have low recall, especially when dealing with named entities (e.g., names of politicians) and noun phrases (e.g., “due diligence”). Therefore, WordNet should not be the only resource used but should be complemented with additional resources. We discuss such resources next.
- **Wikipedia Graph (WGRAPH):** A useful resource for discovering context terms is Wikipedia. In particular, the links that appear in the page of each Wikipedia entry can offer valuable clues about associations with other entries. To measure the level of association between two Wikipedia entries t_1 and t_2 that are connected with a link $t_1 \rightarrow t_2$, we examine two values: the number of outgoing links $out(t_1)$ from t_1 to other entries and the number of incoming links $in(t_2)$ pointing to t_2 from other entries. Using *tf.idf*-style scoring, we set the level of association to $\log(N/in(t_2))/out(t_1)$, where N is the total number of Wikipedia entries. (Notice that the association metric is not symmetric.) When querying the “Wikipedia

Graph” resource with a term t , the resource returns the top- k terms²¹ with the highest scores. For example, there is a page dedicated to the Japanese samurai “Hasekura Tsunenaga”. The “Hasekura Tsunenaga” page is linked to the pages “Japanese Language”, “Japanese”, “Samurai”, “Japan”, and several other pages. There are more than 6 million entries and 35 million links in the Wikipedia graph, creating an informative graph for deriving context. As expected, the derived context terms will be both more general and more specific terms. We will examine in Sect. 7.2.3.2 how we identify the more general terms, using statistical analysis of the term frequencies in the original collection and in the contextualized collection.

- **Wikipedia Synonyms (WSYNONYMS):** We constructed Wikipedia Synonyms as a resource that returns variations of the same term. As we described earlier, we can use the Wikipedia redirect pages to identify variations of the same term. To achieve this, we first group together the titles of entries that redirect to a particular Wikipedia entry. For example, the entries “Hillary Clinton”, “Hillary R. Clinton”, “Clinton, Hillary Rodham”, and “Hillary Rodham Clinton” are considered synonyms since they all redirect to “Hillary Rodham Clinton”.

Although redirect pages return synonyms with high accuracy, there are still variations of a name that cannot be captured like this. For such cases, we use the anchor text that is being used in other Wikipedia pages to link to a particular entry. For example, there is a page dedicated to the Japanese samurai “Hasekura Tsunenaga”. The “Hasekura Tsunenaga” has also pointers that use the anchor text “Samurai Tsunenaga”, which can also be used as a synonym. Since anchor text is inherently noisier than redirects, we use a form of *tf.idf* scoring to rank the anchor text phrases. Specifically, the score for the anchor text p pointing to a Wikipedia entry t is $s(p, t) = tf(p, t)/f(p)$, where $tf(p, t)$ is the number of times that the anchor phrase p is used to point to the Wikipedia entry t , and $f(p)$ is the number of different Wikipedia entries pointed to by the same text p .

At the end of Step 2, we create a *contextualized* collection in which each document contains both the original terms and a set of context terms. Next, we describe how we can use the term frequencies in the original and in the contextualized collection to identify useful facet terms.

Comparative Term Frequency Analysis So far, we have identified important terms in each document and used them to expand the document with general relevant context for each document. In this section we describe how we process both the expanded and original collections to identify terms that are good candidates for facet terms.

Our algorithm is based on the intuition that facet terms are infrequent in the original collection but frequent in the expanded one. So, to identify such terms, we need first to identify terms that occur “more frequently” and then make sure that this difference in frequency is statistically significant, and not simply the result of noise. To measure the difference in frequency, we define the next two functions:

²¹We set $k = 50$ in our experiments.

Input: Original collection D , contextualized collection $C(D)$

Output: Useful facet terms $Facet(D)$

Step 1: Compute $df(t)$ for each term t in collection D as the total number of documents that contain t .

Step 2: Compute $df_C(t)$ for each term t in collection $C(D)$ as the total number of documents that contain t .

Step 3: Let $df(t)$ be equal to zero if term t occurs in collection $C(D)$ but does not occur in collection D .

Step 4: Compute the functions $Shift_f(t)$ (equation (7.1)) and $Shift_r(t)$ (equation (7.4)) for each term t in collection $C(D)$, and add t to the facet terms $Facet(D)$ if both functions are positive.

Step 5: Sort $Facet(D)$ in increasing order of $-\log \lambda_t$ (equation (7.5)) and return the top- k terms.

Algorithm 7.3: Identifying important facet terms by comparing the term distributions in the original and in the contextualized collection (Sect. 7.2.3.2)

- **Frequency-Based Shifting:** For each term t , we compute the frequency difference as:

$$Shift_f(t) = df_C(t) - df(t) \quad (7.1)$$

where $df_C(t)$ and $df(t)$ are the frequencies of term t in the contextualized collection and the original collection, respectively. Due to the Zipfian nature of the term frequency distribution [337], this function tends to favor terms that already have a high frequency in the original collection. High-frequency terms demonstrate higher increases in frequency, even if they are less popular in the expanded collection compared to the original one. The inverse problem appears if we use ratios instead of differences. To avoid the shortcomings of this approach, we introduce a rank-based metric that measures the differences in the ranking of the terms.

- **Rank-Based Shifting:** We use a function B that assigns terms to bins based on their ranking in the original and the contextualized collections, as follows:

$$B(t) = \lceil \log_2(Rank(t)) \rceil \quad (7.2)$$

$$B_C(t) = \lceil \log_2(Rank_C(t)) \rceil \quad (7.3)$$

where $Rank(t)$ is the rank of the term t in the original collection, and $Rank_C(t)$ is the rank of the term t in the contextualized collection. After computing the bin $B(t)$ and $B_C(t)$ of each term t , we define the shifting function as follows:

$$Shift_r(t) = B(t) - B_C(t) \quad (7.4)$$

In our approach, a term becomes a candidate facet term only if both $Shift_f(t)$ and $Shift_r(t)$ are positive. After identifying terms that occur more frequently in the con-

textualized collection, the next test verifies that the difference in frequency is statistically significant. A test such as the chi-square test [65] could be potentially used to identify important frequency differences. However, due to the power-law distribution of the term frequencies [337], many of the underlying assumptions for the chi-square test do not hold for text frequency analysis [96]. Therefore, we use the *log-likelihood* statistic, assuming that the frequency of each term in the (original and contextualized) collections is generated by a binomial distribution:

- **Log-Likelihood Statistic:** For a term t with document frequency df in the original collection D and frequency df_C in the contextualized collection $C(D)$, the log-likelihood statistic for the binomial case is:

$$-\log \lambda_t = \log L(p_1, df_C, |D|) + \log L(p_2, df, |D|) - \log L(p, df, |D|) - \log L(p, df_C, |D|) \quad (7.5)$$

where $\log L(p, k, n) = k \log(p) + (n - k) \log(1 - p)$, $p_1 = \frac{df_C}{|D|}$, $p_2 = \frac{df}{|D|}$, and $p = \frac{p_1 + p_2}{2}$. For an excellent description of the log-likelihood statistic see the seminal paper by Dunning on the subject [96].

The shift functions and the log-likelihood test return a set of terms $Facet(D)$ that can be used for faceted navigation (see Algorithm 7.3). Once we have identified these terms, it is relatively easy to build the actual hierarchies. For our work, we used the subsumption algorithm by Sanderson and Croft [260], with satisfactory results, although newer algorithms [274] may improve performance further.

7.2.4 Evaluating Our Supervised Facet Extraction Technique

In this section, we present the experimental evaluation of our supervised extraction technique of Sect. 7.2.2. First, we describe our experimental settings in Sect. 7.2.4.1, then we evaluate our facet extraction technique in Sect. 7.2.4.2.

7.2.4.1 Experimental Settings

Our classifier variants are trained and tested over as a set of keywords associated across a predefined set of facets. We now describe this data set in and then, we briefly describe our classifier variants. Finally, we present the evaluation metrics that we use to compare our classifier variants.

Data Collection To evaluate our classifier of Sect. 7.2.2, we need a collection of items assigned to descriptive keywords across a set of predefined facet terms. For our experiments, we use 36,820 annotated images from the Corbis image collection, which we mentioned in Sect. 7.2.1. Each image has a title and free-text description, and is associated with a set of keywords. Each keyword is assigned manually by the

Table 7.6 List of the 14 most commonly used facets in Corbis

Facet	Description
ABC	Abstract Concepts
APA	Action, Process, or Activity
ATT	Attributes
ATY	Anatomy
GAN	Generic Animal
GCF	Generic Cultural Features and Works
GEV	Generic Event
GPL	Generic Plant
GTF	Generic Topographic Feature
GTH	Generic Thing
NCF	Named Cultural Features and Works
NORG	Named Organizations and Groups
NTF	Named Topographic Feature
RPS	Religious, Political, Philosophical, and Social Issues

Corbis annotators to one of the 38 facets that are used by Corbis. In total there are 65,521 unique keywords, primarily assigned to 14 of the 38 facets. The remaining 24 facets had less than 100 keywords assigned to them, so we ignored them for the purposes of our evaluation. Table 7.6 lists the 14 most commonly used facets with their full names. Since our facet extraction algorithm relies on the existence of pre-annotated data, we picked 11,000 keywords and their associated facets to train and test our algorithm. To avoid any bias, we randomly picked the 11,000 keywords from 11,000 randomly selected images, choosing one keyword per image.

Techniques for Comparison We evaluated three versions of our classifier using Support Vector Machines (SVM) with linear kernels. The first classifier, which serves as a weak baseline, does not use WordNet hypernyms or the associated keywords as features. The second, which serves as a strong baseline, uses WordNet hypernyms. The last classifier, which serves as our strongest version, uses both WordNet hypernyms and the associated keywords as additional features.

Evaluation Metrics To compare the three versions of our classifier, we use the following metrics, which are commonly used in statistical classification:

- Precision:** The precision of a classification class c is defined as the number of keywords (i.e., learning examples) that are truly labeled (by humans) and automatically classified (by a classifier) as c divided by the total number of keywords classified as c .
- Recall:** The recall of a classification class c is defined as the number of keywords that are both truly labeled and automatically classified as c divided by the total number of keywords that are truly labeled as c .

Table 7.7 The average performance of the facet extraction technique (strong classifier) for each of the 14 facets in the Corbis data set. Results are obtained using 10-fold cross-validation. (Table 7.6 contains the full names of the facets)

Facet	Precision	Recall	F_1 -measure
ABC	85.20%	87.60%	86.38%
APA	75.80%	75.80%	75.80%
ATT	78.20%	83.50%	80.76%
ATY	80.00%	81.30%	80.64%
GAN	92.90%	92.90%	92.90%
GCF	74.70%	76.76%	75.72%
GEV	79.40%	56.30%	65.88%
GPL	81.70%	90.10%	85.69%
GTF	86.70%	75.00%	80.43%
GTH	87.70%	83.00%	85.29%
NCF	82.40%	87.57%	84.91%
NORG	75.40%	76.58%	75.99%
NTF	82.40%	80.30%	81.34%
RPS	85.60%	76.30%	80.68%
Average	82.01%	80.22%	80.89%

F_1 -measure: The F_1 -measure of a classification class is defined as the weighted harmonic mean of precision and recall, or

$$\frac{2 \cdot \textit{Precision} \cdot \textit{Recall}}{\textit{Precision} + \textit{Recall}}$$

7.2.4.2 Experimental Results

Initially, we tested the accuracy of the weak baseline and, as expected, the classifier could not generalize to unseen examples. Its accuracy, as measured by the average F_1 -measure, was 10%, only slightly higher than the accuracy of a random classifier. By adding the WordNet hypernyms, the performance of the baseline classifier improved considerably, reaching an average F_1 -measure of 71%. This improvement confirmed our hypothesis that hypernyms are useful features for allocating keywords to facets. Nonetheless, the sense ambiguity is still a problem in this case: after adding the remaining keywords from each document as extra features, performance improved considerably. Our strong classifier reached an average F_1 -measure of 81% (see Table 7.7).

We also compared our strong classifier against variations of other techniques. One hypothesis was that we can create facets by picking some high-level hypernyms from WordNet, which can serve as root nodes for the corresponding facets.

For example, the term “animal/fauna” in WordNet could serve as the root node for the “Animal” facet. Subsequently, all terms that have “animal/fauna” as a hypernym could be assigned to the “Animal” facet. (This approach is close in spirit to the hierarchy construction algorithm in [281].) To test the accuracy of this approach, we trained RIPPER [71], a *rule-based* classifier, using the keywords and their hypernyms as features. The average F_1 -measure in that case was close to 55%, significantly worse than the corresponding results for SVMs. The results also highlighted that some classes (facets) work well with simple, rule-based assignments of terms to facets, but there are other classes that need more elaborate classifiers. For example, for the facet GAN (Generic ANimals) the rule-based classifier resulted in an F_1 -measure of 93.3%, showing that simple rules work well for this facet. However, for the APA facet (Action, Process, or Activity) the F_1 -measure was only 35.9%, showing that simple rules do not work well for such a complex facet.

7.2.5 Evaluating Our Unsupervised Facet Extraction Technique

In this section, we discuss the experimental evaluation of our unsupervised extraction technique of Sect. 7.2.3. First, we discuss the settings we used for our evaluation, including the data sets and human judgments, our technique variations, and our evaluation metrics. Then, we describe how we evaluated the recall and precision of our techniques. Finally, we present our results on the efficiency of our techniques, and finally we briefly discuss some results of a user study that demonstrates the usefulness of the derived faceted navigational structures.

7.2.5.1 Experimental Settings

Our technique receives as input a set of free-text documents and produces as output a set of hierarchically structured facets that can be used to browse the text collection in an OLAP-like manner. Sect. 7.2.5.1 describes three data collections that we use to evaluate the variants of our techniques, and an Amazon Mechanical Turk study to evaluate the quality of the extracted facets by human annotators. Section 7.2.5.1 describes the variants of our technique. Finally, Sect. 7.2.5.1 presents the evaluation metrics for the experiments.

Data Collections

Single Day of The New York Times (SNYT): A collection of 1,000 news stories from The New York Times archive, covering one day of news from November 2005.

Single Day of Newsblaster (SNB): A collection of 17,000 news stories retrieved by Newsblaster [183] from 24 news sources on one day in November 2005. We use this data set to test how our techniques work over data from multiple sources.

Month of The New York Times (MNYT): A collection of 30,000 news stories from The New York Times archive, covering one month of news from November 2005.

Human Extracted Facet Terms: Since our experiments required extensive input from users and there is no standard benchmark for evaluating the quality of the generated facets, we conducted a human study on the Amazon Mechanical Turk service. Specifically, in our study each Mechanical Turk annotator had to read a story and identify terms that can be used for faceted navigation. We informed the annotators that the terms may or may not appear in the document, and it was up to the annotator to determine whether the terms were useful for the task of faceted navigation. For each article, the annotators were asked to provide up to 10 candidate facet terms. We instructed them to choose terms that were clearly defined, mutually exclusive, and covered as many aspects, properties, or characteristics of the story as possible. Each of the 1,000 stories in *SNYT* was examined by five annotators. For *SNB* and *MNYT*, we picked a random sample of 1,000 stories, and again each story was annotated by five annotators. To eliminate annotation errors or idiosyncratic annotations, we considered an annotation to be valid if at least two of the annotators used the same facet term for a given story. The final set contained 633 facet terms for *SNYT*, 756 facet terms for *SNB*, and 703 terms for *MNYT*. This indicates that the number of facet terms increases relatively slowly as the number of news sources rises (i.e., from *SNYT* to *SNB*) and as we expand the examined time period (i.e., from *SNYT* to *MNYT*). To make sure that this is not an artifact of our sampling approach, we also performed a sensitivity test, examining how the number of facet terms increases as we increase the number of stories in each data set from 100 to 1,000. At 100 documents, we discovered approximately 40% of the facet terms, and at 500 documents we discovered approximately 80% of the facet terms for each of the data sets. Therefore, we believe that the marginal increase in facet terms if we annotated all 17,000 articles for *SNB* and all 30,000 articles for *MNYT* would be relatively small. Figure 7.3 shows a sample of the most frequently identified facet terms for the three data sets.

Precise Extracted Facet Terms: Our techniques extract a significant number of terms that the Mechanical Turk annotators did not identify when marking the important facet terms in the documents (see earlier details on the study). Still, when examining the extracted faceted terms with the generated hierarchies, we could easily determine whether a particular facet term accurately depicts the contents of the underlying collection. So, we asked the annotators to examine the extracted facet terms with the generated hierarchies and determine for each facet term the following: (a) whether the term is useful in the facet hierarchy and (b) whether the term is accurately placed in the hierarchy. To ensure the quality of the annotations, the Mechanical Turk annotators that participated in this experiment had to pass a *qualification test*. To conduct our test, we initially picked random subtrees of the Open Directory²² hierarchy as our “correct” hierarchies.

²²<http://www.dmoz.org>.

Table 7.8 A list of our external resources (Sect. 7.2.3.2)

Name	Description
GOOG	Querying Google and using the results snippets.
WORDNET	Querying WordNet for related Hypernyms.
WSYNONYMS	Querying Wikipedia for related Synonyms.
WGRAPH	Querying Wikipedia Graph for related aspects.
<i>All</i>	Combining all resources.

Table 7.9 A list of our term extractors techniques

Name	Description
LPNE	Extracting terms using LingPipe’s name entity tagger.
YTERM	Extracting terms using “Yahoo Term Extraction” web service.
WTERM	Extracting terms using our Wikipedia term extractor.
<i>All</i>	Combining all extractors.

To generate “noisy” hierarchies, we randomly perturbed some parent–child relations and randomly swapped terms across separate hierarchy subtrees. Then, during the qualification test, each prospective Mechanical Turk annotator had to annotate 20 correct and incorrect hierarchies and was only allowed to proceed with the real annotation task if he or she gave the correct answer for at least 18 out of 20 hierarchies. As in the case of the earlier study, each facet term was examined by five Mechanical Turk annotators. We only consider a term to be a precise facet term if at least four annotators marked the facet term as precise.

Techniques for Comparison We create several variations of the general technique that we described in Sect. 7.2.3 based on (1) four external resources we use to expand the collection, namely, GOOG, WORDNET, WSYNONYMS, and WGRAPH; and (2) three term extractor techniques, namely, LPNE, YTERM, and WTERM. Tables 7.8 and 7.9 list the details of our external resources and term extractors, respectively.

Evaluation Metrics To evaluate the different variations of our technique, we use the following metrics.

Recall: This metric measures how many of the manually extracted facet terms were also identified by our techniques. We define recall as the fraction of the manually extracted facet terms that were also extracted by our techniques.

Precision: We also evaluate the precision of the extracted facets using the same methodology that we used for estimating recall. However, our techniques extract a significant number of concepts that the Mechanical Turk annotators did

politics, money, market, government, history, competition, people, education, location, new york, power, terrorism, war, baseball, event, biography, business, children, development, health, music, real estate, sports, change, comeback, crime, entertainment, greed, national security, nature poverty, spending, success, pride, technology, winning, anger, architecture, branding, foreign lands, bush administration, campaign, capitalism, challenges, civil unrest, civil war, community, compromise, computers, consumer confidence, corruption, countries, culture of fear, disagreement, distribution, power of the Internet, expectations fear, humor, innovation, investigation, Iraq, Italian culture, jobs, leadership, moving, opportunity, optimism, planning, players, police, public relations, publicity, religion, religious, warfare rights, statistics, support, time, torture, U.S., violence, wealth, youth

Fig. 7.3 A sample of the most frequently identified facet terms, as extracted by human annotators. All the terms above were anonymously selected by at least two annotators

year, new, time, people, state, work, school, home, mr, report, game, million, week, percent, help, right, plan, house, high, world, american, month, live, call, thing

Fig. 7.4 Facet terms identified by a simple subsumption-based algorithm [260], without using our techniques

not identify when marking the important facet terms in the documents. We consider an extracted term to be a “precise” facet term if at least four annotators marked it as precise. The precision is then the ratio of precise extracted terms over the total number of extracted terms.

Efficiency: We also measure another important aspect of our techniques, namely, the time required to extract useful facet terms for a collection.

7.2.5.2 Experimental Results

Recall Our first step was to measure how many of the manually extracted facet terms were also identified by our techniques. To examine the individual effect of each term extractor and of each external resource, we computed the fraction of identified facet terms for each of the possible combinations of term extractor and external resource. We also computed the recall for the case in which we used *all* the term extractors and *all* the external resources.

We list the results in Tables 7.10, 7.11, and 7.12 for the *SNYT*, *SNB*, and *MNYT* data sets, respectively. The results were consistent across data sets. In general, recall increases as we increase the number of term extractors and as we increase the number of external resources. *WSYNONYMS* and *WORDNET* tend to perform relatively poorly compared to *Google* and *WGRAPH*, especially when using named entities (*LPNE*) as the term extractor. However, both resources are helpful when combined with *GOOG* and *WGRAPH*, and increase the overall recall of the results.

Precision Recall that we consider a facet term to be “precise” if and only if it was identified by our annotator as useful within the generated hierarchy of the facet

Table 7.10 The recall of the extracted facets, as judged by the human annotators for the *SNYT* data set

External resource	Term extractors			
	LPNE	YTERM	WTERM	All
GOOG	0.529	0.703	0.761	0.819
WORDNET	0.090	0.510	0.491	0.592
WSYNONYMS	0.105	0.156	0.345	0.408
WGRAPH	0.632	0.791	0.801	0.881
All	0.746	0.891	0.899	0.945

Table 7.11 The recall of the extracted facets, as judged by the human annotators for the *SNB* data set

External resource	Term extractors			
	LPNE	YTERM	WTERM	All
GOOG	0.515	0.658	0.699	0.751
WORDNET	0.084	0.487	0.395	0.514
WSYNONYMS	0.112	0.162	0.306	0.314
WGRAPH	0.615	0.755	0.773	0.823
All	0.707	0.861	0.856	0.881

Table 7.12 The recall of the extracted facets, as judged by the human annotators for the *MNYT* data set

External resource	Term extractors			
	LPNE	YTERM	WTERM	All
GOOG	0.522	0.658	0.699	0.793
WORDNET	0.087	0.487	0.395	0.555
WSYNONYMS	0.109	0.146	0.331	0.392
WGRAPH	0.627	0.778	0.790	0.853
All	0.733	0.859	0.860	0.921

and is accurately placed in the hierarchy. And we defined the precision to be the ratio of precise terms over the total number of extracted facet terms. (We discuss the generation of faceted hierarchies in the following chapter.)

We list the precision results in Tables 7.13, 7.14, and 7.15 for the *SNYT*, *SNB*, and *MNYT* data sets, respectively. Again, the results were consistent across data sets. The highest precision hierarchies are those generated by WordNet; this is not surprising since the hypernyms naturally form a hierarchy. The use of Google as an external resource tends to lead to a drop in precision. In our implementation, for efficiency, we only use the terms that appear in the titles and snippets in the Google results; we do not retrieve the actual HTML pages of the returned results. This

Table 7.13 The precision of the extracted facets, as judged by the human annotators for the *SNYT* data set

External resource	Term extractors			
	LPNE	YTERM	WTERM	All
GOOG	0.615	0.769	0.751	0.678
WORDNET	0.923	0.901	0.932	0.915
WSYNONYMS	0.734	0.815	0.845	0.819
WGRAPH	0.828	0.813	0.842	0.827
All	0.817	0.796	0.858	0.866

Table 7.14 The precision of the extracted facets, as judged by the human annotators for the *SNB* data set

External resource	Term extractors			
	LPNE	YTERM	WTERM	All
GOOG	0.505	0.796	0.751	0.714
WORDNET	0.897	0.919	0.909	0.922
WSYNONYMS	0.633	0.904	0.875	0.853
WGRAPH	0.789	0.851	0.885	0.822
All	0.796	0.815	0.834	0.831

Table 7.15 The precision of the extracted facets, as judged by the human annotators for the *MNYT* data set

External resource	Term extractors			
	LPNE	YTERM	WTERM	All
GOOG	0.487	0.818	0.834	0.794
WORDNET	0.878	0.925	0.932	0.917
WSYNONYMS	0.691	0.851	0.880	0.879
WGRAPH	0.801	0.824	0.837	0.810
All	0.713	0.836	0.855	0.861

approach introduces a relatively large number of noisy terms. An interesting direction for future research is to examine whether the introduction of a term extraction mechanism from the HTML pages could improve the precision of our hierarchies. In contrast to Google, the use of the Wikipedia resources produces more precise hierarchies. Given the high precision of the WordNet- and Wikipedia-based hierarchies, it would be interesting to see if we could use ontologies that combine WordNet and Wikipedia in a single resource [283] as external resources.

Efficiency In our experiments, term extraction took 2–3 seconds per document, and the main bottleneck was the Yahoo! Term Extractor. If we eliminate the Ya-

hoo! Term Extractor, then we can process approximately 100 documents per second. Similarly, document expansion takes approximately one second per document when using Google as an external resource. Using Wikipedia and WordNet, which are stored locally, is significantly faster: we can process more than 100 documents per second, effectively making the term extraction the real bottleneck in the process. The facet term selection phase is extremely fast (i.e., about a few milliseconds), and we use an efficient hierarchy construction using the techniques described in [79], to create the facet hierarchies in 1–2 seconds.

In a real deployment scenario, we can considerably increase facet extraction efficiency by performing the term and context extraction offline. In this case, the results are ready before the real facet computation, which then takes only a few seconds and is almost independent of the collection size. (So, we can generate facet hierarchies over the complete collection *and* dynamically over a set of lengthy query results.) If term and context extraction need to be performed on-the-fly over thousands of documents, and it is important to compute the facet hierarchies quickly, then it would be preferable to avoid using web-based resources, such as Yahoo! Term Extractor and Google, and instead use only locally available resources, such as LingPipe, Wikipedia, and WordNet.

User Study Finally, we examine the reaction of users to automatically extracted facets. For this, we recruited five subjects to use our system to locate news items of interest, and we asked them to repeat the task 5 times. We provided a keyword-based search interface that was augmented with our faceted hierarchies located on the side. We measured how often during each search session the users clicked on the facet terms and how often they used the keyword-based search. We also measured the time required to finish the task. At the end of each session, we asked users to indicate their level of satisfaction, on a scale of 0–3, where 0 = dissatisfied, 1 = slightly dissatisfied, 2 = slightly satisfied, and 3 = satisfied.

We observed that, in the *first* interaction with the system, the users typed a keyword query of a named entity associated with the general topic in which they were interested (e.g., “war in Iraq”). Then they proceeded to locate news stories of interest by clicking on the facet hierarchies until they had generated a small subset of news stories associated with the same topic. Interestingly, in subsequent interactions with the system, the users started by using the faceted hierarchies directly, and their use of the keyword-search interface was gradually reduced by up to 50%. In addition, the time required to complete each task dropped by 25%, and the level of satisfaction remained statistically steady, with a mean level of 2.5 in the 0–3 scale. These results are consistent with previous research studies that relied on manually generated facet hierarchies [327] or on hierarchies extracted only from WordNet [282].

Results Summary The results of our user study indicate that users are generally comfortable using facet terms, and grow even more comfortable with greater exposure to them. Furthermore, by using the facets, users can locate items of interest more quickly, without any decline in satisfaction. The similarity of the interface with existing OLAP tools means that our techniques can be seamlessly integrated

with current OLAP systems that provide easy access to structured, relational data. Our techniques can expand existing OLAP systems to work over unstructured, or semi-structured data, allowing OLAP users to quickly discover interesting associations.

7.2.6 Further Discussion and Future Work

We evaluated our supervised approach over a text-annotated collection, refined with the use of training data in which descriptive keywords are assigned to the items in the collection and the keywords are organized across facets terms. We evaluated our unsupervised approach over a news collection with the benefit of a hypothesis that useful high-level browsing terms can be found in external knowledge resources. Unfortunately, we only evaluated the supervised approach over the Corbis collection mainly because we do not have access to other collections that are annotated in a similar manner as Corbis, and we only evaluated the unsupervised approach over collections of news articles. For future work, we are interested in combining the two approaches to extend our work for semi-annotated collections such as YouTube: each YouTube item (or video clip) is tagged with several keywords, as in the Corbis collection, but is not associated with useful facets, unlike in the Corbis collection.

Our unsupervised techniques rely on external informative resources and we only query three such resources. In fact, there are several other useful resources within specialized contexts that could be relatively straightforward to integrate into this framework. For instance, the *Taxonomy Warehouse*²³ developed by Dow Jones contains a large list of controlled vocabularies and specialized taxonomies that can be used for term identification and term expansion, respectively. For example, when browsing literature for financial topics, we can use one of the available glossaries to identify financial terms in the documents. Then, we can expand the identified terms using one or more of the available financial ontologies and thesauri. In fact, we plan to incorporate many such resources in our framework for a variety of topics, and use them all, irrespectively of the topics that appear in the underlying collection. This will allow us to handle a variety of collections, beyond archives of news articles.

7.2.7 Conclusion

We presented techniques for automatically identifying terms that are useful for building faceted hierarchies over two kinds of collections. For a text-annotated collection, such as Corbis, we built a classifier that can associate, with a high level of accuracy, annotated keywords across a pre-defined set of facets. For a free-text

²³<http://www.taxonomywarehouse.com/>.

collection such as The New York Times archive, we built a set of techniques that develop the idea that external resources, when queried with the appropriate terms, provide useful context that is valuable for locating the facets that appear in a collection of text documents. We demonstrated the usefulness of Wikipedia, WordNet, and Google as external resources. Our experimental results, validated by an extensive study using human subjects, indicate that our techniques generate facets of high quality that can improve the browsing experience for users.