Taxonomizer: Interactive Construction of Fully Labeled Hierarchical Groupings from Attributes of Multivariate Data

Salman Mahmood and Klaus Mueller, *Senior Member, IEEE*

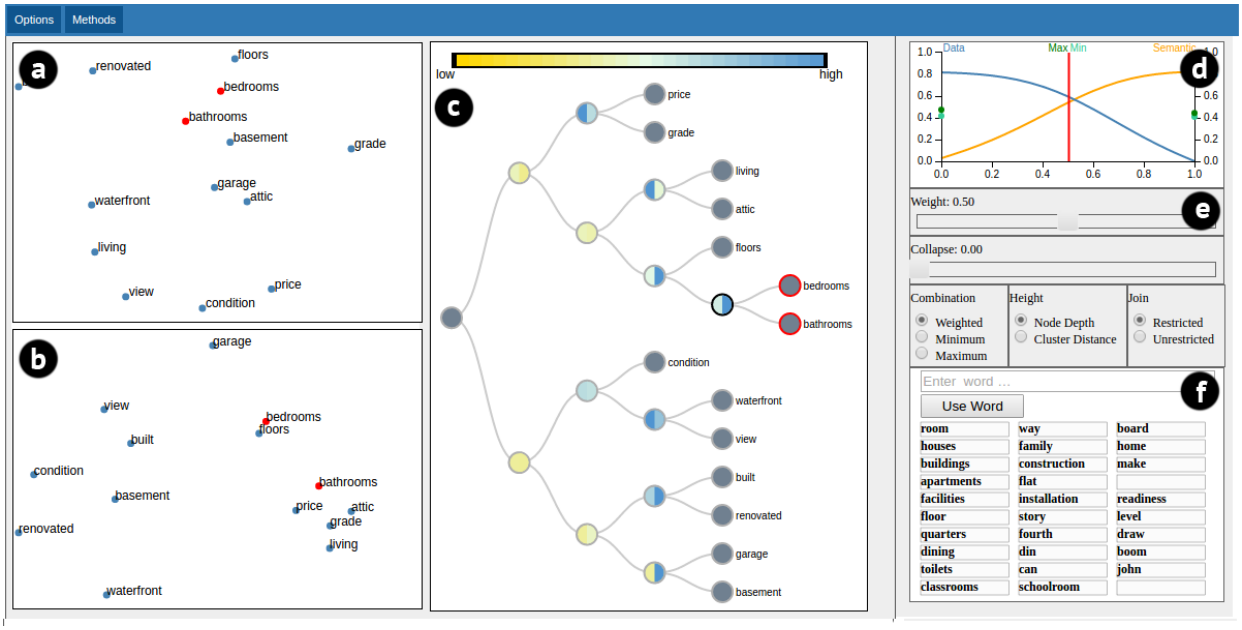


Figure 1: The housing dataset for Kings County displayed using Taxonomizer. The user interface consists of six coordinated views. (a) The semantic space. (b) The data space. (c) The hierarchy built from combining information spaces (a) and (b). (d) The cophenetic correlation plot which allows users to specify the weight of (a) and (b) to generate (c). (e) The control panel gives the user various options to manipulate the structure of the hierarchy. (f) The word suggestion panel gives suggestions for labeling the nodes of the hierarchy.

Abstract— Organizing multivariate data spaces by their dimensions or attributes can be a rather difficult task. Most of the work in this area focuses on the statistical aspects such as correlation clustering, dimension reduction, and the like. These methods typically produce hierarchies in which the leaf nodes are labeled by the attribute names while the inner nodes are often represented by just a statistical measure and criterion, such as a threshold. This makes them difficult to understand for mainstream users. Taxonomies in science, biology, engineering, etc. on the other hand, are easy to comprehend since they provide meaningful labels at the inner nodes as well. Labeling inner nodes of taxonomies automatically requires the identification of hypernyms. Our proposed framework, called Taxonomizer, takes a visual analytics approach to meet this challenge. It appeals to the wisdom of humans to liaise with state of the art data analytics, neural word embeddings, and lexical databases. It consists of a set of visual tools that starts out with an automatically computed hierarchy where the leaf nodes are the original data attributes, and it then allows users to sculpt high-quality taxonomies for any multivariate dataset.

Index Terms—High-Dimensional Data, Data Fusion and Integration, Hierarchy Data, Taxonomy, Neural Embeddings, Lexical Databases

1 Introduction

In an era of unprecedented data fecundity one of the biggest challenges is the difficulty of extracting relevant information from these data. Realistic datasets often have an abundance of attributes.This is a blessing as it enables better decision-making based on a more complete depiction and modeling of the world. But it is also a curse because having too many factors around at the same time can be overwhelming to most if not all humans. There is hence a growing need for developing systems that can aid human cognition in organizing the attributes and factors in a way that facilitates pattern discovery, analysis, and characterization.

The most natural way to allow users to navigate a complex attribute space is via a hierarchy. There are a number of tools available that can organize multivariate data into hierarchies mostly using statistical aspects of the data such as correlation clustering, dimension reduction, and others. In these hierarchical structures, however, while the leaf nodes bear the real-world attribute names, the inner nodes are often labeled with criteria related to the statistical measures, such as thresholds or ranges, making them difficult to interpret for mainstream users. We propose to make the inner nodes more interpretable by using a taxonomy paradigm to facilitate the exploration of multivariate space.

• *Salman Mahmood and Klaus Mueller are with the Visual Analytics and Imaging Laboratory, Department of Computer Science, Stony Brook University, NY and SUNY Korea, Incheon, Korea. E-mail: (samahmood, mueller)@cs.stonybrook.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

Taxonomy is the process of naming and classifying things such as animals and plants into groups within a larger system, according to their similarities and differences. Hierarchies are often presented in visual form (Figure 2 shows the taxonomy of a university dataset). The internal nodes of a hierarchical tree are labeled to facilitate the storing and retrieving of items from a repository. The label of each node in the tree captures the high-level concept expressed by its descendants.

Visualization tools often provide a list of attributes for users to explore datasets. However, this method is very limiting because it restricts the exploration process to single attributes. Higher level concepts can also be explored by combining the attributes of the data. A dataset on universities has 14 attributes[1]. Attributes like 'academic' and 'faculty' relate to education. Whereas, attributes like 'nightlife' and 'dining' relate to the city/area the university belongs to. The exploration process may benefit by having access to these higher level abstractions of the data. We want to use the paradigm of taxonomy to organize the attributes of a dataset in a way that allows the user to interact with the data at various levels of granularity. Figure 2 shows a taxonomy for the university dataset. The attributes of the dataset form the leaves of the tree. The data for the inner nodes can be estimated by applying a dimension reduction algorithm on the attributes that are descendants of that node. The taxonomy structure allows the user to explore different aspects of the data and it arranges the attributes in a way that makes it easier for the user to navigate the attribute space.

Creating a taxonomy, for example in biology or engineering, is usually done by domain experts and evolves over time. Conversely, our objective is to create a visual analytics tool that allows mainstream users to construct a taxonomy for the attributes in real-world multivariate data. To achieve this we need to model the two inherent features of a taxonomy 1) a hierarchical grouping system to construct the tree structure and 2) labels for the internal nodes of the tree to facilitate navigation. Generating a taxonomy for a multivariate dataset presents the following three challenges:

- **Flexibility**, since there are several ways to construct a taxonomy and since the final outcome depends on the user's preferences we need to design a tool that is flexible enough to allow the user to shape the taxonomy in the preferred way.
- **Labeling**, labeling the internal nodes is a difficult task. Given a group of attributes the user may find it difficult to think of a word that can capture the concept expressed in that group. We will borrow ideas from hypernym generation research to help the user find labels for the internal nodes (a hypernym is a word whose meaning includes the meaning of other words, e.g. "animal" is a hypernym of "lion").
- **Tree Structure**, users must be enabled to construct a hierarchical structure that is semantically consistent i.e. there should be a common theme in the groups created by the hierarchal structure (the semantic meaning of an attribute is expressed by its label). The semantic structure allows the user to properly label the internal nodes.

There are various data-centric similarity metrics that can be used to create a hierarchical grouping, such as Euclidean distance, correlation distance etc. However, it can be difficult to label the internal nodes if we use only data-centric techniques because these techniques are agnostic to the semantics of the data. Therefore, they may group attributes that have different meanings and are not semantically consistent which is required for a meaningful hierarchy.

To create groups that are semantically consistent we need to find the semantic distance between two attributes. We use a neural network based word embedding system to model the semantic distance between different attributes. However, the semantic space alone is unable to capture the relationship between the data which is the primary objective of the taxonomy. The challenge is therefore to design a framework that allows a human user to help in the process and effectively combine the data and the semantic aspects to create a semantically consistent hierarchical structure. This motivated us to follow a visual analytics approach that appeals to humans soliciting their wisdom and commonsense to merge the two spaces.

[1] score, academic, faculty, tuition, housing, population, income, safety, transport, location, nightlife, dining, weather, athletic

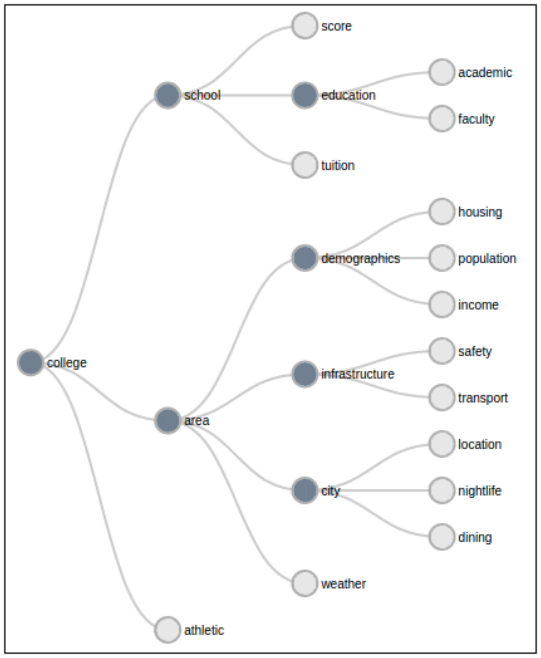


Figure 2: Taxonomy of the university dataset. The attributes of the dataset are at the leaves of the taxonomy.

In order to create a meaningful hierarchical grouping at a given level, we need to find a redundancy measure for each pair of attributes. Redundant attributes are those that vary together in some sense. We will use both a data centric and a semantic centric distance to estimate the levels of redundancy between a pair of attributes. Then, having quantized the levels of redundancy and uniqueness for all pairs we can construct a dendrogram of these attributes by ways of continually lowering the threshold required for joining inner or leaf nodes of the emerging hierarchy. For added manageability, the dendrogram can be coarsened by pooling some of the nodes into groups. This will flatten the depth of the hierarchy making it easier to explore the data, but at the cost of specificity.

The paper is structured as follows. Section 2 presents related work. Section 3 discusses the theoretical underpinnings of the methods used to construct the taxonomy. Section 4 shows how our visual tool can be used to construct a taxonomy. Section 5 describes a usage scenario. Section 6 discusses both design and results of a user study we conducted with our visual tool. Section 7 ends with conclusions.

2 Background

Taxonomies are essential for many semantic-based tasks such as content organization, guided navigation etc. There has been an ample amount of research on automated taxonomy generation. Automatic taxonomy generation involves three processes: concept mining, concept relation discovery, and concept hierarchy building. Taxonomies generated in that way are mostly built for a certain text where the algorithm first identifies the key concepts of the text and then finds a hierarchical relationship between these concepts. Similar word contexts are used to aid the discovery of a relation between concepts [39]. Finally, the different concepts are arranged into a hierarchical structure. Word embedding is a popular method for discovering the hierarchal structure of concepts [11, 15, 39].

2.1 Word Embeddings

An essential component of our system is the language model. Vector space models can be used to represent words as vectors in high dimensional space. Various models have been suggested that learn word

embeddings from a large text corpus. The central idea behind these models is that words with similar context have similar meaning. They use the context of a word to map it to vector space. The position of a word in vector space represents the context in which the word was used and the distance between words represents the similarity of their contexts. Therefore, words that are near in vector space will have a similar meaning. In its simplest form, the context of a word can be captured by using the raw co-occurrence counts of the word and context items. However, using a raw co-occurrence matrix is prohibitively expensive in terms of the space and the computational power required [9].

The solution is to use neural networks to map concepts to continuous space. This idea can be traced back to Hinton [13, 14]. Neural networks have been proven to be much more efficient at learning language models. Bengio et al. [3] presented a neural language model for learning word embeddings. Their model outperformed the traditional n-gram based techniques. Various methods have been proposed that scale and speed up large neural models [32, 37]. Mikolov et al. [28] proposed skip-gram models for learning word embeddings and demonstrated that these models have the capacity to learn linguistic patterns as linear relationships between vectors [29, 30]. Tsuboi et al. [49] incorporated word2vec embeddings with GloVe embeddings into a Part of Speech (POS) tagging system and show that it produces better results compared to the individual systems. These embeddings have been used for various Natural Language Processing tasks such as named entity recognition (NER), information retrieval [25], word sense related tasks [27] and many others. Appendix A provides more detail on skip-grams.

WordNet

WordNet [31] is a large lexical database of English in which words are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. WordNet provides various relationships between words. Some of the relationships which will be mentioned in our paper are defined as follows:

- **Synsets** or synonym set is a group of similar words.
- **Lemma**, synonyms contained within a synset are called lemmas.
- **Meronym**, a word that denotes part of something but is used to refer to the whole, e.g., the word 'wheels' when referring to a car.

The network structure of WordNet can also be used to quantify the distance between words. WordNet provides various similarity metrics such as the Wu-Palmer similarity which returns a score denoting how similar two word senses are based on the depth of the two words in the taxonomy and that of their Least Common Subsumer (most specific ancestor node). WordNet has many other similarity measures as well, however, we found that the results are not comparable to the embeddings learned from a neural network [50].

2.2 Semantic Hierarchies and the Derivation of Hypernyms

The taxonomy in Figure 2 shows the semantic hierarchy between the labels of the tree. In a semantic hierarchy, the semantic field of a node is included within that of its parent word's semantic field. There are various ways in which this relation can be captured, such as hypernym, meronym etc. Since there is a high amount of research on automatically extracting hypernyms we will mostly focus on these. The words 'dog; and 'canine' have a hyponym-hypernym ("is-a") relation where 'dog' is the hyponym of 'canine' and 'canine' is the hypernym of 'dog'.

WordNet [31] is one of the most widely used lexical resources in computer science. While it captures various relationships between words, it is nevertheless rather limited. On the other hand, many researchers have also attempted to automatically extract these relationships from text. There are two main approaches for hypernym-hyponym detection – *path-based* and *distributional*. We describe these in more detail in the following.

2.2.1 Path-based hypernym derivation

The path-based approach to detecting hyponym-hypernymy relation uses the lexico-syntactic paths that connect the joint occurrences of word pairs in a large corpus. Hearst [12] uses frequently occurring lexical and syntactic patterns in a large text corpus to find hypernyms. Snow et al. [45] use features extracted from parse trees to find hypernyms. Their methods rely on the accurate construction of parse trees and the precision of automatically generated patterns. A limitation of this method is that the paths may vary at the lexical level, and as a result, the recall is low for these methods. For this reason, Nakashole et al. [34] added a generalized version where for each path a subset of the words is replaced by Part of Speech tags, their ontological type or wild cards. Their method increased recall while maintaining precision. Suchanek et al. [48] extracted an ontology from Wikipedia and WordNet.

2.2.2 Distribution-based hypernym derivation

In these methods, the hyponym-hypernym relation is based on the distributional representation of the words. Here the distributional representation of a word captures the context within which that word occurs. These methods assume that the context of the hypernym will be broader than the context of the hyponym. Kotlerman et al. [20] propose a directional distributional measure to infer hypernym–hyponym relations based on the standard IR Average Precision evaluation measure. Fu et al. [10] propose a method for constructing semantic hierarchies using word embeddings. They propose that hypernym-hyponyms pairs have similar semantic properties as the linguistic regularities discovered by Mikolov et al. [29], for example, v(queen)-v(king) $\approx$ v(woman)-v(man). Words can be projected to their hypernyms based on a uniform transition matrix. That is, given a word x and its hypernym y a transition matrix ϕ exists such that $y = \phi x$. Other methods use difference concatenation and dot products [42, 52]. Shwartz et al. [44] propose a method that pairs path-based methods and distribution-based methods.

Other techniques have been proposed that use large knowledge graphs to build taxonomies [40]. Ristoski et al. [39] combine class labels with vector space embeddings to build the taxonomy. Their method is based on the assumption that instances of a more specific class should be positioned closer to each other on average than instances of a broader class. Ponzetto et al. [36] combine network connectivity with lexico-syntactic patterns to construct the taxonomy. However, both of these techniques and others of that nature are dependent on labels and network connectivity and therefore may not be able to capture relationships for which data is not available. Another approach is to use mass collaboration amongst users of an ontology [38].

2.3 Hierarchy Visualization

Browsing hierarchical structures to make sense of their content and organization is an essential activity. Schulz et al. [43] present a survey of hierarchy visualization techniques. The most common paradigms for visualizing hierarchies are tree node diagrams [33], tree maps [17] and icicle plots [19]. There exist many variations of hierarchy visualizations. Researchers have experimented with 3D designs [1]. Turo et al. [51] use 2D representation for inner nodes and 3D representation for the leaves. Designers have also experimented with different node designs [2, 35]. Different metaphors are used to capture a hierarchical relationship:

- **Edges**, edges are used to link nodes. The node-link tree is an example of this technique [24, 33].
- **Proximity**, a node is placed next to a larger node to represent a link between nodes [4].
- **Indentation**, indentation is used to represent the relationship. This technique is omnipresent in graphical file browsers [6].
- **Containment**, in this metaphor the lower level node is contained within the higher level node [41].

3 Theoretical Underpinnings

The input to our system is a multivariate dataset. Each element (row) of the data is composed of multiple attributes (columns). Each attribute

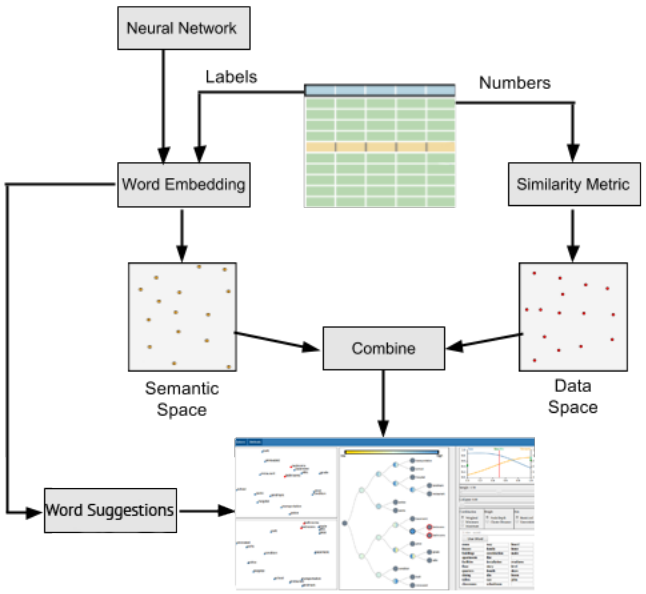


Figure 3: An overview of our system. The input to the system is a multivariate dataset which is used to generate a semantic space and a data space. The two spaces are then combined to create a taxonomy. Word suggestions are generated to help the user label the nodes of the hierarchy.

in the dataset is defined using a text label. To generate a tree that can allow meaningful labeling of the internal nodes we need to quantify the distance between the "semantics" and the "data" of different attributes. The semantic distance represents the difference in the meaning expressed by the attributes. The meaning of an attribute is inferred using the attribute label.

In order to quantify the semantic distance between different variables a neural network is trained using a large text corpus to map words to continuous vector space. We use this vector space to get the word embeddings for the attribute names. These embeddings are then used to construct the semantic space and to suggest words suited for labeling the internal nodes. On the other hand, the data distance represents the difference in numeric data. We use a similarity metric on the numeric part of the dataset to construct the data space. The two different spaces are then combined.

We visualize the combined space using a dendrogram structure. A dendrogram is a tree diagram frequently used to illustrate the arrangement of the clusters produced by hierarchical clustering (see Figure 1c). The user is given various options by which to carve and label the hierarchical tree and convert it into a taxonomy. An overview of our framework is given in Figure 3. More detailed explanations of the different parts are given in the following.

3.1 Data Space

The data space defines how the different attributes are related to each other numerically. We represent the data space using a distance matrix M, which has n x n dimensions with n being the number of attributes in the dataset. A cell $c_{i,j}$ in the matrix represents the distance between attribute i and j. We assume that the distance of an attribute to itself is 0 and that the matrix is symmetric. The user can choose a similarity metric such as correlation, mutual information etc. to represent the data space.

The data space can help capture relationships that may be difficult to attain using semantics alone. For example, a dataset on animals[2] can have attributes that relate to the class of an animal e.g. mammal, reptile etc. "Hair" and "Milk" are predominantly mammalian attributes, but they do not have a strong semantic relationship. However, the data shows that they are very strongly correlated. Therefore, the data space can be helpful in finding relations between the attributes.

3.2 Semantic Space

Like the data space the semantic space is defined using a distance matrix M. A cell $c_{i,j}$ in the matrix represents the distance between attributes i and j. The distance between two attributes is defined as the cosine distance between word embeddings. To learn the word embeddings we use a skip-gram neural network [29]. The objective of our model is to return similar results for two words that have similar context (defined by nearby words). For example, we expect synonyms like "good" and "excellent" to have similar context. The context of a word is defined as the list of words inside a window. For a window of size 2 the context of a word w_t would be $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$. Our training objective is to maximize the probability of words $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ appearing in the context of word w_t.

In our implementation, we use the gensim [3] implementation of skip-gram and use the Wikipedia corpus to train the neural network. The context window size is set to 5 and the size of the word embedding space dimension d is 128. We use a minimum word count of 100. Any word that has a frequency of less than 100 is removed from the vocabulary. To estimate the embeddings of a phrase or of multiple words we take the average of the words in the text. This is a commonly used method and has been shown to be competitive when compared with other techniques [22, 46].

Word embeddings learned using neural networks suffer from the problem of having a conflation of word senses – a word sense is a meaning of the word. Each word is represented using only one vector. If a word has many meanings the vector representation of the word will be the union over all the different meanings of the word. On the other hand, it is also possible that there are not enough samples of the word in the text corpus to learn the embedding of the word properly. Due to any of these circumstances, the semantic space will be potentially inaccurate. Another limitation is the model's vocabulary. Words that are not in the training corpus will not be part of the model and so we may not be able to use them. In addition, the labels are also expected to be not more than a few words long.

Acronyms and jargon words can be both permissible attributes and valid inner node labels if they are part of the corpus used to train the neural model. Some acronym and jargon words will be more common than others. For example, FAQ, LOL, and 9-to-5 are much more part of everyday English language and corpuses than BP (medical jargon for blood pressure) and SCOTUS (Supreme Court of the United States). If the corpus contains a wide selection of medical literature – to support a taxonomy of medical attributes – or it spans a wide gamut of political news articles – to support a taxonomy of data that contain political aspects – then our framework will be applicable to these use cases. At the heart of the semantic analysis is a neural network that needs to be trained for the task at hand. In that sense, it is not unlike an educated human who is endowed with background knowledge of specific acronym and jargon words commonplace in the human's area of expertise.

3.3 Combining Semantic Space and Data Space

Once the semantic space and the data space have been constructed the next aim is to integrate the two spaces. During our research we found that there is typically some correlation between the data space and semantic space. However, it is also possible for the two spaces to be completely contradictory. In general, the data and semantic space may conflict in some parts and corroborate in others. In some cases, the data space may even point the semantic space into the right direction.

Merging the data space with the semantic space has two advantages. Firstly, certain relationships are better captured in the data space (an example of this is given in subsection 3.1). Secondly, any data driven processes that may be performed on the subspace defined by the internal

[2] http://archive.ics.uci.edu/ml/datasets/zoo

[3] https://github.com/RaRe-Technologies/gensim

nodes, such as dimension reduction, will benefit from having a subspace that is more consistent. Thus, allowing users to merge the two spaces in a way that captures the essence of both is desirable.

The data space and the semantic space are represented using a distance matrix. To merge the two spaces we use a weighted average, minimum, or maximum operation. Balancing a layout according to two possibly conflicting aspects by ways of a weighting function has been a common practice. A recent example is the approach proposed by Martins et al. [26] who used linear weighting to smoothly arbitrate two possibly conflicting aspects with respect to node neighborhoods in a 2D layout, namely the associations coded by a graph imposed on the nodes and the associations coded by node vector similarity.

Cophenetic Correlation

To give the user a clear understanding of the compromise between semantic and data space we use the cophenetic correlation. The cophenetic correlation [47] is the correlation between the distance obtained from the dendrogram and the original distance (semantic/data distance). It is a measure of how faithfully a dendrogram structure captures the pairwise distances between the original data points. Since a dendrogram is the basic structure for the taxonomy we are building (see Figure 1(c)) the cophenetic correlation is a natural choice for estimating how distant the taxonomy is from the original information spaces. The cophenetic distance is defined as follows:

$$c = \frac{\sum_{i<j}(x(i,j)-\bar{x})(t(i,j)-\bar{t})}{\sqrt{[\sum_{i<j}(x(i,j)-\bar{x})^2][\sum_{i<j}(t(i,j)-\bar{t})^2]}}. \quad (1)$$

Here $x(i,j)$ is the distance between elements i and j in the native information space (data space or semantic space), while $t(i,j)$ is the distance between elements i and j in the dendrogram, which is equal to the height of the lowest common ancestor. $\bar{x}$ is the average of all $x(i,j)$ and $\bar{t}$ is the average of all $t(i,j)$. Figure 1(d) visualizes the cophenetic correlation. The yellow and blue lines represent the change in cophenetic correlation with respect to the semantic space and the data space, respectively. The line chart is created by merging the two spaces at different weights and calculating the cophenetic correlation.

3.4 Automated Word Suggestions

As mentioned, an important aspect of the hierarchy we wish to construct is the labeling of the internal nodes. To label an internal node we want to find a word that captures the meaning expressed by all of its descendants. To be more precise we want to find a word whose *semantic* meaning best captures the *semantic* meaning of all of its descendants. We have intentionally kept this definition vague because there can be a number of relations that might be used to create a hierarchy whereby the relation need not be limited to hypernyms. Nodes can share a "part of" relation, 'engine' is part of a 'car' or a subcategory relation, 'man' is a subcategory of 'human'. We will call these words related-words to distinguish them from hypernyms, meronyms etc. Ideally, we would want to automatically find the best related-word but this can be challenging for various reasons:

- The natural language processing algorithms developed so far are not good enough to capture the semantic space perfectly.
- There might be multiple words that can be used to label the internal nodes. For example, for the word 'New York', the terms 'city' and 'USA' are both acceptable as each represents a different aspect of the word.
- Words can be used in multiple senses, However, as discussed in the previous section the different senses are difficult to model algorithmically and resolve automatically.
- We wish to suggest a word that can be representative for multiple words, but so far algorithms for finding hypernyms only work for one word.

Bringing the human in the loop

Since automatically generated results are not sufficient for our needs we use them to aid the user in finding the appropriate labels. We have designed an interface that presents the user a selection of related words. From this list, the user can select the word they feel best portrays the hierarchical structure. Keeping a human in the loop for labeling is not a new idea. Gupta et al. [11] use human assistance in naming clusters of concepts. For naming, they check whether some elements of the concept clusters are present in WordNet [31] and propose the name to the user. Then they check the corpus for so-called Hearst patterns (lexico-syntactic patterns for recognizing hyponyms). They note, however, that the method is only applicable to high-level concepts. Kandogan et al. [18] automatically identify visual features by detecting clusters, outliers and trends, and annotate them by finding distinguishing attributes for that feature. Contextifier [16] produces annotated visualizations of stock behavior using news articles about a company. The system described by Bryan et al. [5] helps users in finding points of interest and provides a workflow for annotating the data.

To find related words our assumption is that for a set of words the related-word is a word that can be used in a similar context. We will be giving preference to the word that has the most contextual overlap with the words in the set. Our method is based on the Distributional Inclusion Hypothesis (DIH) according to which the context of a narrow term is also shared by the broad term. We use the vectors learned using the neural network to estimate the overlap. If a vector representing word u is semantically narrower than a vector representing word v, then a significant number of vector weights of u are included in the vector weights of v as well [52]. In our tool, we implement a variation of the degree of entailment measure introduced by Lenci and Benotto [23]. Lenci and Benotto show that this method outperforms other distribution-based methods. We also give more importance to words that have a higher frequency by multiplying the degree of entailment measure by the logarithm of the frequency of the word. Appendix B gives more detail on the background and mechanisms of these concepts.

4 Constructing the Taxonomy

The objective of Taxonomizer is to aid the user in constructing a valid taxonomy for multivariate datasets. Taxonomizer consists of an interlinked dashboard of two multivariate visualization displays, an interactive hierarchy editor, a curve-based mixing board, and a word selection interface (see Figure 1). In this section, we will explain both the design rationale and the use of the dashboard's components by ways of a running example – a dataset on the housing available in Kings County with attributes related to the houses [4]. Let us assume a website that allows visitors to view choropleth maps of the Kings County real estate, one attribute at a time. Given the many attributes available navigating the long unordered list can be tedious. To make this task easier, a first design goal for the Taxonomizer is to construct a hierarchy that groups the attributes into themes or aspects that can be further explored by visiting their constituents. A second design goal is to ensure that attributes in the same theme are similar in their data so that there is not a huge change in the choropleth map when the user changes to a different attribute in the same group. To create the taxonomy for the dataset the user opens Taxonomizer and then uploads the data to the system.

4.1 Data and Semantic Space

Taxonomizer uses the meta-data learned from the neural embedding (see Section 3.2) to find the vector representation of each attribute. It then uses the cosine distance to construct the semantic space. Figure 4(a) shows the scatter plot representation of the semantic space. The scatter plots are generated by using the Non-metric Multidimensional Scaling (NMDS) [21] algorithm. We used NMDS because it gives us the flexibility to add new points to the system. The semantic space

[4] The attributes of the Kings County dataset are: price, bedrooms, bathrooms, condition, grade, attic, basement, built, renovated, transportation, landmark, restaurant, hospital, police, parks, school.

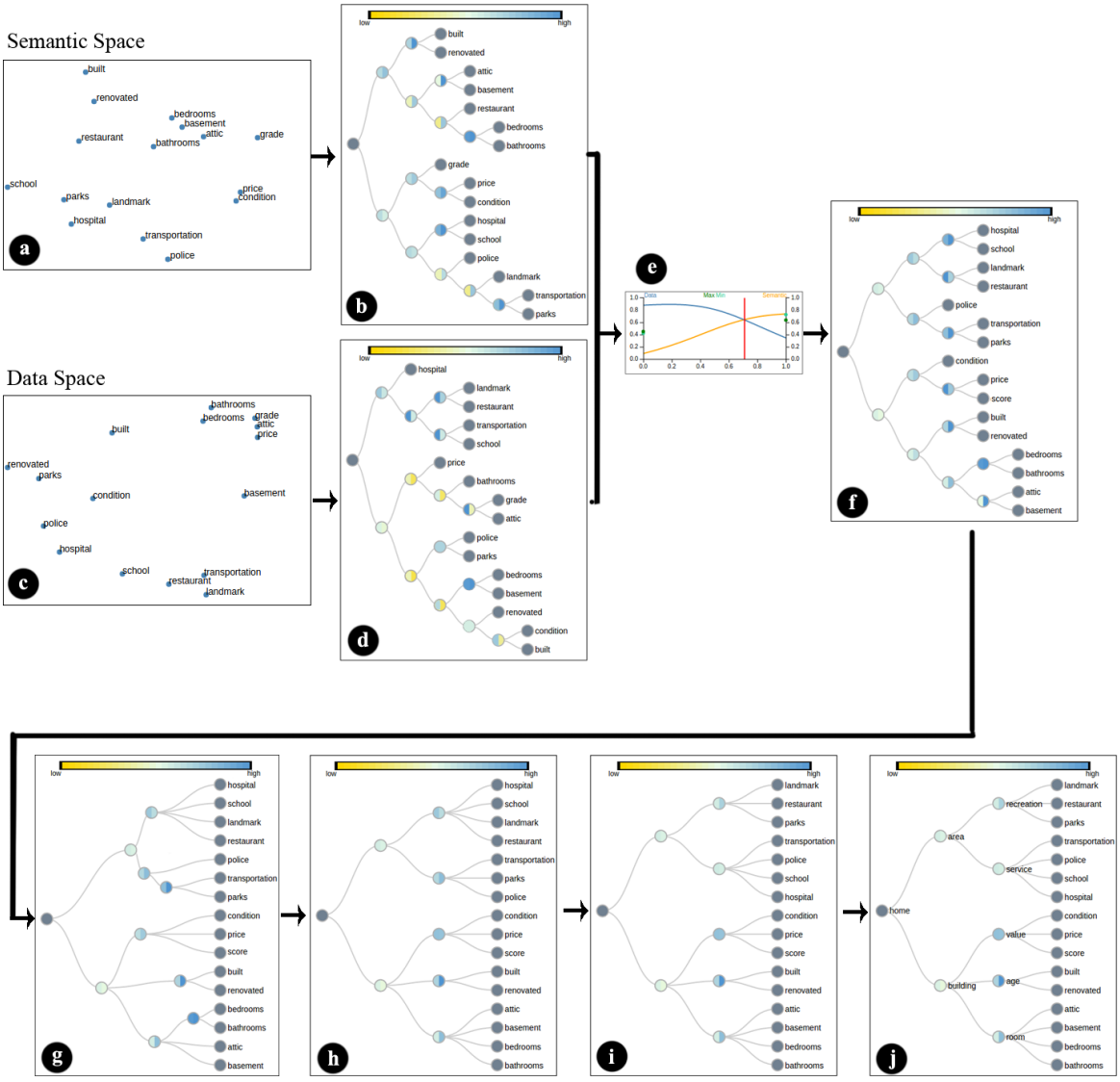


Figure 4: The workflow for the Kings County housing dataset. (a) Semantic space. (b) Taxonomy based on the semantic space. (c) Data space. (d) Taxonomy based on the data space. (e) Cophenetic plot. (f-j) Evolution of the taxonomy.

captures the semantic relations between the features – words with similar meaning locate in close proximity. For example, 'basement', 'attic', 'bedrooms' and 'bathrooms' are all close together. Other words like 'school', 'parks, 'landmarks' and 'hospital' also seem to be spread out in an appropriate manner.

Figure 4(c) shows the data space. The data space is created using the correlation distance. We see that 'price' and 'attic' are strongly correlated but semantically they are very different. Similarly, there are other discrepancies between the semantic space and the data space, for example, 'basement' is further away from 'attic', 'bedrooms' and 'bathrooms'. 'Transportation', 'landmark' and 'restaurants' are closer together.

Figure 4(b,d) shows the taxonomy structure of the semantic space and the data space, respectively. The taxonomy structure is based on a dendrogram of the distance matrices. The color of each node in the taxonomy represents the quality of the grouping at that node. The left side of the node shows the quality of grouping in the data space and the right side of the node shows the quality of grouping in the semantic space. We use the Dunn Index to estimate the quality of a grouping. The Dunn Index is a metric for evaluating clustering quality. It is the ratio of the smallest distance between observations not in the same cluster to the largest intra-cluster distance. To be more precise it is defined as follows:

$$DI_m = \frac{min_{i\epsilon m, j\notin m}\,|i,j|}{max_{i\epsilon m, j\epsilon k}\,|i,k|}. \tag{2}$$

To calculate the Dunn Index for a node in the hierarchy we treat all the attributes represented by the child leaf nodes of that node as one cluster and the remaining attributes as the second cluster. The Dunn Index is undefined for clusters with a single entity and clusters that contain the entire set. Therefore, the Dunn index is undefined for the leaves and the root of the tree, and thus we have colored these gray. When the user modifies the structure of the taxonomy the Dunn Index values will act as a guide to help him/her understand how consistent the tree structure is to the semantic and data space. Figure 4(b) is based on the semantic space, therefore, the right side of the nodes show higher Dunn Index values, while the opposite is true for Figure 4(d). It is

noticeable that the semantic side of the nodes in Figure 4(d) shows low values. This is because some of the features that are grouped together are not semantically related. For example, 'grade' and 'attic' are part of the same group.

4.2 Taxonomy View

The construction of the taxonomy is the fundamental objective of Taxonomizer. We use a tree structure to visualize the taxonomy because it is the most common method to visualize a taxonomy. Scalability is an issue with tree structures, however, the problem can be mitigated by allowing the user to collapse parts of the tree.

We start by creating a very simple binary tree structure where the inner nodes are unlabeled. This binary tree structure is constructed using a hierarchical clustering scheme on a combination of the data and semantic space. The data and semantic spaces are each represented as a distance matrix and can be combined in multiple ways. The user can take a minimum or maximum of two corresponding distances or take a weighted average of the two distance matrices. By default, we use a weighted average with equal weights. Initially, the user sees the default binary tree structure. Various tools are available to chisel the structure of the tree according to the user's preferences, knowledge, and requirements.

Taxonomizer provides three ways by which the user can modify the structure of the taxonomy: (1) merging data space and semantic space, (2) collapsing the taxonomy, and (3) joining nodes in the taxonomy. We have also added an undo feature so that the user can undo any changes made to the tree. The following subsections explain these operations in closer detail.

4.2.1 Merging Data and Semantic Space

There are three ways to merge the data and semantic space.

Weighted Average

The user can take a weighted average of the two spaces by using a slider interface. This method merges the two spaces by taking an element-wise weighted average of the two distance matrices. By adjusting the weight the user is making a compromise between the two information spaces and he can decide what weight to use according to his preferences and needs. We use the cophenetic distance to measure the information lost in creating the tree structure. It gives the user a concrete estimate of the gains and losses of the different weighting schemes. How the cophenetic correlation changes with weight is plotted in a line chart (see Figure 4 (e)).

Minimum/ Maximum

The data space and the semantic space are merged by taking a pointwise minimum and maximum between the two distance matrices. Similar to the weighted average scheme we use the cophenetic correlation to determine how the taxonomy compares to the original spaces. The cophenetic correlation for the minimum and maximum merge are represented using light and dark green circles, respectively, on the cophenetic graph.

A user who is more interested in the semantic space may choose a weight of 0.7. The line for the semantic distance flattens out beyond 0.7, meaning that there is little to gain. Whereas, the line for the data plot shows that there will be significant losses (see Figure 4(e)). If the user wants to conserve the data space he might decide to use a weight of 0.5. The cophenetic distance graph allows the user to visualize the compromise between the data space and the semantic space. In the case shown here, the user decides to use a weighted average of 0.7. The taxonomy created by this weighted average is pictured in Figure 4(f). The overall blue coloring of the nodes shows that both the semantic space and the data space have been captured quite well. The taxonomy seems to have a good overall structure. We observe that the attributes directly related to the structure of the house are in the lower half of the taxonomy while the attributes related to the locality are on the upper side.

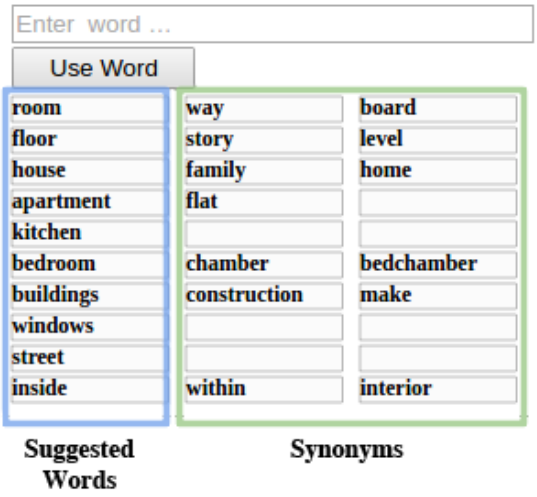


Figure 5: The word suggestions panel. Shows the list of suggestions for 'attic', 'bedroom', 'bathroom' and 'basement'.

4.2.2 Collapse

The taxonomy starts as a binary structure. The collapse feature allows the user to merge child nodes with their parents if their height difference is within a given threshold. The child node is removed and the parent node takes over the children of the child node. The collapse threshold can be controlled by adjusting a slider. Collapsing the taxonomy reduces the amount of detail represented. We have two layout schemes, depth-based (Figure 4(f)) and height-based (Figure 4(g)). In the depth-based scheme, the x-coordinates of the node are determined by the height of the node in the tree. Conversely, in the height-based scheme, the x-coordinates of the node are determined by the distance between the clusters.

We added these two different layouts because the depth-based layout is better at spacing out the different nodes giving the user a clear picture of the structure of the tree. On the other hand, the height-based layout shows the actual height of the nodes in the dendrogram. This view is useful when determining the collapse threshold for the tree. The user sets a collapse threshold of 0.2 to reduce the height of the tree (see (Figure 4(g)).

4.2.3 Join Nodes

To further chisel the structure of the taxonomy the user has the option to join nodes together. He can join two nodes by dragging one node to another. We remove the node that is being dragged and the other node takes over the children of the dragged node. When the user starts to drag a node the nearest node is highlighted by changing its color to red. This facilitates the merging of nodes.

There are two modes for merging – restricted and unrestricted. In the restricted mode the user is only allowed to merge the node with either its parent or its sibling. In the unrestricted mode, the user is allowed to merge a node with any node other than its children. The restricted node is added to ensure that the user does not stray too far from the original structure of the tree.

We see this feature at work in Figure 4(h) where we reduced the depth of the taxonomy using the restricted join features. A user might notice that 'parks' is in the same group as 'police' and 'transport' and might feel that all entertainment-related attributes ('landmarks' and 'restaurants') should be in the same group. The unrestricted join feature can be used to fix this (Figure 4(i)). The color of the affected nodes changes to signify the effects of the alterations. We observe that one node shows better semantic consistency than before while the other shows less semantic consistency. Both nodes show a loss in data consistency.

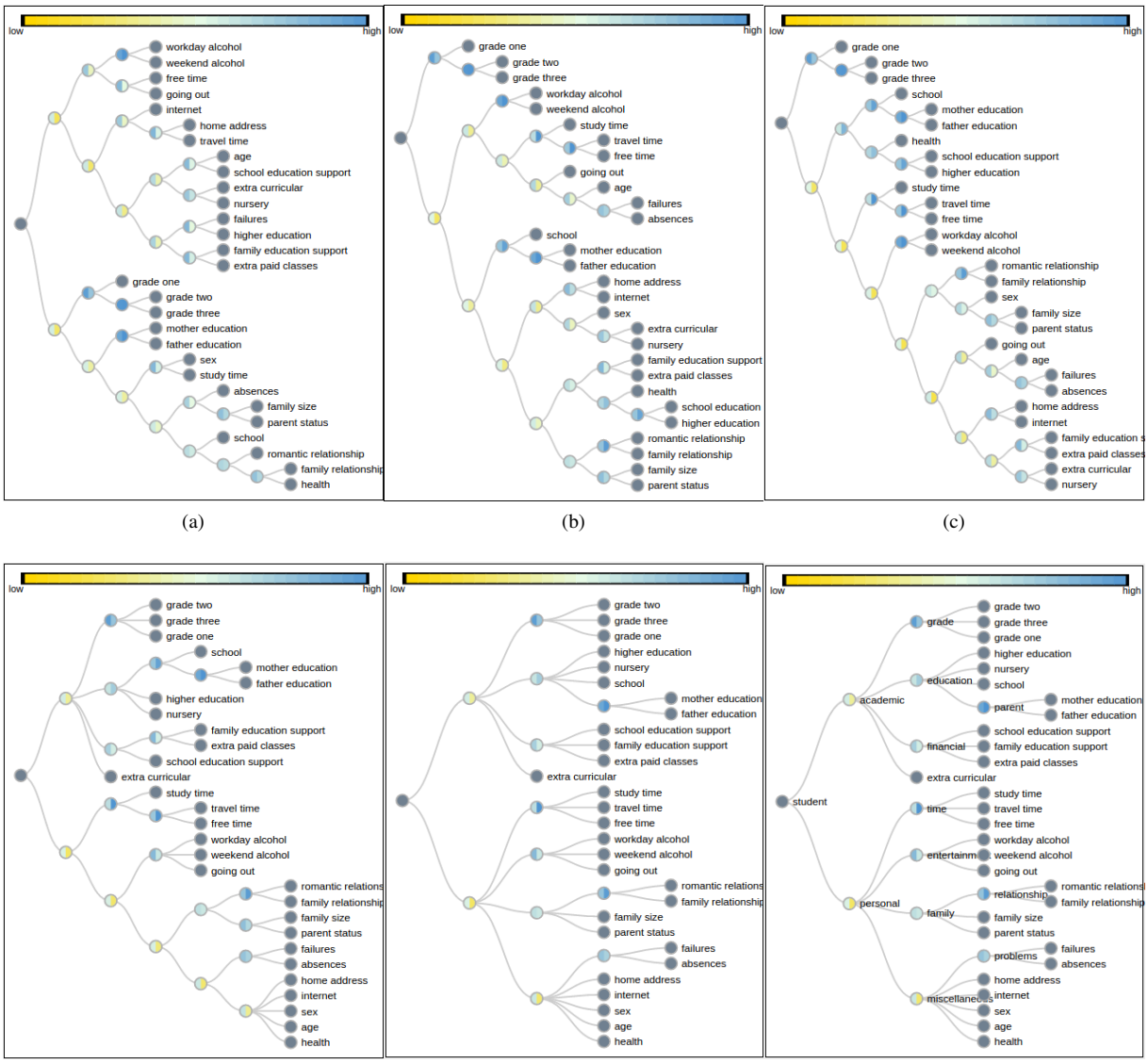


Figure 6: The evolution of the taxonomy for the student performance dataset. (a) Dendrogram of the data space. (b) Dendrogram of the semantic space. (c) Structure after weighted average merge scheme, with weight 0.31. (d) Sculpting the structure using the join features. (e) Reducing the height of the tree. (f) The fully labeled taxonomy.

4.3 Node Labeling

Having adjusted the hierarchy the final task is to perform the labeling of the internal nodes. The user can add labels by double-clicking on the nodes. To label an internal node Taxonomizer provides the user with a list of suggestions that can be used for this purpose. In order to find words that can make good suggestions we use the degree of entailment method [23] (see section 3.4).

During our experiments, we discovered that adding synonyms to the list can make it more helpful. Figure 5 shows the list of suggestions for 'attic', 'bedroom', 'bathroom' and 'basement'. The first column shows the top 10 suggested words retrieved using the Degree of Entailment method and the second and third column show the synonyms for the words in the first column (if available). The synonyms are obtained using WordNet. We use the synonyms with the highest word frequency. Finding an appropriate label is a difficult task and it is possible that the word does not exist in the list of suggestions. In this case, the user is provided with a text box to add a word of their own choice.

Figure 4(j)) The fully labeled taxonomy as generated by one of our user study participants. In this example, one label was provided by the user ('recreation') and one synonym was used ('home'). The remaining labels were all suggestions retrieved by the Degree of Entailment method.

5 Usage Scenario

We follow Jane who has compiled some data on student performance in secondary education [5]. She wants to create a web tool that allows users to visualize the data and decides to use Taxonomizer to create a hierarchical structure where users can explore the data at various levels of granularity. Jane begins by loading the dataset into the program. As a first step she visualizes her data using scatterplots to reveal correlations and she also does some brushing for selection operations. Then she opens her data in Taxonomizer and uses correlation as the distance metric for the data space. The hierarchical structure generated using the data space can be seen in Figure 6(a). We can see that some parts of the hierarchy are semantically consistent. Figure 6(b) shows the

[5] https://archive.ics.uci.edu/ml/datasets/Student+Performance

Figure 7: The results of the user survey, where the interactions of the user with the Taxonomoizer are logged. The patterns related to how the participants interact with the visual tool can be observed.

dendrogram constructed using only the semantic space. Even though the hierarchy is semantically consistent there are some deficiencies.

To improve the structure of the hierarchy Jane uses a value of 0.31 in the weighted average scheme. She uses 0.31 because up to this value the cophenetic correlation does not show any significant reduction in the semantic space. Figure 6(c) shows the hierarchical structure generated by taking a weighted average of the semantic space and the data space. We can observe that some of the deficiencies in the semantic structure have been removed. For example, the attributes related to education are closer to each other.

Jane then proceeds to condense the hierarchy using the collapse feature and the restricted join feature (see Figure 6(d)). She notices that the attributes related to education are divided into two subgroups. She uses an unrestricted join to fix this and makes other minor adjustments (e.g. the 'going out' attribute is placed in the subtree related to alcohol consumption). Then she reduces the height of the tree using the collapse and restricted join features (see Figure 6(e)). Now that her hierarchical structure is complete she clicks on the inner nodes to label them. For each node the word suggestions panel provides some suggestions of possible words for labeling it. She can use one of these suggestions or label the node herself (see Figure 6(f)).

Jane, or a person using Jane's generated labeled hierarchy, can now select any two labels in the taxonomy to view the scatter plot for those variables. For a leaf node the program selects the original values for the corresponding variable. Conversely, for an inner node the program computes the main Principal Component (PC) of the subspace formed by its leaf nodes and so determines the most representative projection for it. As the correlation gets weaker at increasing levels of the hierarchy the corresponding scatter plots will appear less correlated as a consequence. With these mechanisms in place the taxonomy structure enables users to visualize the data at any level of granularity, and do so with ease.

In the following we offer a few comments on the final structure of the secondary education taxonomy we just created (see Figure 6(f)).

- The 'nursery' attribute refers to whether a student attended preschool. Therefore, it is part of the 'education' subtree.
- The alcohol-related attributes are in the 'entertainment' subtree because they are distractions (from education). Fittingly, they are also closely connected in the data space.
- The label 'parent' is an auto-generated word. Therefore, it is singular. It is part of the 'education' subtree because it relates to the education of the parents which plays a critical role in the upbringing of a child.
- 'Extra-curricular' is a separate node in the 'academic' subtree because it is different from the other attributes, it refers to activities like cooking, karate, soccer etc.
- The 'relationship' subtree is part of the 'family' subtree because it is strongly related in the data space.
- The 'problems' subtree could have gone as a direct descendant of the 'personal subtree but the data space shows that placing it below the 'miscellaneous' node is a better fit.

6 User Study

The objective of the user study was to see how users interacted with the visual tool and if they were able to construct semantically consistent taxonomies.

6.1 Experiment Setup

The user study was divided into two parts. In the first part, the participants were asked to construct the taxonomy manually. In the second part, the participants were asked to construct the taxonomy using the visual analytics tool.

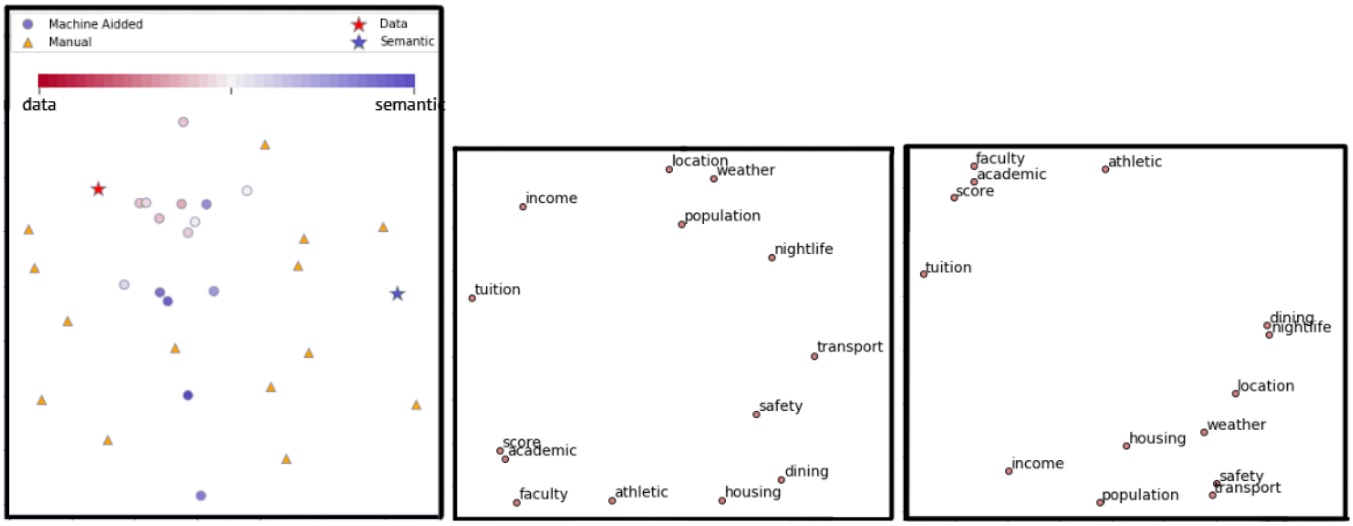
Manual Taxonomy Generation

In the first part of the user study, the participants were asked to construct a taxonomy using sticky notes as nodes. We used sticky notes to construct the hierarchy because it was easier for the participants to move nodes and alter the structure of the taxonomy. We recruited 15 participants for this experiment (12 males and 3 female) with age of 21-29. Before starting the experiment we took 5 minutes to explain the concept of a taxonomy and used a dataset to show an example of how a taxonomy can be created. The example started with the attributes of the dataset and one root node. Inner nodes were then added to the taxonomy to create more depth. The final tree had a depth of 4.

The participants were asked to construct a taxonomy for the university dataset. They were provided with the sticky notes and a pen to create and label the inner nodes. The participants did not have access to the numerical correlation values for each attribute pair. We recorded the number of times the participants moved a node. There was no time limit for the participants.

Machine-aided Taxonomy Generation

For the second experiment, we recruited 15 participants of similar demographics and age range as before. They were first given a demonstration of the visual tool where they were shown how to use its different functions using an example dataset. After the demonstration, the participants were given a dataset on which they could practice using the tool for constructing a taxonomy. Once they reached good aptitude using



(a) (b) (c)

Figure 8: (a) A scatter plot of the taxonomies generated by the participants. The circles representing machine aided taxonomies are colored according to the weight used by the participant. The red and blue stars represent trees generated using the data space and the semantic space only, respectively. (b) The scatter plot of the attributes based on the structure of the manually generated taxonomies. (c) The scatter plot of the attributes based on the structure of the taxonomies generated using the visual tool.

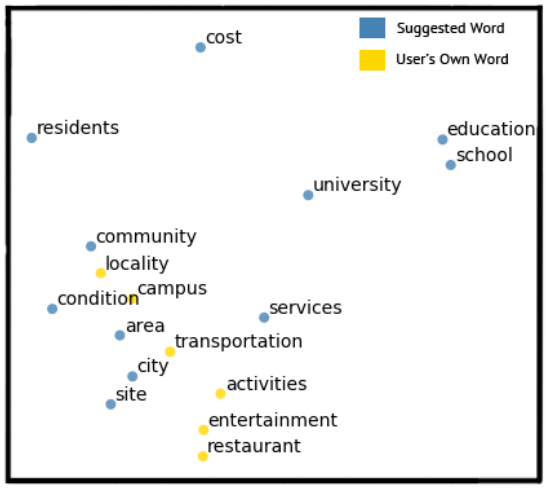


Figure 9: Scatter plot of the words used to label the inner nodes of the Taxonomies.

the tool they were given a new dataset and were asked to construct a taxonomy for that dataset. To observe how the users interacted with the tool, we logged all of the interactions made by the user in the software. We kept track of five major activities which are defined as follows:

- **Collapse**, collapsing an internal node whose height difference with their parents is within a certain threshold.
- **Join Nodes**, the node join can be restricted and unrestricted.
- **Suggested Word**, select the label for an internal node from the list of suggestions.
- **User's Own Word**, the user writes his or her own label.
- **Merge Scheme**, using a weighted average, minimum or maximum to merge the data and semantic space.

6.2 Results and Feedback

The user interactions with the visual tool are shown in Figure 7. From this table, we can see that the participants followed the pattern of first merging the data space followed by using the join operations and then adding the labels at the end. Most participants spent some time merging the data and semantic spaces. However, once they decided on how to merge the data space and the semantic space they did not adjust it again. Later on, only three participants decided to re-adjust using the merging scheme. All participants used the weighted average merging scheme. The join feature was also used by all participants and they found it very useful in customizing the structure of the taxonomy. To label the nodes, the participants used labels from the suggested words 59.0% of the time, hinting that the suggested words provide useful options for labeling the internal nodes. We discovered that the collapse option was not overly popular – it was used by only 4 participants.

The average number of interactions with the visual tool was 27.8 and the average number of nodes moved in the manual setup was 30.1. The average depth of a node in the trees constructed using the visual tool (2.67) was significantly higher in comparison with the trees constructed using the manual method (2.18). The initial binary tree computed by the software and displayed in the visual tool was larger in height compared to the starting point of the manual tree which started in a state not aided by any computational support, i.e., with all the leaf nodes directly attached to the root node. This encouraged the users to construct trees derived from the visual tool with greater depth. As a result, the taxonomies generated with the visual tool had more detail.

To compare how correctly the participants used the labels we created a co-occurrence matrix between the labels and the attributes of the dataset (we use labels that have been used at least two times). A label and attribute co-occur if that attribute is a descendant of the label node. We use the cosine distance to create a similarity matrix and visualize it using an MDS plot (see Figure 9). We can see that words with similar meanings are closer together indicating that the participants were able to find the correct words and used the labels in a consistent manner.

We compare the structures of the different taxonomies created by the participants manually and using the visual tool (see Figure 8(a) – the actual taxonomies created by the participants are given in the supplementary material of this paper). The plot is generated using MDS with cophenetic correlation as the similarity measure. The yellow triangles represent taxonomies generated with the manual procedure. Conversely, the circles represent machine aided taxonomies and are

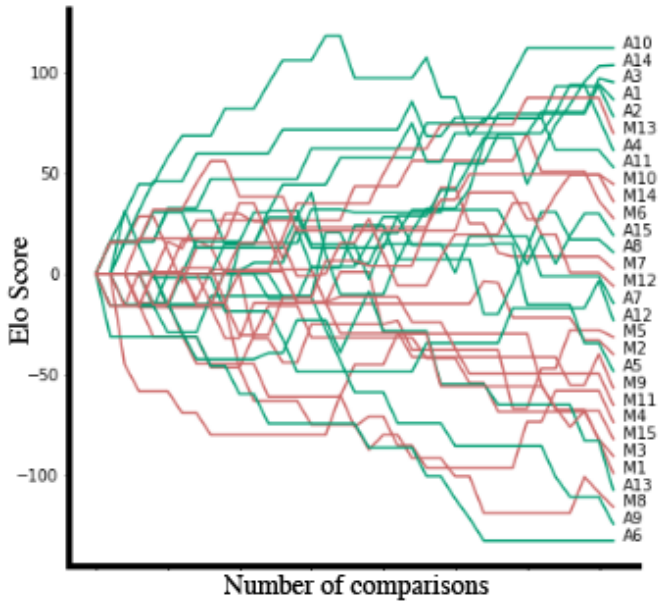


Figure 10: The Elo rankings of the different taxonomies based on the results of the rankings generated by the participants. The labels of the taxonomies are given at the end of each taxonomy. The green lines correspond to a machine aided taxonomy and red lines correspond to a manual taxonomy. All the taxonomies are given in the appendix of the paper.

colored according to the data/semantic space weight used by the participant. The red star represents the tree generated from the data space only, while the blue star denotes the tree generated from the semantic space only.

The larger spread of the triangles suggests that there is more diversity in the manually created trees. On the other hand, the still significant spread of the trees generated using the visual tool (the circles) suggests that the participants were able to create a fairly wide variety of taxonomies. This confirms that Taxonomizer is flexible enough to generate various kinds of structures from the same dataset. Figure 8(a) shows that taxonomies with similar weights (indicated by color) are closer together. It indicates that the starting point of a taxonomy played an important role in the final outcome. This observation is corroborated by participants who reported that the cophenetic correlation plot helped them identify a good starting point for the construction of the taxonomy. Further, we also observe that the trees with low weights (closer to the data space) are closer to the tree generated using the data space only (the red star). Even trees with higher semantic weight (the blue circles) are still closer to the data space-only tree than to the semantic space-only tree (the blue star). This indicates that trees generated using our visual tool tend to adhere more to the data space and so capture more of the data space similarities while still taking up a good portion of of the semantic space relationships as well.

To estimate the semantic consistency of the structures generated by the participants we created a similarity matrix between the attributes of the dataset. The similarity between two attributes is defined as the average height of the lowest common ancestor node. Nodes with the same parent will be closer to each other than nodes with the same grandparent. The similarity matrix is visualized using an MDS plot. The plot for manually generated trees and the machine-aided trees can be seen in Figure 8(b) and Figure 8(c), respectively. We observe that in both plots words with similar meanings are closer together. For example, faculty, score and academic are in close proximity. This indicates that the participants were able to construct semantically consistent trees in both experiments. But we also observe that machine-aided trees had a clearer separation of academic and area attributes which is consistent with the data.

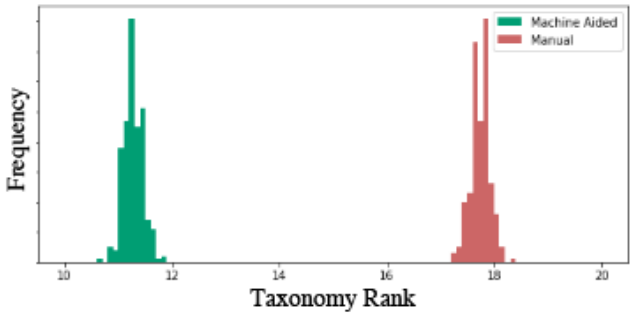


Figure 11: The distribution of the average rank for the taxonomies generated using the manual (red) and machine aided methods (green). Lower rank is better and there is very clear separation of the to distributions.

6.3 Manual vs Machine Aided Taxonomies

In the final part of our user study, we compared the taxonomies generated by the machine-aided and the manual methods. At first we presented them to a linguistics expert, but he found it challenging to compare all the taxonomies that were generated. He suggested that since the taxonomies were to be useful to humans they should be evaluated by humans (humans with no specific academic background in linguistics, that is). Following this advice we performed an experiment in which the participants were given 6 randomly selected taxonomies (3 generated using the machine aided method and 3 generated using the manual method). We recruited 15 participants for this test (11 males and 4 females). The participants were in the age range 23-31. The participants were explained the concept of a taxonomy using examples. They were then asked to rank the taxonomies from best to worst.

To combine the results of the rankings generated by the participants we converted the rankings into a set of pairwise comparisons and used the Elo ranking algorithm to calculate the final rankings. The Elo rating scheme was introduced as a chess rating system by Aprad Elo [8]. Variants of the rating scheme are still used to rate chess players. Similar to chess matches between two competing players, we use pairwise comparisons to rate different designs. Each participant-generated ranking (6 taxonomies) is converted into 15 pairwise comparisons. The taxonomy with the higher rank is then declared the winner of the comparison. We asked the participants to rank the taxonomies instead of comparing them individually because it is more engaging for the participants to rank among a set than to pick among a pair. Results of the Elo ranking are shown in Figure 10. We can see that taxonomies generated using the machine aided method (colored green) are in general rated higher than the taxonomies generated by the manual method (colored red).

The result of the Elo ranking is dependent on the order of the comparisons. This creates ambiguity in the results of the rankings. To make the results of the Elo algorithm more concrete we calculated the average rank of the manual and machine-aided taxonomies. This process was repeated multiple times with random orders of the comparisons. It generated a distribution of the manual and machine aided methods which is shown in Figure 11. We observe a clear separation of the two methods with the machine aided taxonomies having the better rank distribution.

To show that the difference between two distribution are statistically significant we disprove the null hypothesis which assumes that there is no difference between the two methods. To achieve this we performed the ANOVA test which returned a p-value of 0 which impressively shows that there is a significant difference between the rankings of the manual and machine aided methods.

7 Conclusion and Future Work

We presented Taxonomizer, a visual analytics tool that can be used to organize the attributes from a multivariate dataset into a meaningful and fully-labeled taxonomy structure. Specifically, Taxonomizer allows users to fuse two separate aspects of the data attributes – statistics and

semantics – to compose a consistent hierarchical representation of the attribute space. Our visual tool provides a high degree of interactivity and is flexible enough to create a taxonomy that fits a user's specific need and domain understanding. A user study we conducted showed that the participants were able to create deeper trees using the visual tool. The visual tool also allowed the study participants to construct trees that were closer to the data space.

However, there are some caveats. One of them is that the natural language models used to model the semantic space have some deficiencies. Also, we use a tree structure to visualize the taxonomy. Tree structures have issues with scalability in particular when the tree is deep. Another limitation is that only single word hypernyms are generated using our method. Even though these deficiencies put some limitations on the effectiveness of the system, we believe that they have been mitigated to a good extent by the tools we added to the system. Both case and users studies show that despite these adverse circumstances Taxonomizer is still able to organize A dataset's features into a consistent and meaningful labeled taxonomy.

We believe that statistical analysis of attributes of a structured data set is an exciting area of research that opens many more avenues. In future work, we would like to see how taxonomy building can be further refined. We would like to experiment with learning embedding models using a more specific corpus, for example, using only those Wikipedia articles that are related to the dataset labels. It might also be interesting to explore how the semantic space can be used for other visualization tasks such as dimensionality reduction. Finally, while in the present work we do not evaluate the goodness of the taxonomy generated by the user, in the future we would like to find ways to evaluate how mutually exclusive and unambiguous the labels are.

APPENDIX

A. The Skip-Gram Model

The skip-gram model [28] uses each word w_t as input to a log-linear classifier with continuous projection layer and predicts the context of w_t i.e. words within a certain window before and after the word w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}. Increasing the window size improves the quality of the resulting word vectors, but it also increases the computational complexity.

Given a corpus C, our objective is to learn word embeddings ϕ for a vocabulary V. We start by randomly initializing the vector representation for words. Our training objective is to maximize the probability of words w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2} appearing in the window of word w_t. To be more precise we are maximizing the following probability function:

$$Pr(w_j|w_i) = \frac{e^{w_j.w_i}}{\sum_{w_k \varepsilon V} e^{w_k.w_i}}$$

Here w_i and w_j are the vector representations for the words(w) and their contexts. The learning process is made up of multiple iterations, each iteration is called an epoch. In each epoch, we iterate over each word in V to minimize the negative log-likelihood of the context words. The function is defined as follows:

$$E = \sum_{w \varepsilon C} \sum_{j=i-m}^{i+m} -logPr(w_j|w_i)$$

Note here that the denominator of equation (7) is a summation over words from the entire vocabulary $|V|$. To make this computation feasible we limit the number of context words that must be updated in training instance. This is done using the hierarchical softmax method that uses a binary tree to represent words in the vocabulary. The vocabulary words are the leaves of the binary tree. For each leaf unit, there exists a unique path from the root to the unit; and this path is used to estimate the probability of the word represented by the leaf unit. This reduces the time complexity of this step from $O(|V|)$ to $O(log(|V|))$.

B. Finding Related Words

To find related words our assumption is that for a set of words a related-word is a word that can be used in a similar context. We will be giving preference to the word that has the most contextual overlap with the words in the set. Our method is based on the Distributional Inclusion Hypothesis (DIH) [20]. according to which the context of a narrow term is also shared by the broad term. We use the vectors learned using the neural network to estimate the overlap. If a vector representing word u is semantically narrower than a vector representing word v, then a significant number of vector weights of u are included in the vector weights of v as well.

Clark proposed the degree of entailment measure [7] which is based on DIH. It quantifies the weighted coverage of the vector of the narrower word by the vector of the broader word . The DIH metric is defined as follows:

$$DH(u,v) = \frac{\sum_f min(w_u(f), w_v(f))}{\sum_f w_u(f)}. \quad (3)$$

Here $w_u(f)$ represents the weight at dimension f for the vector of the word u. In our tool, we implement a variation of the degree of entailment measure introduced by Lenci and Benotto. which takes into account the inclusion of u into v as well as the non-inclusion of v in u. It is defined as follows:

$$invCL(u,v) = \sqrt{DH(u,v) * (1 - DH(v,u))}. \quad (4)$$

Lenci and Benotto [23] showed that this method outperforms other distribution-based methods. We also give more importance to words that have a higher frequency by multiplying the degree of entailment measure by the logarithm of the frequency of the word. To implemented the distribution based method, we used the word embeddings already learned for finding the semantic space. The distributional method is that the nearest neighbor of a word can include hypernyms, meronyms synonyms etc. making it difficult to single out a relationship. In our case however we are not interested in defining relationships, we want to identify words that share the context between multiple words.

ACKNOWLEDGMENTS

This work was supported in part by NSF grant IIS 1527200 and the Ministry of Science, ICT and Future Planning, Korea, under the "IT Consilience Creative Program (ITCCP)" supervised by NIPA. We would also like to that Prof. Jeffrey Heinz from the Stony Brook University Department of Linguistics for his advice on the final user study.

REFERENCES

[1] M. Balzer and O. Deussen. *Hierarchy based 3D visualization of large software structures*. IEEE, 2004.

[2] M. Balzer and O. Deussen. Voronoi treemaps. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pp. 49–56. IEEE, 2005.

[3] Y. Bengio and J.-S. Senécal. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 19(4):713–722, 2008.

[4] A. Bosca, D. Bonino, and P. Pellegrino. Ontosphere: more than a 3d ontology visualization tool. In *Swap*. Citeseer, 2005.

[5] C. Bryan, K.-L. Ma, and J. Woodring. Temporal summary images: An approach to narrative visualization via interactive annotation generation and placement. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):511–520, 2017.

[6] M. Burch, M. Raschke, and D. Weiskopf. Indented pixel tree plots. In *International Symposium on Visual Computing*, pp. 338–349. Springer, 2010.

[7] D. Clarke. Context-theoretic semantics for natural language: an overview. In *Proceedings of the workshop on geometrical models of natural language semantics*, pp. 112–119. Association for Computational Linguistics, 2009.

[8] A. E. Elo. *The rating of chessplayers, past and present*. Arco Pub., 1978.

[9] K. Erk. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 57–65. Association for Computational Linguistics, 2009.

[10] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu. Learning semantic hierarchies via word embeddings. In *ACL (1)*, pp. 1199–1209, 2014.

[11] N. Gupta, S. Podder, K. Annervaz, and S. Sengupta. Domain ontology induction using word embeddings. In *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*, pp. 115–119. IEEE, 2016.

[12] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pp. 539–545. Association for Computational Linguistics, 1992.

[13] G. Hinton, D. Rumelhart, and R. Williams. Learning internal representations by back-propagating errors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1, 1985.

[14] G. E. Hinton. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, vol. 1, p. 12. Amherst, MA, 1986.

[15] Z. Hu, P. Huang, Y. Deng, Y. Gao, and E. P. Xing. Entity hierarchy embedding. In *ACL (1)*, pp. 1292–1300, 2015.

[16] J. Hullman, N. Diakopoulos, and E. Adar. Contextifier: automatic generation of annotated stock visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2707–2716. ACM, 2013.

[17] B. S. Johnson. *Treemaps: visualizing hierarchical and categorical data*. PhD thesis, 1993.

[18] E. Kandogan. Just-in-time annotation of clusters, outliers, and trends in point-based data visualizations. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pp. 73–82. IEEE, 2012.

[19] B. Kleiner and J. A. Hartigan. Representing points in many dimensions by trees and castles. *Journal of the American Statistical Association*, 76(374):260–269, 1981.

[20] L. Kotlerman, I. Dagan, I. Szpektor, and M. Zhitomirsky-Geffet. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(04):359–389, 2010.

[21] J. B. Kruskal. Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, 29(2):115–129, 1964.

[22] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pp. 1188–1196, 2014.

[23] A. Lenci and G. Benotto. Identifying hypernyms in distributional semantic spaces. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pp. 75–79. Association for Computational Linguistics, 2012.

[24] S. Lohmann, S. Negru, F. Haag, and T. Ertl. Visualizing ontologies with vowl. *Semantic Web*, 7(4):399–419, 2016.

[25] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, vol. 1. Cambridge university press Cambridge, 2008.

[26] R. M. Martins, J. Kruiger, R. Minghim, A. C. Telea, and A. Kerren. Mvn-reduce: Dimensionality reduction for the visual analysis of multivariate networks. In *19th EG/VGTC Conference on Visualization (EuroVis' 17), 12-16 June 2017, Barcelona, Spain*. Eurographics-European Association for Computer Graphics, 2017.

[27] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, p. 279. Association for Computational Linguistics, 2004.

[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[29] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[30] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In *Hlt-naacl*, vol. 13, pp. 746–751, 2013.

[31] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[32] F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *Aistats*, vol. 5, pp. 246–252. Citeseer, 2005.

[33] E. Motta, P. Mulholland, S. Peroni, M. d'Aquin, J. M. Gomez-Perez, V. Mendez, and F. Zablith. A novel approach to visualizing and navigating ontologies. In *International Semantic Web Conference*, pp. 470–486. Springer, 2011.

[34] N. Nakashole, G. Weikum, and F. Suchanek. Patty: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1135–1145. Association for Computational Linguistics, 2012.

[35] K. Onak and A. Sidiropoulos. Circular partitions with applications to visualization and embeddings. In *Proceedings of the twenty-fourth annual symposium on Computational geometry*, pp. 28–37. ACM, 2008.

[36] S. P. Ponzetto and M. Strube. Deriving a large scale taxonomy from wikipedia. In *AAAI*, vol. 7, pp. 1440–1445, 2007.

[37] R. Raina, A. Madhavan, and A. Y. Ng. Large-scale deep unsupervised learning using graphics processors. In *Proceedings of the 26th annual international conference on machine learning*, pp. 873–880. ACM, 2009.

[38] M. Richardson and P. Domingos. Building large knowledge bases by mass collaboration. In *Proceedings of the 2nd international conference on Knowledge capture*, pp. 129–137. ACM, 2003.

[39] P. Ristoski, S. Faralli, S. P. Ponzetto, and H. Paulheim. Large-scale taxonomy induction using entity and word embeddings. In *Proceedings of the International Conference on Web Intelligence*, pp. 81–87. ACM, 2017.

[40] P. Ristoski and H. Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference*, pp. 498–514. Springer, 2016.

[41] W. Rivadeneira and B. B. Bederson. A study of search result clustering interfaces: Comparing textual and zoomable user interfaces. *Studies*, 21(5), 2003.

[42] S. Roller, K. Erk, and G. Boleda. Inclusive yet selective: Supervised distributional hypernymy detection. In *COLING*, pp. 1025–1036, 2014.

[43] H.-J. Schulz, S. Hadlak, and H. Schumann. The design space of implicit hierarchy visualization: A survey. *IEEE transactions on visualization and computer graphics*, 17(4):393–411, 2011.

[44] V. Shwartz, Y. Goldberg, and I. Dagan. Improving hypernymy detection with an integrated path-based and distributional method. *arXiv preprint arXiv:1603.06076*, 2016.

[45] R. Snow, D. Jurafsky, A. Y. Ng, et al. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*, vol. 17, pp. 1297–1304, 2004.

[46] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

[47] R. R. Sokal and F. J. Rohlf. The comparison of dendrograms by objective methods. *Taxon*, pp. 33–40, 1962.

[48] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3):203–217, 2008.

[49] Y. Tsuboi. Neural networks leverage corpus-wide information for part-of-speech tagging. In *EMNLP*, pp. 938–950, 2014.

[50] Y. Tsvetkov, M. Faruqui, W. Ling, G. Lample, and C. Dyer. Evaluation of word vector representations by subspace alignment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2049–2054, 2015.

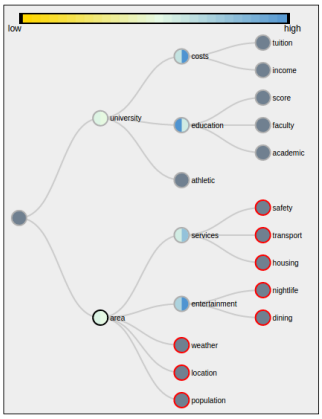
[51] D. Turo and B. Johnson. Improving the visualization of hierarchies with treemaps: design issues and experimentation. In *Visualization, 1992. Visualization'92, Proceedings., IEEE Conference on*, pp. 124–131. IEEE, 1992.

[52] J. Weeds, D. Clarke, J. Reffin, D. Weir, and B. Keller. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2249–2259. Dublin City University and Association for Computational Linguistics, 2014.

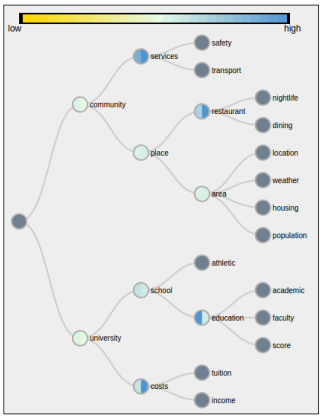
Salman Mahmood received his BSc. from Lahore University of Management Sciences (LUMS). He completed his MS from Stony Brook University. He is now a Ph.D. candidate in Computer Science at Stony Brook University, under the guidance of Prof. Klaus Mueller. His research interests include information visualization data analytics and machine learning. He received the Stony Brook Computer Science Fellowship and the IT Consilience Creative Program Scholarship.

Klaus Mueller Klaus Mueller received the PhD degree in computer science from The Ohio State University. He is currently a professor of computer science at Stony Brook University and an adjunct senior scientist at Brookhaven National Lab. His current research interests include visualization, visual analytics, data science, and medical imaging. He won the US National Science Foundation (NSF) Early CAREER award in 2001, the SUNY Chancellor's Award in 2011, and the IEEE CS Meritorious Service Certificate in 2016. He has authored more than 180 papers which were cited more than 8,000 times. He is currently an Associate Editor-in-Chief at IEEE Transactions on Visualization and Computer Graphics and a senior member of the IEEE. For more information, see http://www.cs.sunysb.edu/ mueller

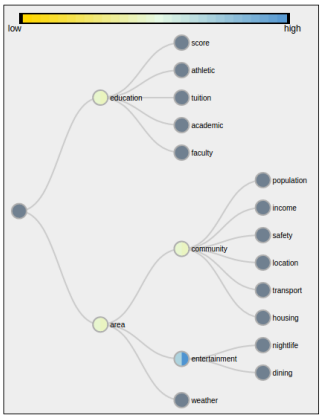
Appendix



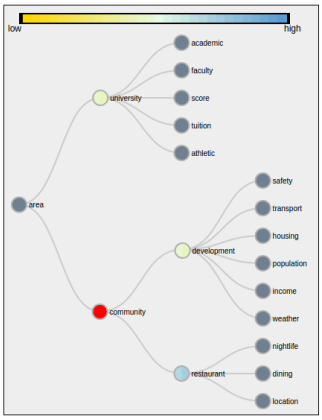
(1) Semi-automated Taxonomy



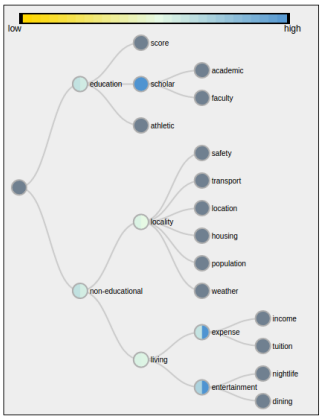
(2) Semi-automated Taxonomy



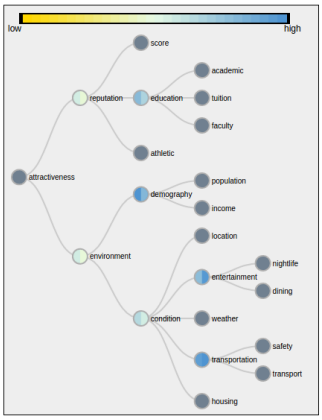
(3) Semi-automated Taxonomy



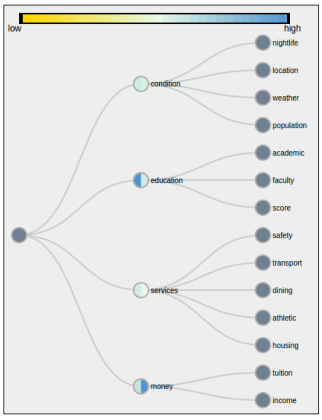
(4) Semi-automated Taxonomy



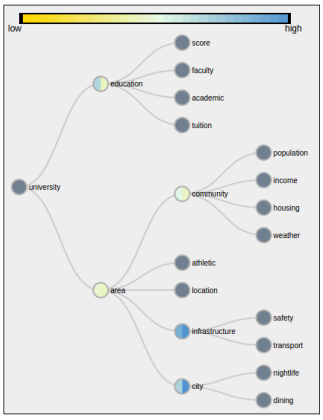
(5) Semi-automated Taxonomy



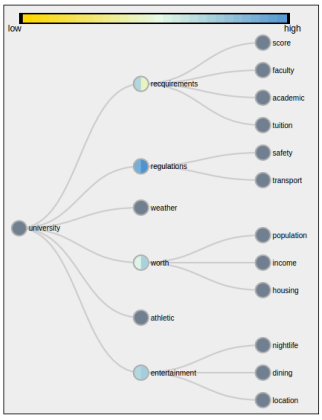
(6) Semi-automated Taxonomy



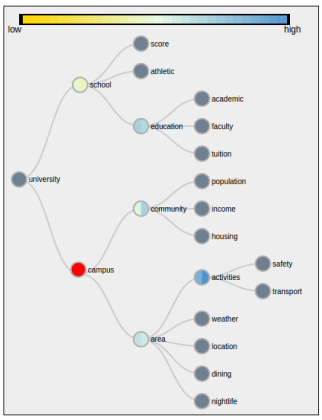
(7) Semi-automated Taxonomy



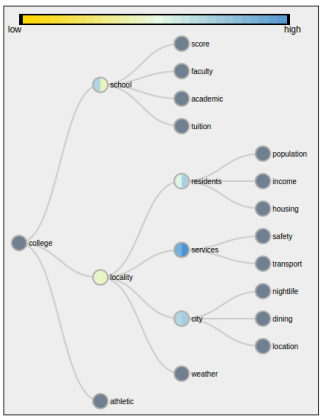
(8) Semi-automated Taxonomy



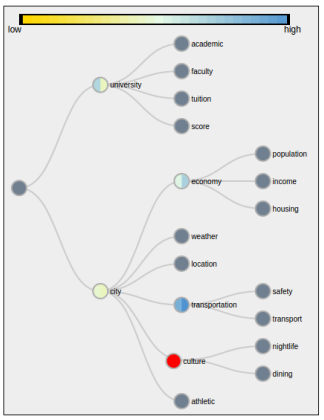
(9) Semi-automated Taxonomy



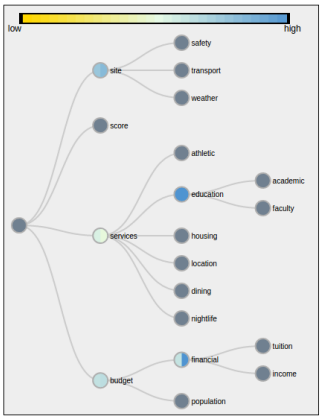
(10) Semi-automated Taxonomy



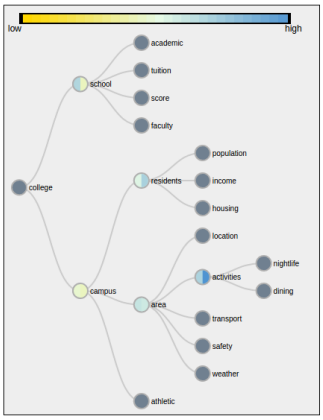
(11) Semi-automated Taxonomy



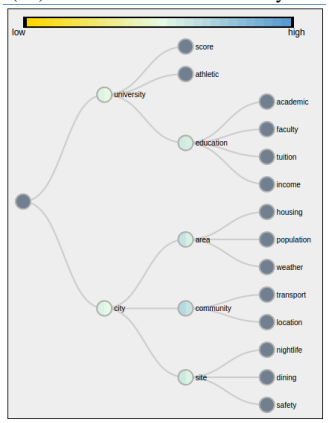
(12) Semi-automated Taxonomy



(13) Semi-automated Taxonomy

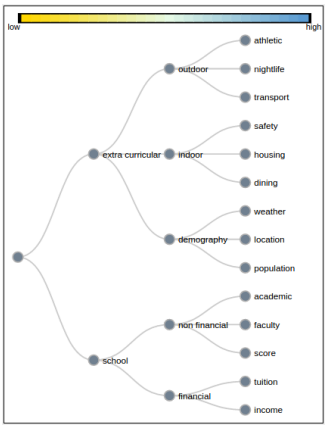


(14) Semi-automated Taxonomy

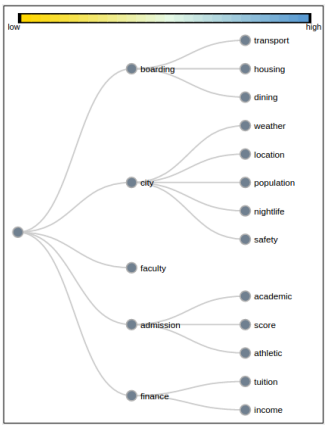


(15) Semi-automated Taxonomy

Figure 1. Shows the taxonomies generated by the participants using the semi-automated method.

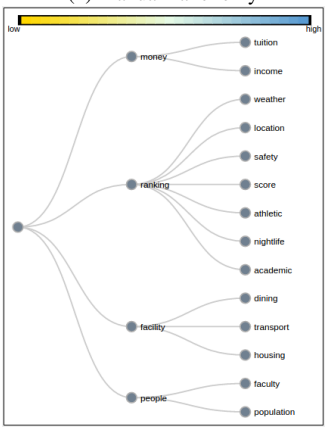


(1) Manual Taxonomy

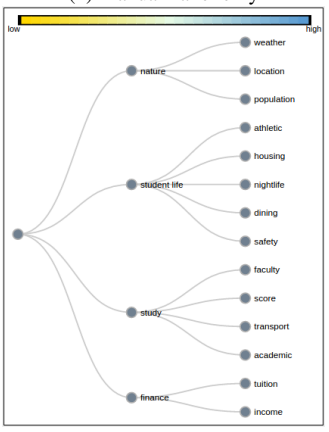


(2) Manual Taxonomy

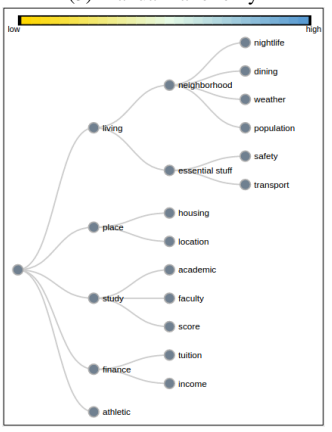
(3) Manual Taxonomy



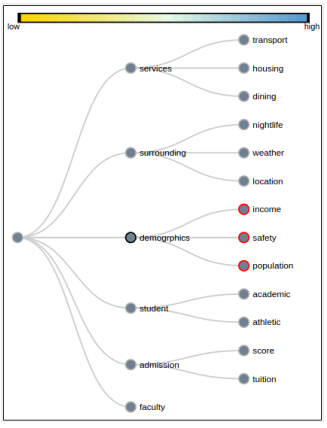
(4) Manual Taxonomy



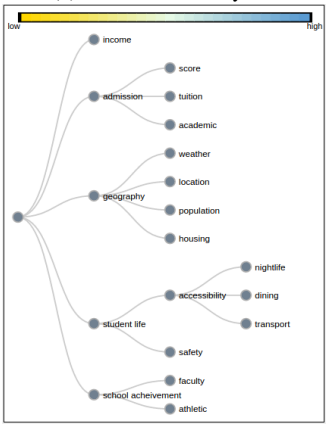
(5) Manual Taxonomy



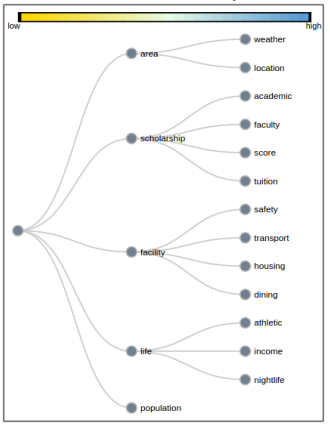
(6) Manual Taxonomy



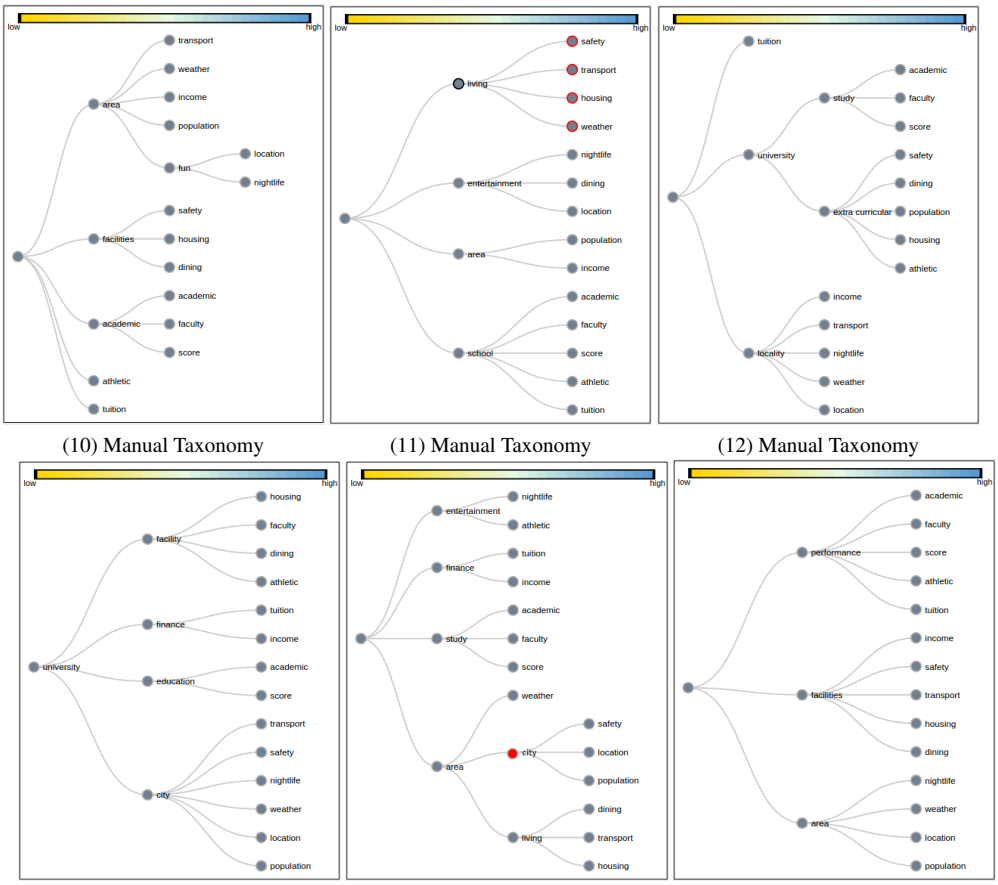
(7) Manual Taxonomy



(8) Manual Taxonomy



(9) Manual Taxonomy



(10) Manual Taxonomy (11) Manual Taxonomy (12) Manual Taxonomy

(13) Manual Taxonomy (14) Manual Taxonomy (15) Manual Taxonomy

Figure 2. Shows the taxonomies generated by the participants using the manual method.