AutoLCA: A Framework for Sustainable Redesign and Assessment of Products

M. SHAHRIAR HOSSAIN, University of Texas at El Paso
MANISH MARWAH and AMIP SHAH, HP Labs
LAYNE T. WATSON and NAREN RAMAKRISHNAN, Virginia Tech

With increasing public consciousness regarding sustainability, companies are ever more eager to introduce eco-friendly products and services. Assessing environmental footprints and designing sustainable products are challenging tasks since they require analysis of each component of a product through their life cycle. To achieve sustainable design of products, companies need to evaluate the environmental impact of their system, identify the major contributors to the footprint, and select the design alternative with the lowest environmental footprint. In this article, we formulate sustainable design as a series of clustering and classification problems, and propose a framework called AutoLCA that simplifies the effort of estimating the environmental footprint of a product bill of materials by more than an order of magnitude over current methods, which are mostly labor intensive. We apply AutoLCA to real data from a large computer manufacturer. We conduct a case study on bill of materials of four different products, perform a "hotspot" assessment analysis to identify major contributors to carbon footprint, and determine design alternatives that can reduce the carbon footprint from 1% to 36%.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications

General Terms: Algorithms, Design

Additional Key Words and Phrases: Disparate clustering, life cycle analysis, sustainable redesign

ACM Reference Format:
M. Shahriar Hossain, Manish Marwah, Amip Shah, Layne T. Watson, and Naren Ramakrishnan. 2014. AutoLCA: A framework for sustainable redesign and assessment of products. ACM Trans. Intell. Syst. Technol. 5, 2, Article 34 (April 2014), 21 pages.
DOI: http://dx.doi.org/10.1145/2505270

1. INTRODUCTION

Increasingly, environmental sustainability is becoming an important criterion of product design and competitive differentiation. For example, about 75% of enterprise customers surveyed were expected to use some type of environmental criteria in their buying decisions [Plummer et al. 2008]. A common approach to quantify broad environmental impacts is the method of life cycle assessment (LCA) [Baumann and Tillman 2004], which takes a comprehensive view of multiple environmental impacts, such as, greenhouse gas emissions, resource consumption, toxicity, carcinogenicity, etc., across the entire life cycle of a product (from cradle to grave). LCA can, for example, be used to answer questions like which is more environmentally sustainable, an e-reader or

This work is supported in part by the Institute for Critical Technology and Applied Science (ICTAS), Virginia Tech, the HP Labs Innovation Research Program (IRP), and U.S. National Science Foundation grant CCF-0937133.
Authors' addresses: M. S. Hossain (corresponding author), Department of Computer Science, University of Texas at El Paso; email: mhossain@utep.edu; M. Marwah and A. Shah, HP Labs; L. T. Watson and N. Ramakrishnan, Department of Computer Science, Virginia Tech.

Table I. Sample Nodes from a PCB BOM

Type	Part #	Description
Capacitor	71211838211Y	cap-chip-270pf-50v-k-x7r-0603-tap
	71211858231V	cap-chip-470pf-50v-j-x7r-0603-tap
	7121B1159312	capacitor-al,220uf,16v,m,-55˜+105c...
Resistor	7124A1235812	res-chip-976-1%-1/10w-0603-tap
	7124A1235812	res-chip-976-1%-1/10w-0603-tap
	7124B1216112	resistor-ar,4p2r,0,5%,1/16w,1616,tr
Inductor	7125A1123812	idut-4.7uh-20%-43mhz-650ma-smd
	7125B1147812	inductor,0.22uh,+/-10%,25mhz,250ma
	7125B1147812	inductor,0.22uh,+/-10%,25mhz,250ma

an old-fashioned book? [Goleman and Norris 2010], or how does the environmental footprint of a *Kindle*™ compare with that of a *Nook*™?

However, performing LCA is not a straightforward process. A typical system may easily consist of several thousand input variables; for example, an environmental evaluation of a server involves creating an inventory of all its components, usually down to parts, such as ICs, resistors, capacitors, fans, screws, heat sink, etc.; estimating their mass (or volume); and finally, manually mapping each component to representative entries in an environmental impact factor database. Since this process is so labor intensive, it is a challenge to estimate the environmental footprint of any realistic system, and designing to reduce environmental footprint requires deep domain expertise and experience.

Thus, two significant problems exist in achieving sustainable design on a large scale within reasonable cost and time bounds. First, an approach that allows for automation of the assessment process that eliminates the need for detailed manual data collection and inventory creation is desired. Second, a system that automatically provides a limited, narrow set of reasonable design recommendations for a given product would be very helpful.

In this article, we propose a framework (AutoLCA) composed of a palette of data mining algorithms—iterative *k*-medoids clustering, classification, disparate clustering, and constrained optimization algorithms—to address these problems. To understand how these algorithms interoperate, it is helpful to view objects/nodes (i.e., components, parts, and processes) as existing in two domains–the *substitutability* domain and the *impact factor* domain. The former is indicative of how replaceable one node is with another, captured by the node text description. For example, it might be possible to replace a particular kind of capacitor with another kind. The second domain corresponds to the impact factors of the nodes, which relates to their environmental sustainability. For instance, it has been claimed anecdotally that a Google search uses the equivalent energy of boiling a kettle of water, and thus while these two elements have widely different descriptions, they have similar sustainability characteristics.

First, we use iterative clustering methods in the *substitutability* domain to organize bill of material (BOM) data that is typically available for products in a company. An excerpt from a printed circuit board (PCB) BOM, listing a few components with attributes such as a part number and a short, unstructured text description is shown in Table I. Although a BOM provides a comprehensive listing of the components, it does not include their environmental impacts. We induce a naive Bayes classifier to map BOM nodes to nodes in an environmental impacts (EI) database. Nodes in the EI database have at least two attributes: a short, text description of the node, and a set of impact factor values. Table II shows an example of a few nodes in an EI database. The transformation of BOM nodes to the EI database achieves two purposes: (1) an

Table II. Impact Factors of Some Nodes in the EI Database

Description	SO_2 (kg)	CO_2 (kg)
capacitor, electrolyte type, > 2cm height	0.21549	47.78
resistor, SMD type, surface mounting	13.123	11.204
inductor, miniature RF chip type, MRFI	0.38215	54.542
integrated circuit, IC, memory type	2.6046	505.92

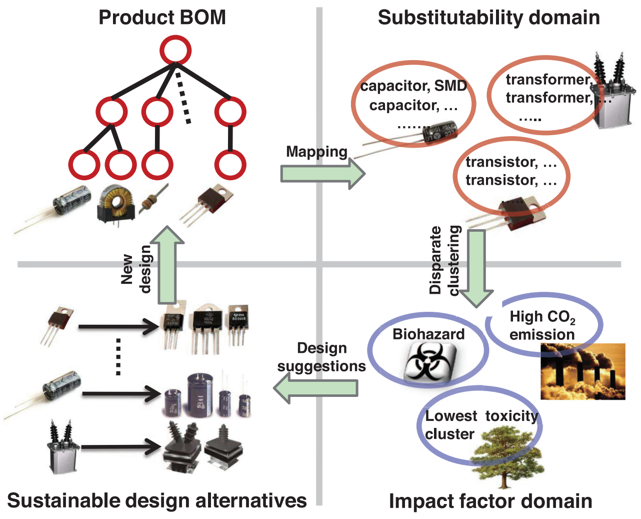


Fig. 1. The elements of AutoLCA.

automated mechanism for environmental assessment of BOM components to identify the components where redesign efforts should be focused; and (2) recommendation of sustainable design alternatives based on disparateness between the substitutability and impact factor domains, as shown in Figure 1.

To validate our approach, we applied it to real data for four different products to (1) estimate environmental footprint of the BOM components, (2) identify the top contributors to a particular impact (carbon emissions), and (3) suggest design alternatives for the top impact contributors, which on implementation could reduce the carbon footprint as much as high as 36%. The products we used are a printed circuit board (PCB) for an enterprise computer, a desktop computer (without screen), a laser jet printer, and a color laser jet printer. Although in some cases, the improvement is modest (as low as 1%), the millions of such products routinely purchased across the globe could add up to constitute a sizable contribution to sustainability. Our contributions are the following.

—We formulate sustainable design as a series of clustering–classification–clustering tasks. We address the task of finding a set of more eco-friendly alternative components by a disparate clustering formulation between two domains in the EI database.
—We demonstrate how AutoLCA simplifies estimating the environmental footprint of the BOM by more than an order of magnitude over current methods.
—We validate our techniques on real data from a large computer manufacturer.

2. RELATED WORK

A common tool used for environmental design is life-cycle assessment (LCA) [Baumann and Tillman 2004]. In this approach, detailed inventories are compiled for a given product across the material extraction, manufacturing, transportation, use, and disposal phases of a product or service. These inventories are aggregated, often by mass or energy, to obtain a comprehensive view of resources going into the product or system across the life-cycle. The aggregated inventory is then translated into environmental outputs using a set of known impact factors across predefined environmental impact categories [Glasson et al. 2005]. Once these impacts of a given product are known, the designer then tries to uncover alternative designs that might result in a lower environmental footprint. While simple in concept, sustainable product design is often challenging due to the requirements for extensive inventory data. For many complex products, the inventory could easily exceed several thousand components [Vigon 1994]. As a result, the time and cost associated with collecting accurate data across the entire life-cycle is often prohibitive. A few researchers have applied data mining techniques for sustainabile design and LCA [Patnaik et al. 2009; Marwah et al. 2011; Ramakrishnan et al. 2012; Sundaravaradan et al. 2011a, 2011b]. Sousa and Wallace [2006] develop an automated classification system to support LCA. Park and Seo [2006] propose a knowledge-based approximate life cycle assessment system to assess the environmental impacts of product design alternatives. However, neither of these consider product bill of material data.

The framework we have developed is analogous to transfer learning methods, where knowledge gained in one domain can be used to solve a similar or different problem in another domain. Transfer learning methods have received attention in recent years [Ling et al. 2008; Davis and Domingos 2009; Kuhlmann and Stone 2007] due to the fact that solutions of real-life problems require diverse knowledge from other domains. They have been studied for cross-language text classification [Ling et al. 2008], to transfer knowledge within a number of domains including molecular biology, Web, and social networks [Davis and Domingos 2009], and in gaming domains [Kuhlmann and Stone 2007]. Since bill of material nodes and environmental impact factors are in two different domains, we solve the problem of finding design alternatives in one domain (BOM) by transfering required knowledge from another (envirnomental impact). In fact, the problem is actually one of transferring negative information, because we desire to identify nodes that are substitutable for each other but have different environmental impact factors, so that we can design more sustainable alternatives.

Our research goal of finding design alternatives also has parallels to work in identifying more than one clustering solution, including alternative clustering [Qi and Davidson 2009; Dang and Bailey 2010], subspace clustering [Agrawal et al. 1998; Cheng et al. 1999], nonredundant clustering/views [Cui et al. 2007; Gondek and Hofmann 2005], associative clustering [Sinkkonen et al. 2004; Kaski et al. 2005], meta-clustering [Caruana et al. 2006; Zeng et al. 2002], and consensus clustering [Monti et al. 2003; Strehl and Ghosh 2003]. A key distinguishing feature of our work is the formulation of information-theoretic functions for clustering alternatives using a uniform contingency table framework. While contingency tables have been employed elsewhere [Sinkkonen et al. 2002; Brohee and van Helden 2006], they have been used primarily as criteria to evaluate clusterings. The few works [Nadif and Govaert 2005; Greenacre 1988] that do use contingency tables to formulate objective criteria use them in the context of a specific algorithm, such as co-clustering or block clustering, whereas we use them to find design alternatives through disparate clustering. This work can also be viewed as a form of relational clustering [Hossain et al. 2010, 2013], because we use data from two different domains, the *substitutability* domain and the *impact factor* domain, to model the 'alternativeness' property of two clusterings.

3. PROBLEM FORMULATION

Let $\mathcal{B}$ be a collection of bill of material nodes where $\mathcal{B} = \cup_i \{p_i \cup b_i\}$, $i = 1, \ldots, n_b$ are associated part numbers and short text descriptions of the nodes of the collection. The ith bill of material node is expressed as $B_i = \{p_i \cup b_i\}$. Let $\mathcal{E}$ be a database that contains the environmental impacts of a collection of n_e nodes. The attributes of $\mathcal{E}$ can be divided into two sets, t and $\mathbf{e}_j$, $j = 1, \ldots, l_e$ where t is the text description attribute and $\mathbf{e}_j$'s are the impact factor attributes. t forms a text description dataset $\mathcal{T}$ and l_e impact factor attributes form a real-valued impact factor matrix $\mathcal{X}$ of size $n_e \times l_e$. We use t_s and $\mathbf{x}_s$, respectively, to refer to the text description and impact factor vector of the s-th node E_s of $\mathcal{E}$.

Recall that we address two main problems here: (1) environmental assessment of B_i's, where 'environmental assessment' can be defined as the process of estimating different environmental impacts (such as carbon emissions, toxic effluents, energy use, etc.) for a given product; and (2) discovering more sustainable design alternatives to B_i's with the largest footprint, which we reduce to finding alternatives to corresponding E_i's.

Figure 2 shows the overall architecture of AutoLCA, which consists of two main parts. The top figure shows how we build a classifier to map BOM nodes to the environmental database and then prepare a list of design alternatives of all components. At first we cluster the BOM nodes using an iterative k-medoids clustering algorithm, then we map each medoid to the environmental database to map all node of each cluster. We train a Naive Bayes classifier on top of the mappings so that we can automatically map any new BOM to the environmental DB nodes. Finally, we use disparate clustering to find functionally similar components with disparate environmental impact factors, which provide candidates for more sustainable design recommendations.

Figure 2 (bottom) shows how we can find an alternative design of a specific product. At first, we use the classifier we trained in the figure at top to map each node of the bill of materials of the product to the environmental DB nodes. Then we use a constrained optimization fit to find environmental footprint of each component. We also find the design alternatives from the list generated earlier. We suggest replacements for the components that have high environmental footprint with similar but more environment friendly components to design a more sustainable product.

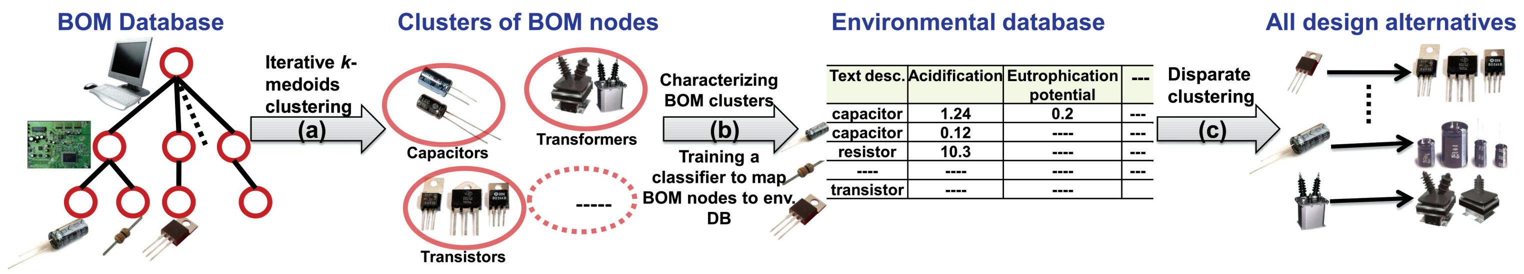
We describe AutoLCA in detail in the following three sections.

4. BOM NODES TO EI NODES MAPPING

Since BOM nodes do not have impact factor information, they need to be mapped to an environmental impact factor database. To this end, we used a real BOM dataset containing about 6.8K nodes, obtained from a large computer manufacturer, to build a classifier.

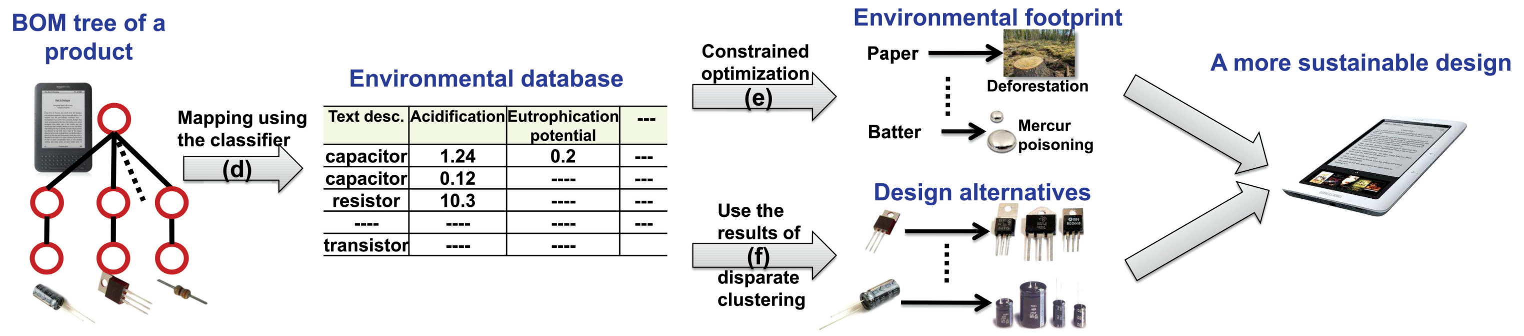
4.1. Parsing, List of Synonyms, and Units

Table I shows some sample bill of material nodes. Along with the short description, every bill of material node is associated with a part number. Similar nodes typically have a common prefix in their part numbers. They also have similar tokens in their description string. For example, a capacitor generally starts with "cap" or "capacitor", a resistor with "res" or "resistor", and an inductor with "idut" or "inductor". We maintain a list of synonyms (Table III) to combine semantically equivalent terms. Further, these descriptions often contain different units such as those for capacitance, voltage, inductance, etc. The same unit can appear in different forms. For example, capacitance can have units pf (pico-Farad), uf (μ-Farad), mfd (milli-Farad), etc. The appearance of



Text desc.	Acidification	Eutrophication potential	---
capacitor	1.24	0.2	---
capacitor	0.12	----	---
resistor	10.3	----	---
----	----	----	---
transistor	----	----	

(top) Mapping nodes of a BOM database to EI DB nodes and preparing design alternatives.



Text desc.	Acidification	Eutrophication potential	---
capacitor	1.24	0.2	---
capacitor	0.12	----	---
resistor	10.3	----	---
----	----	----	---
transistor	----	----	

(bottom) Finding a design alternative for a product.

Fig. 2. Architectural overview of AutoLCA. (Top) The steps required to build a classifier for mapping BOM nodes to environmental DB nodes and finally preparing a database of possible alternatives of components. (Bottom) how we can find a more sustainable design of a product using our framework.

Table III. List of Synonyms (Partial)

ID	Synonyms	ID	Synonyms
1	integrated circuit, ic	7	htsk, heatsink
2	capacitor, cap	8	insul, insulator
3	resistor, res	9	fstnr, fastener
4	inductor, idut,..	10	covr, cover
5	led, diode, di,..	11	crbd, cardboard
6	hdd, disk,	—	—

Table IV. List of Units (Partial)

Unit of	Units	Unit of	Units
Capacitance	pf, uf, mfd	Time	ms, ns
Resistance	ohm, kohm, mohm	Voltage	volts, v, vdc
Inductance	uh, nh	Current	a, ma
Frequency	hz, khz, mhz	Power	watts, w, kw

any of these terms denotes a capacitor. To combine similar units to a single term we maintain unit lists (Table IV).

4.2. Dissimilarity Measures

We used several measures to calculate the dissimilarity between two bill of material nodes. All these measures involve two basic dissimilarity calculations: (1) D_{LCS}—based on longest common substring (LCS), and (2) D_{LCP}—based on longest common prefix (LCP) between two strings, defined by

$$D_{LCS}(s_1, s_2) = |s_1| + |s_2| - 2 \times LCS(s_1, s_2),$$

$$D_{LCP}(s_1, s_2) = |s_1| + |s_2| - 2 \times LCP(s_1, s_2).$$

(1) *Dissimilarity based on whole strings.*

$$d_1(B_i, B_j) = D_{LCP}(p_i, p_j) + D_{LCS}(b_i, b_j).$$

(2) *Token-based Euclidean distance.* For this and the next metric, we tokenize the description based on common separators such as space, comma, semicolon, etc. We associate each distinct term in the two descriptions b_i and b_j of the bill of material nodes B_i and B_j with a dimension of Euclidean space. Note that we collapsed similar terms to a single one using our list of synonyms. The frequencies of each of the terms in the descriptions b_i and b_j form two vectors $\vec{b}_i$ and $\vec{b}_j$. In addition to this, we have the part numbers p_i and p_j. We generate two vectors $\vec{p}_i$ and $\vec{p}_j$ based on the frequencies of each of the letters of the two part numbers p_i and p_j. Then we define a dissimilarity by

$$d_2(B_i, B_j) = \|\vec{b}_i - \vec{b}_j\|_2 + \|\vec{p}_i - \vec{p}_j\|_2.$$

(3) *Token-based dissimilarity.* In this measure, we tokenize each of the bill of material node descriptions and calculate the LCS-based dissimilarity between each pair of tokens of two descriptions. We also include the LCP-based distance between the two part numbers. Suppose that the tokens of b_i are b_i^u, $u = 1, \ldots, \eta_i$ (likewise $b_j^{u'}$, $u' = 1, \ldots, \eta_j$ for b_j). We define this dissimilarity by

$$d_3(B_i, B_j) = D_{LCP}(p_i, p_j) + \frac{1}{\eta_i} \sum_{u=1}^{\eta_i} \min_{u'} \left(D_{LCS}(b_i^u, b_j^{u'}) \right),$$

where $\eta_i > \eta_j$.

We analyzed all three distance measures (d_1, d_2, and d_3) using clustering results with feedback from an expert. Based on the analysis, we selected d_3 as the dissimilarity measure for the k-medoids clustering algorithm. Section 7.1 describes the details of this selection procedure.

4.3. Clustering BOM Nodes

Mapping each of the bill of material nodes to the environmental database is a daunting task, for example, a typical BOM can contain several hundred to a few thousand nodes. Since we desire to minimize the human effort in this process, we propose mapping clusters instead of mapping the individual bill of material nodes. We use an iterative k-medoids clustering algorithm to bring similar bill of material nodes together in the same group. We use the k-medoids algorithm instead of k-means for two reasons.

(1) The bill of material database contains short text descriptions, so it is very hard to fit them in the vector space model. Although we can tokenize the text descriptions and use the frequencies of the tokens for vector quantization, it leads us to the second problem.
(2) A long mean prototype in a vector form does not provide the expert an intuition about the theme of the cluster. On the other hand, a medoid is an easy to understand representative node of a cluster.

Instead of manually mapping all the nodes of the bill of material database to the environmental database, the expert only maps the medoids of the clusters. This reduces the effort of mapping by orders of magnitude.

We construct the clusters iteratively using a k-medoids clustering algorithm. We illustrate this iterative clustering process in Alg. 1. We, at first, generate clusterings with different number of clusters and select the best clustering using overall the Average Silhouette coefficient (ASC) (steps 2 and 3). We then set a sum of squared distance (SSD) and an ASC threshold to select candidate clusters from the selected clustering for the next iteration (step 4). SSD measures the cohesion of the clusters and ASC is a measure that takes both cohesion and separation into account. Lower SSD and higher ASC values are better. We do not set any absolute value of SSD as a threshold, rather we focus on a plot of SSD where the horizontal axis refers to clusters sequenced by the ascending order of their corresponding SSD. We calculate the rate of increase of SSD (slope of the curve) at every point and find a threshold where it increases rapidly. However, we set a fixed ASC threshold of 0 to select clusters for the next iteration. Any negative ASC value of a cluster indicates overlap with other clusters, and we make the nodes of those clusters to be candidates for the next level of clustering.

The number of clusters k can vary from iteration to iteration. In our experiment, a domain expert provided the maximum number of clusters k_m that would be acceptable for manual mapping. As illustrated in Alg. 1, at every iteration, we attempt to generate around 50% of the left number of clusters. Instead of strictly generating 50% of the left clusters at any certain iteration, we generate a range of number of clusters around 50%. For example, in the first iteration, we generate clusterings with different numbers of clusters ranging from $(0.5 \times k_m) \pm (q \times k_m)$, where $q < 1.0$ reflects a percentage over k_m. We generated clusterings with different k within this range and selected the one with the highest ASC.

The purpose of the bill of material node clustering is not to present a good set of clusters of the entire dataset, but rather to present a subset of clusters that contain enough varieties of nodes in different clusters so that one medoid can represent an entire group of bill of material nodes. An expert maps the medoids of the clusters of the bill of material nodes to the environmental database. At the end of all iterations,

ALGORITHM 1: Iterative Clustering Algorithm

```
Procedure Iterative k-medoids($\mathcal{B}$, $k_m$, $q$):
    Description of parameters:
    $\mathcal{B}$ = collection of bill of material nodes,
    $k_m$ = maximum number of clusters expected,
    $q$ = a percentage ($q < 1.0$) over $k_m$.
1.  $S = \mathcal{B}$
2.  Apply k-medoids clustering on $S$ using different numbers of clusters, $k$, in the following
    range $(0.5 \times k_m) \pm (q \times k_m)$.
3.  Select the clustering that has the highest overall ASC.
4.  Store the clusters of the selected clustering possessing desired SSD and ASC.
5.  Reject the rest of the clusters and form a new $S$ combining all the nodes of the rejected
    clusters. If all the clusters are rejected, then exit.
6.  Check if nodes of the new $S$ can be assigned to any of the stored (nonrejected) clusters. A
    node of a rejected cluster can be assigned to a nonrejected cluster if the distance between
    the node of the rejected cluster and the medoid of the nonrejected cluster is less than
    the distance between the medoid and the furthest node of the same nonrejected cluster.
    If all the nodes (or a desired percentage) can be assigned to the existing stored clusters,
    then exit.
7.  $k_m = k_m -$ total number of stored (nonrejected) clusters so far.
8.  Go to step 2.
```

we had 80 clusters with 80 representative medoids. An expert manually mapped all of these 80 medoids to the nodes of environmental database. Mapping only the medoids results in a reduction of almost two orders of magnitude compared to mapping all the bill of materil nodes. Section 7.2 provides the detailed results.

4.4. BOM Node to EI Node Classification

The purpose of clustering the bill of material nodes is to reduce the human effort of manual mapping to the environmental database. As stated earlier, all the clusters generated by the iterative k-medoids clustering algorithm are given to an expert. The expert manually maps the clusters medoids to the environmental database. Note that the expert can avoid mapping a cluster if he/she feels that the medoid is not a representative one for that cluster. $k_b \leq e_n$, where k_b is the number of cluster-medoids mapped to e_n unique environmental DB node. After the mapping, we consider that each of the BOM nodes of each cluster also maps to the same environmeltal DB node as the corresponding medoid. Now that we have a number of such mappings, we build a classifier on top of it. Any new bill of material node can be mapped to an environmental DB node using this classifier. We use a naive Bayes classifier for this purpose.

Sections 7.3 and 7.4 depict experimental results on automatic mapping and how terms class probabilities can be used to characterize each cluster.

4.5. Confidence Score for Classification

Here we formulate a confidence score for the naive Bayes classification-based automatic mapping we use. Let all the T terms of all the descriptions of the training set be ordered in descending order of their conditional probabilities ($P(t|e)$) of being mapped to a particular environmental node e. The confidence score of a bill of material node b, if b is mapped to environmental DB node e and binary vector $\mathbb{B}$ records presence/absence

Table V. A "Capacitor"-Related Text Cluster in the Environmental DB

[Seed] capacitor, unspecified, at plant
capacitor, electrolyte type, $>$ 2cm height, at plant
capacitor, electrolyte type, $<$ 2cm height, at plant
capacitor, smd type, surface-mounting, at plant
capacitor, film, through-hole mounting, at plant
capacitor, tantalum-, through-hole mounting, at plant

of each of the T terms in b, is computed by the following formula:

$$S(e) = \frac{\sum_{i=1}^{T} (\mathbb{B}_i \times \exp(-i))}{\sum_{i=1}^{T} \exp(-i)}. \tag{1}$$

Note that this confidence score $S(e)$ of a bom of being mapped to environmental node e ranges from 0 to 1.0, where 0 indicates that none of the terms of the bom node match with the important terms of the description of e and 1.0 indicates a complete match. In practice, a complete match is rare, because the number of terms in a BOM node $\eta_b \ll T$. However the match can be very close to 1.0 becasue of the negative exponent used in the formula for higher order indices. In Section 7.3, we show that our classification technique maps BOM nodes to EI nodes with high confidence.

5. PREPARING DESIGN ALTERNATIVES

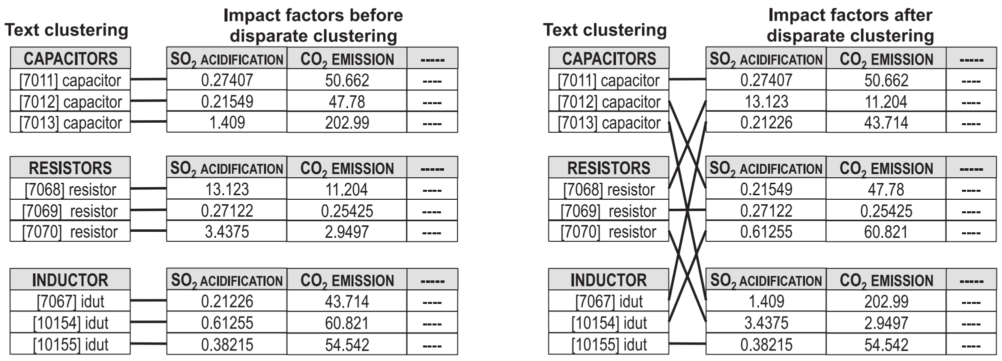
In Section 4, we described how BOM nodes can be mapped to e_n unique nodes in the environmental database. In this stage of the framework, we use these e_n nodes as seeds to form clusters around them. We select a set of neighbors for each of these e_n environmental nodes based on their text descriptions. We might obtain k_e clusters which is less than e_n because some of the seeds can be neighbors of each other. We select only those nearest neighbors that contain at least the first token of the seed node. A text cluster with a seed related to "capacitor" is shown in Table V. It shows that all the elements of the cluster are related to capacitor. We restrict the neighbor search by applying the constraint that the neighbors should contain at least the first token of the seed node because we desire the text-description based clusters to be cohesive enough to contain similar as well as substitutable elements.

Now that we have the clusters based on neighbors and text descriptions, we seek to generate an impact factor clustering that is cohesive in the impact factor space as well as dissimilar from the text-based clustering. Note that the text based k_e clusters group functionally similar components. We seek to identify a second clustering that would contain components possessing similar impact factors, at the same time, we desire this second clustering to be disparate from the text clustering. The disparateness between functionality (text) based clustering and environmental impact based clustering would provide us the design alternatives.

5.1. Formulation

Let $\mathcal{T}$ be the text dataset with k_e clusters and $\mathcal{X}$ be the corresponding environmental impact factor dataset. Each of the text descriptions of $\mathcal{T}$ is associated with one vector of $\mathcal{X}$ forming an implicit one-to-one relationship. Let us consider that there are l_x impact factors, that is, $\mathbf{x}_s \in \mathbb{R}^{l_x}$.

Since we desire to find design alternatives for each of the clusters of $\mathcal{T}$, we expect the elements of each cluster of $\mathcal{T}$ to be scattered over multiple clusters of $\mathcal{X}$. Let $C_{(t)}$ and $C_{(x)}$ be the cluster indices, that is, indicator random variables, corresponding to $\mathcal{T}$


Text clustering
Impact factors before disparate clustering
Text clustering
Impact factors after disparate clustering

CAPACITORS	SO_2 ACIDIFICATION	CO_2 EMISSION	-----
[7011] capacitor	0.27407	50.662	----
[7012] capacitor	0.21549	47.78	----
[7013] capacitor	1.409	202.99	----

RESISTORS	SO_2 ACIDIFICATION	CO_2 EMISSION	-----
[7068] resistor	13.123	11.204	----
[7069] resistor	0.27122	0.25425	----
[7070] resistor	3.4375	2.9497	----

INDUCTOR	SO_2 ACIDIFICATION	CO_2 EMISSION	-----
[7067] idut	0.21226	43.714	----
[10154] idut	0.61255	60.821	----
[10155] idut	0.38215	54.542	----

CAPACITORS
[7011] capacitor
[7012] capacitor
[7013] capacitor

RESISTORS
[7068] resistor
[7069] resistor
[7070] resistor

INDUCTOR
[7067] idut
[10154] idut
[10155] idut

SO_2 ACIDIFICATION	CO_2 EMISSION	-----
0.27407	50.662	----
13.123	11.204	----
0.21226	43.714	----

SO_2 ACIDIFICATION	CO_2 EMISSION	-----
0.21549	47.78	----
0.27122	0.25425	----
0.61255	60.821	----

SO_2 ACIDIFICATION	CO_2 EMISSION	-----
1.409	202.99	----
3.4375	2.9497	----
0.38215	54.542	----

Fig. 3. (Left) There are implicit one-to-one relationships between the text cluster nodes and the impact factor vectors. This is the initial state where our disparate clustering framework is not yet applied. (Right) The clustering of impact factor vectors is influenced by its locality as well as the text clustering. Note that we optimize the objective function over the mean prototypes of the impact factor DB only. As a result, the mean prototypes of the text clustering remains unchanged after disparate clustering.

and $\mathcal{X}$ and let k be the corresponding number of clusters. Thus, both $C_{(t)}$ and $C_{(x)}$ takes values in $\{1, \ldots, k\}$.

Let $\mathbf{m}_{i,\mathcal{X}}$ be the prototype vector for cluster i in impact factor vector-set $\mathcal{X}$. $\mathbf{m}_{i,\mathcal{X}}$'s are precisely the quantities we wish to estimate/optimize for, but in this section, assume they are given. Let $v_i^{(\mathbf{x}_s)}$ be the cluster membership indicator variables, that is, the probability that vector $\mathbf{x}_s$ is assigned to cluster i in vector-set $\mathcal{X}$. Similarly, $v_j^{(\mathbf{t}_s)}$ refers to the cluster membership indicator variables for the text dataset $\mathcal{T}$. Thus, $\sum_{i=1}^{k} v_i^{(\mathbf{x}_s)} = \sum_{j=1}^{k} v_j^{(\mathbf{t}_s)} = 1$. The traditional *hard* assignment is

$$v_i^{(\mathbf{x}_s)} = \begin{cases} 1 & \text{if} \left|\left|\mathbf{x}_s - \mathbf{m}_{i,\mathcal{X}}\right|\right| \leq \left|\left|\mathbf{x}_s - \mathbf{m}_{i',\mathcal{X}}\right|\right|, \quad i' = 1 \ldots k, \\ 0 & \text{otherwise.} \end{cases}$$

Ideally, we would like a continuous function that tracks these hard assignments to a high degree of accuracy. A standard approach is to use a Gaussian kernel to smooth out the cluster assignment probabilities:

$$v_i^{(\mathbf{x}_s)} = \frac{\exp\left(-\frac{\rho}{D}||\mathbf{x}_s - \mathbf{m}_{i,\mathcal{X}}||^2\right)}{\sum_{i'=1}^{k} \exp\left(-\frac{\rho}{D}||\mathbf{x}_s - \mathbf{m}_{i',\mathcal{X}}||^2\right)}, \tag{2}$$

where $D = \max_{s,s'} ||\mathbf{x}_s - \mathbf{x}_{s'}||^2, 1 \leq s, s' \leq n$ is the pointset diameter. ρ/D is the width of the Gaussian kernel. Notice that D is completely determined by the data but ρ is a user-settable parameter, and precisely what we can tune.

5.2. Setting up the Disparate Clustering Framework

$\mathcal{T}$ and $\mathcal{X}$ are two different but related datasets of the environmental database $\mathcal{E}$. As stated earlier, the relationships between them is one-to-one. At this point, we have the clusters of $\mathcal{T}$ and we seek to identify a clustering based on the impact factors. One straightforward approach is to apply k-means clustering on $\mathcal{X}$. In an ideal case, this approach will not provide any difference between the text clusters and the impact factor vector-set clusters. Figure 3 (left) shows this ideal situation, where every element of a text cluster is mapped to the elements of exactly one cluster of the impact factor clustering. This situation does not provide any information regarding design alternatives. In contrast, Figure 3 (right) shows a scenario where the elements of a text cluster are

Table VI. Two Contingency Tables for the Scenarios of Figure 3

3	0	0
0	3	0
0	0	3

1	1	1
1	1	1
1	1	1

distributed among multiple impact factor clusters. The benefit of this second clustering is that we can find design alternatives for the nodes of the text clustering. For example, in Figure 3 (right), we find that capacitors 7011 and 7012 can be used as alternatives of capacitor 7013. Similarly, resistor 7068 has two alternatives 7069 and 7070. The disparateness between the text clustering and the impact factor vector-set clustering provides us the information about design choices.

Now, we desire to identify a clustering in $\mathcal{X}$ which maintains its own locality but is different than a given set of assignments in the text dataset $\mathcal{T}$. We formulate this problem in a disparate clustering fashion. The assignments learnt from the nearest neighbors of the medoids of $\mathcal{T}$ will be used to influence the clustering of $\mathcal{X}$ in addition to the local memberships of $\mathcal{X}$.

We construct a $k \times k$ contingency table to capture the relationships between entries in clusters across $\mathcal{T}$ and $\mathcal{X}$. We simply iterate over the implicit one-to-one relationships between $\mathcal{T}$ and $\mathcal{X}$: we suitably increment the appropriate entry in the contingency table in a one-to-one relationship fashion: $w_{ij} = \sum_{s=1}^{n} v_i^{(\mathbf{t}_s)} v_j^{(\mathbf{x}_s)}$, where $\mathbf{t}_s$ and $\mathbf{x}_s$ refer to the sth text description and impact factor vector, respectively.

We also define $w_{i.} = \sum_{j=1}^{k} w_{ij}$, $w_{.j} = \sum_{i=1}^{k} w_{ij}$, where $w_{i.}$ and $w_{.j}$ are the row-wise and column-wise counts of the cells of the contingency table, respectively. Table VI shows two contingency tables before and after our disparate clustering framework is applied. Note that each of these contingency tables capture the relationships between two clusterings.

We will find it useful to define the distributions of the row-wise and column-wise random variables:

$$\alpha_i(j) = \frac{w_{ij}}{w_{i.}} \quad \text{and} \quad \beta_j(i) = \frac{w_{ij}}{w_{.j}}.$$

The row-wise distributions represent the conditional distributions of the clusters in $\mathcal{T}$ given the clusters in $\mathcal{X}$; the column-wise distributions are also interpreted analogously.

5.3. The Objective Function

Now that we have a contingency table, we must evaluate it to see if it reflects disparateness of the two clusterings. Ideally, we expect that the contingency table would be uniform in a perfect disparate clustering if the clusters of $\mathcal{T}$ are of equal size. Therefore in the ideal case of our objective criterion, we compare the row-wise and column-wise distributions from the contingency table entries to the uniform distribution. If the clusters of $\mathcal{T}$ are not of equal size, then the relationships of $\mathcal{X}$ should be compared with the cluster distribution of $\mathcal{T}$. Note that the distribution of the relationships of $\mathcal{T}$ is still uniform, since the clustering of $\mathcal{X}$ is unknown. We use KL-divergences to define the objective function (lower values are better):

$$\mathcal{F} = \frac{1}{k}\sum_{i=1}^{k} D_{KL}\left(\alpha_i || U\left(\frac{1}{k}\right)\right) + \frac{1}{k}\sum_{j=1}^{k} D_{KL}\left(\beta_j || U^T\left(\frac{1}{k}\right)\right)$$

$$
\begin{aligned}
&- \frac{1}{n_e} \sum_{s=1}^{N} D_{KL}\Big(p\left(V^{(\mathbf{t}_s)}\right) || U\left(\frac{1}{k}\right)\Big) \\
&- \frac{1}{n_e} \sum_{s=1}^{N} D_{KL}\Big(p\left(V^{(\mathbf{x}_s)}\right) || U^T\left(\frac{1}{k}\right)\Big), \qquad (3)
\end{aligned}
$$

where U is the uniform distribution over k clusters and U^T is the probability distribution of the elements of k text clusters, and the second two terms help guard against degenerate solutions.

We minimize this objective function over the mean prototypes of the impact factor dataset $\mathcal{X}$. For this optimization purpose, we adopt an augmented Lagrangian formulation with a quasi-Newton trust region algorithm.

6. DETERMINING ENVIRONMENTAL FOOTPRINT OF BOM COMPONENTS

In order to estimate the environmental impact of components of a BOM, they need to be mapped to nodes in the EI DB. The naive Bayes classifier addressed in the previous section, provides this transformation. However, one pending challenge with the mapping is that the units specified on the BOM and EI DB may differ. For example, most product BOMs specify the number of repeating instances for a particular part number, while the environmental DB nodes may be specified per mass (kg) or volume. Considering a given BOM as a tree with the product or part node as the root and its components as its children, we can address this problem by recognizing the following property—the parent impact is approximately equal to the sum of the impacts of the children. Knowing the quantity, in compatible units (usually weight or volume), of any EI node (parent or any child) in the product allows this can be formulated as a constrained optimization problem:

$$\min_x \sum_{i=1}^{l_e} \frac{W(i)}{\varrho I_p} \quad \text{subject to}\, x \geq 0,$$

where $W = \text{abs}(Ax - \varrho I_p)$, I_p is the impact factor vector of the parent node, A is a matrix with l_e rows (l_e is the total number of impact factors) and N_c columns (N_c is the number of child nodes), and ϱ is a coefficient or weight associated with the parent node. Note that x is a vector-valued variable of length N_c. Minimization of x allows the environmental impact of all the BOM components to be estimated. The environmental impact $\mathcal{L}_j$ of the jth factor can be computed as $\mathcal{L}_j = \sum_{i=1}^{N_c} A_{j,i} x_i$, where $A_{j,i}$ corresponds to the (j, i)th cell of matrix A and x_i corresponds to the ith element of x. We draw a pie chart for an impact factor to illustrate the corresponding environmental footprint. Section 7.6 describes some of these pie charts for a number of products.

7. EXPERIMENTAL RESULTS

In this section, we present a comprehensive evaluation of AutoLCA against a broad range of qualitative and quantitative measures. We apply AutoLCA on real data from a large computer manufacturer. The dataset we use has a total of 6,812 nodes from thirteen different bill of materials. The environmental database has 3,949 nodes with more than 200 impact factors. Then, we specifically apply AutoLCA to four different BOMs, evaluate them on the basis of carbon impact of the child nodes, present a list of substitutes resulting from disparate clustering to an expert, and finally, estimating the potential carbon savings from substitutions that may be feasible based on the recommendations of the expert.

The specific questions we seek to answer in this section are as follows.

(1) What is the performance of the distance measures introduced in Section 4.2? (Section 7.1)
(2) Does the iterative *k*-medoids clustering algorithm provide cohesive clusters? (Section 7.2)
(3) How well does automatic classification perform in mapping bill of material nodes to environmental database nodes? How is this reflected in the estimated confidence scores? (Section 7.3)
(4) How can we characterize the BOM clusters? (Section 7.4)
(5) Can disparate clustering bring functionally similar (substitutable) but environmentally diverse (based on impact factors) nodes into the same cluster? (Section 7.5)
(6) Can AutoLCA recommend viable alternatives for product components resulting in a more sustainable product design? (Section 7.6)

7.1. Analyzing Distance Measures

We analyzed all three distance measures (d_1, d_2, and d_3) introduced in Section 4.2 using clustering results with feedback from an expert. We generated 50 clusters with these three dissimilarity measures and the expert then marked BOM nodes wrongfully placed in each cluster. We observed that the whole string based dissimilarity d_1 and the token & LCS based dissimilarity d_3 have small number of clusters with large error. On the other hand, the Euclidean distance d_2 results in a number of clusters with large error. A distribution analysis on the number of erroneous BOM nodes in the clusterings using three similarity measures lead us to choose either d_1 or d_3 as our dissimilarity measure because the very few clusters with erroneous BOM nodes can become candidates for "re-clustering" and the rest of the clusters are cohesive enough to keep as they are. The total percentages of erroneous BOM nodes in these three clusterings with d_1, d_2, and d_3 are 29.5%, 32.2%, and 18.8%, respectively. We finally select d_3 because it provides clusters with the lowest amount of erroneous elements.

7.2. Results of Iterative *k*-Medoids Clustering

We perform an SSD and ASC analysis at the end of each iteration of the iterative *k*-medoids clustering. Note that some of the very high SSD and negative ASC clusters are removed at the end of each iteration which become candidates for the next iteration of the algorithm. Our observation is that at the end of every iteration, there are a few clusters with high SSD and negative ASC that contain a heterogeneous mix type of bill of material nodes. The algorithm stops iterating if all (or a desired percentage) of these nodes can be assigned to any of the previous good clusters. For our dataset with 6,812 bill of material nodes, the algorithm stopped after the third iteration leaving 864 nodes. These 864 nodes did not have enough resemblance with the elements of the existing clusters and they did not have enough structure to form a new clusters either. As stated earlier in Section 4.3, at the end of all iterations we had 80 clusters with 80 representative medoids which were manually mapped to the environmental DB nodes by an expert.

7.3. Mapping BOM Nodes to EI Nodes

Eighty medoids generated by our iterative *k*-medoids clustering were mapped to 30 unique environmental DB nodes by an expert. For the classification purpose, we consider that each of the BOM nodes of each cluster also maps to the same environmental DB node as the corresponding medoid. We build a naive Bayes classifier on top of this

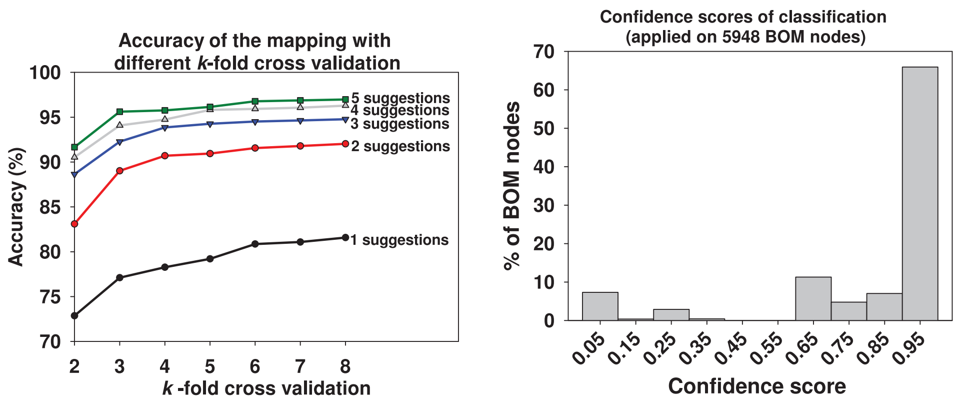


Fig. 4. (Left) Classifier's evaluation: we calculated the accuracy with *k*-fold cross validation (*k* being varied from 2 to 8). The accuracies are shown for different number of suggestions. (Right) Classifier's confidence score: around 90% of the bill of material nodes have confidence score greater than 0.6.

mapped data. We have a total of 1,932 terms in 5,948 bill of material descriptions of the 80 clusters. All these terms become features for the naive Bayes classifier.

We found that the accuracy of the classifier improves (around 4%) with the use of synonyms along with the units. More importantly, it is essential to keep the synonyms in the same bucket for characterization (described in the following subsection). We evaluate the mapping process using cross validation technique. We calculated the accuracies using different number of folds ranging from two to eight with varying number of suggested mappings ranging from one to five. The plot of Figure 4 (left) shows that the classifier has more than 95% accuracy when we use 3- to 8-folds with five suggested mappings from a pool of 30 options. With the lowest amount of training (2-fold), the accuracy is still more than 90% with five suggestions. The accuracy is more than 80% with any number of folds for more than one suggestions. The accuracies of the mappings are fairly high for a pool of thirty class labels.

We applied naive Bayes classification on 5,948 nodes of all the clustered data and computed confidence score (Section 4.5) for each of the nodes. Figure 4 (right) shows the distribution of the confidence scores computed over all 5,948 nodes. We observed that around 90% of the bill of material nodes have confidence score greater than 0.6 which indicates high confidence in automatically mapping the bill of material nodes using naive Bayes classification.

7.4. Characterization of the BOM Clusters

The most frequently occurring terms in a BOM cluster allow us to characterize it by associating (tagging) that cluster with those terms. In fact, this can assist an expert in the manual mapping process. After we have the medoids (clusters) mapped and the classifier trained, we can characterize the different classes using the terms with high class probabilities. These terms could also be shown to the user to justify the categorization of a new BOM node to a particular EI DB node. The top left plot of Figure 5 shows the list of terms in the horizontal axis ordered based on their probabilities of being mapped to EI DB node 7013 (capacitor). All the BOM clusters mapped to EI DB node 7013 can be characterized by the terms, "capacitor", "cap", "capacitance", "voltage", etc. In the same figure, we show three more similar plots for inductor, integrated circuit, and resistor related clusters. We have a total of thirty such plots (only four of them are shown in Figure 5) to characterize 80 BOM clusters.

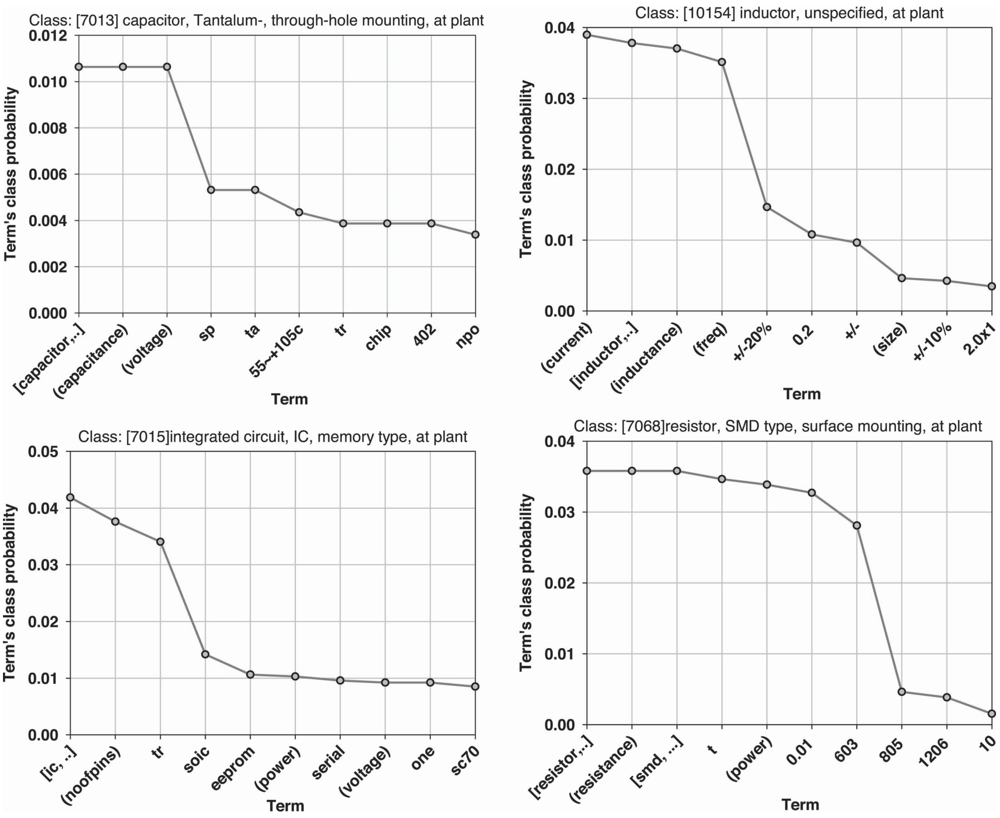


Fig. 5. We characterize the bill of material clusters by the top terms with highest class probabilities of being mapped to a corresponding environmental DB node. All the plots show the top term probabilities which can be used to characterize any BOM cluster. For example, the upper-left plot shows that the characterizing terms of any BOM cluster which maps to the environmental DB node 7013 are "capacitor", "cap", "capacitance", "voltage", etc. The rest of the plots show the characterizing terms of inductor (10154), integrated circuit (7015), and resistor (7068). Note that terms inside square brackets indicate a set of synonyms, and a term inside parentheses indicates a unit.

7.5. Construction of Text Clusters and Disparate Impact Factor Clusters

For the disparate clustering phase of the experiments, the experts selected 46 impact factors of their interest out of more than 200 for ease of analysis and verification. As described in Section 5, we construct text description clusters based on neighbors in the EI database. We seek to identify a second clustering that would contain components possessing similar impact factors, at the same time, we desire this second clustering to be disparate from the text clustering. We obtained a total of 170 nodes around 30 seeds. (These 30 seeds are the unique environmental DB nodes used as class labels in Section 7.3). The disparateness between functionality (text) based clustering and environmental impact based clustering would provide us the design alternatives. Table VII shows some text-based clusters ($\mathcal{C}_1$) and the disparate cluster IDs ($\mathcal{C}_2$) of the correspondign nodes. In Table VII, two integrated circuits (7016 and 7015) are shown to be in the same text-based cluster but they are in two different clusters in the impact factor-based disparate clustering. Similarly, capacitor 7014 has two alternatives (7010 and 7013), all of which are in different impact factor cluster than capacitor 7014. Figure 6 shows distributions of percentage difference of the impact factors between capacitor

Table VII. Text Description-Based and Impact Factor-Based Disparate Cluster Samples

EI ID	Text description	$\mathcal{C}_1$	$\mathcal{C}_2$
7016	integrated circuit, logic type	23	25
7015	integrated circuit, memory type	23	6
7014	capacitor, unspecified, at plant	27	23
7013	capacitor, tantalum type	27	12
7012	capacitor, electrolyte type I	27	23
7011	capacitor, electrolyte type II	27	23
7010	capacitor, smd type, surface-mounting	27	17
8054	tantalum, powder, capacitor-grade	27	10
7067	inductor, ring core choke type	14	23
10154	inductor, unspecified	14	6
7111	diode, unspecified	25	27
7077	light emitting diode	25	22
7075	diode, glass-, smd type, surface..	25	27
7076	diode, glass-, through-hole mounting	25	22
1056	aluminium, production mix	39	27
1057	aluminium, production mix, cast alloy	39	3
1058	aluminium, production mix, wrought alloy	39	22
1670	alkyd paint, white, 60% in solvent	7	20
1669	alkyd paint, white, 60% in h2o	7	10

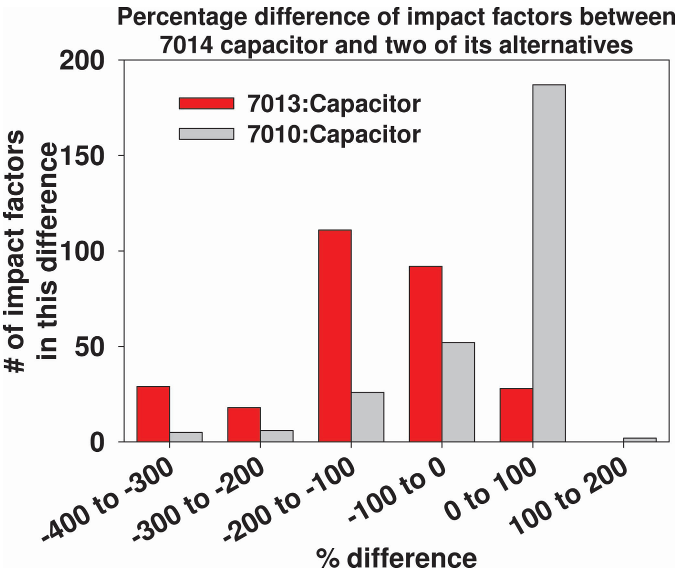
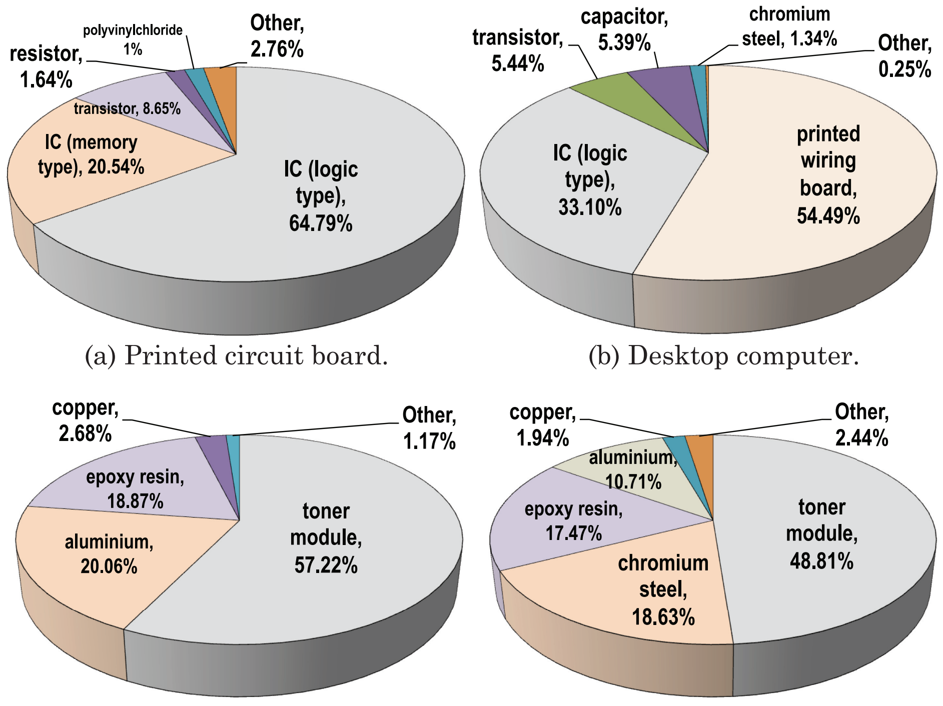


Fig. 6. Percentage difference between the impact factors of 7014:Capacitor and two of its alternatives 7010 and 7013. Negative difference in impact factor indicates that the alternative has larger corresponding impact than 7014:Capacitor.

7014 and two of its alternatives. A positive difference indicates that the alternative has lower impact whereas a negative difference indicates that a replacement with the alternative would result in a higher impact at that specific factor. Note that 7010 has higher positive difference than 7013. On the other hand, 7013 has higher negative difference than 7010 indicating that the impact factors of these two alternatives are very diverse.

Table VIII. Error between the Sum of the Child Node Impacts and the Parent (percentage median and mean errors)

Product	#components	#children	Median error(%)	Mean error(%)
Printed circuit board	560	14	0.2069	2.31687
Desktop computer	159	16	4.85987	12.5731
Laser jet printer	435	26	3.78832	14.9138
Color laser jet printer	456	24	3.83017	14.9089



(a) Printed circuit board. (b) Desktop computer.

(c) Laser jet printer. (d) Color laser jet printer.

Fig. 7. Hotspot analysis of the carbon footprint of four different products.

7.6. Carbon Hotspot Analysis and Redesign

We apply AutoLCA to four specific products: a printed circuit board (PCB) for an enterprise computer, a desktop computer (without screen), a laser jet printer, and a color laser jet printer. The BOM of these products is mapped to EI nodes using the classifier. In order to determine the quantity of each component in the products (which in turn is used to assess the environmental impact of the components), we solve the constrained optimization problem described in Section 6. The results are summarized in Table VIII and show good fits.

After determing the component impact factors, these can be used to perform a carbon hotspot analysis, where essentially a Pareto list of the biggest environmental contributors to the parent footprint is generated so that a designer or LCA practitioner can zoom in on where further efforts should be focused. Figure 7 shows such a hotspot analysis of the four products for the impact of carbon emissions. Both for PCB and desktop, the largest contributors are integrated circuits (ICs) (85% for PCB and 33% for desktop; note that for desktop, the printed wiring board itself would contain ICs, so essentially this percentage would be actually higher). This is followed by discrete components

Table IX. Carbon Footprint Reduction for Four Products

Product	Reduction
Printed circuit board	2%
Desktop computer	36%
Laser jet printer	1%
Color laser jet printer	1%

such as transistors, resisters, capacitors. This is quite similar to the results uncovered through a manual LCA, where the ICs were found to have the largest contributions. For the printers, the largest contributor is the toner, which again matches manual studies. However, aluminum is higher than expected, particularly for the laser jet printer. The contribution of epoxy (plastics) also matches well. A designer desiring to further reduce the carbon footprint of any of these products can correctly identify where efforts should be focused. This is important, because it becomes possible for someone who has no LCA or environmental background to automatically obtain feedback regarding the sustainability of their design.

Next, we aim to provide suggestions for more sustainable design of the four products. For each product, we look for child nodes that belong to the same cluster in $\mathcal{C}_1$ (text-based clusters) but in different ones in $\mathcal{C}_2$ (disparate clusters). Such nodes are likely similar in functionality but different in their environmental impacts, making them good candidates for substitution in order to reduce environmental footprint. A list of such substitutions were presented to an expert to advise on their feasibility, based on which the savings in the carbon footprint of each product was estimated as shown in Table IX. While redesign of desktop results in a potential savings of 36% in the carbon footprint, for the PCB and the printers, the savings are in the range of 1–2%. The reason for low savings in the printers is that the main contributors to carbon footprint, namely, toner, epoxy and chromium steel do not have good substitutes in the database.

The fact that the savings are not large for the printers is very useful from a practitioner's standpoint—it essentially means that Auto-LCA can be used as a tool to expediantly decide where further Design for Environment (DfE) efforts should be focused. For example, it could help a large company to decide which class of its products it should focus on first (e.g., desktops vesus printers). Thus Auto-LCA is useful in identifying environmental impact hotspots both within a product and across products.

8. CONCLUSION

We have proposed a framework for environmental assessment and redesign of products from their bill of materials. Sustainable redesign is formulated as a transfer learning problem to map between diverse attribute spaces. A detailed case study on several complex electronic products shows that our framework can expeditiously determine the environmental hotspots in products providing valuable information on which components to focus on for redesign. For the four products considered, our redesign recommendations based on disparate clustering can reduce carbon footprint from 1% to 36%, enabling companies to decide which class of products is most suitable for environmental redesign. In future we will expand our work to other domains to be able to further assess the compatibility of the discovered design alternatives.

REFERENCES

R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. 1998. Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.* 27, 2 (1998), 94–105.

H. Baumann and A. M. Tillman. 2004. *The Hitchhiker's Guide to LCA: An Orientation in Life Cycle Assessment Methodology and Application*. Studentlitteratur, Sweden.

S. Brohee and J. van Helden. 2006. Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics* 7 (2006), 488.

R. Caruana, M. Elhawary, N. Nguyen, and C. Smith. 2006. Meta clustering. In *Proceedings of ICDM'06*. 107–118.

C. Cheng, A. W. Fu, and Y. Zhang. 1999. Entropy-based subspace clustering for mining numerical data. In *Proceedings of KDD'99*. 84–93.

Y. Cui, X. Fern, and J. G. Dy. 2007. Non-redundant multi-view clustering via orthogonalization. In *Proceedings of ICDM'07*. 133–142.

X. Dang and J. Bailey. 2010. Generation of alternative clusterings using the CAMI approach. In *Proceedings of SDM'10*. 118–129.

J. Davis and P. Domingos. 2009. Deep transfer via second-order Markov logic. In *Proceedings of the International Conference on Machine Learning (ICML'09)*. 217–224.

J. Glasson, R. Therivel, and A. Chadwick. 2005. *Introduction to Environmental Impact Assessment: Principles and Procedures, Process, Practice and Prospects* (3rd ed.). Taylor & Francis, New York, NY.

Daniel Goleman and Gregory Norris. 2010. How green is my iPad? http://www.nytimes.com/interactive/2010/04/04/opinion/04opchart.html?hp. (Last accessed 4/2010).

D. Gondek and T. Hofmann. 2005. Non-redundant clustering with conditional ensembles. In *Proceedings of KDD'05*. 70–77.

M. Greenacre. 1988. Clustering the rows and columns of a contingency table. *J. Classification* 5, 1 (1988), 39–51.

M. S. Hossain, N. Ramakrishnan, I. Davidson, and L. T. Watson. 2013. How to alternatize a clustering algorithm. *Data Min. Knowl. Discov.* 27, 2 (2013), 193–224.

M. S. Hossain, S. Tadepalli, L. T. Watson, I. Davidson, R. F. Helm, and N. Ramakrishnan. 2010. Unifying dependent clustering and disparate clustering for non-homogeneous data. In *Proceedings of KDD'10*. 593–602.

S. Kaski, J. Nikkilä, J. Sinkkonen, L. Lahti, J. E. A. Knuuttila, and C. Roos. 2005. Associative clustering for exploring dependencies between functional genomics data sets. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 2, 3 (2005), 203–216.

G. Kuhlmann and P. Stone. 2007. Graph-based domain mapping for transfer learning in general games. In *Proceedings of ECML'07*. 188–200.

X. Ling, G. R. Xue, W. Dai, Y. Jiang, Q. Yang, and Y. Yu. 2008. Can Chinese web pages be classified with english data source? In *Proceedings of WWW'08*. 969–978.

M. Marwah, A. Shah, C. Bash, C. Patel, and N. Ramakrishnan. 2011. Using data mining to help design sustainable products. *IEEE Computer* 44, 8 (2011), 103–106.

S. Monti, P. Tamayo, J. Mesirov, and T. Golub. 2003. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learn.* 52, 1–2 (2003), 91–118.

M. Nadif and G. Govaert. 2005. Block clustering of contingency table and mixture model. In *Proceedings of IDA'05*. 249–259.

J.-H. Park and K.-K. Seo. 2006. A knowledge-based approximate life cycle assessment system for evaluating environmental impacts of product design alternatives in a collaborative design environment. *Adv. Eng. Inform.* 20, 2 (2006), 147–154.

D. Patnaik, M. Marwah, R. Sharma, and N. Ramakrishnan. 2009. Sustainable operation and management of data center chillers using temporal data mining. In *Proceedings of KDD'09*. 1305–1314.

D. Plummer, C. Smulders, L. Flering, Y. Natis, S. Mingay, M. Driver, J. Fenn, L. McLellan, and D. Wilson. 2008. Gartner's top predictions for IT organizations and users, 2008 and beyond. Report ID G00154035, Gartner Inc. (2008).

Z. Qi and I. Davidson. 2009. A principled and flexible framework for finding alternative clusterings. In *Proceedings of KDD'09*. 717–726.

N. Ramakrishnan, M. Marwah, A. Shah, D. Patnaik, M. S. Hossain, N. Sundaravaradan, and C. Patel. 2012. Data mining solutions for sustainability problems. *IEEE Potentials* 31, 6 (2012), 28–34.

J. Sinkkonen, S. Kaski, and J. Nikkilä. 2002. Discriminative clustering: Optimal contingency tables by learning metrics. In *Proceedings of ECML'02*. 418–430.

J. Sinkkonen, J. Nikkilä, L. Lahti, and S. Kaski. 2004. Associative clustering. In *Proceedings of ECML'04*. 396–406.

I. Sousa and D. Wallace. 2006. Product classification to support approximate life-cycle assessment of design concepts. *Technol. Forecast. Social Change* 73, 3 (2006), 228–249.

A. Strehl and J. Ghosh. 2003. Cluster ensembles—A knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* 3 (2003), 583–617.

N. Sundaravaradan, M. Marwah, A. Shah, and N. Ramakrishnan. 2011a. Data mining approaches to life cycle assessment. In *Proceedings of IEEE (ISSST'11)*.

N. Sundaravaradan, D. Patnaik, N. Ramakrishnan, M. Marwah, and A. Shah. 2011b. Discovering life cycle assessment trees from impact factor databases. In *Proceedings of AAAI'11*.

B. W. Vigon. 1994. *Life-Cycle Assessment: Inventory Guidelines and Principles*. CRC Press, New York, NY.

Y. Zeng, J. Tang, J. Garcia-Frias, and G. R. Gao. 2002. An adaptive meta-clustering approach: Combining the information from different clustering results. In *Proceedings of CSB'02*. 276–287.

Received October 2012; revised May 2013; accepted June 2013