Accelerating Innovation Through Analogy Mining

Tom Hope
The Hebrew University of Jerusalem
tom.hope@mail.huji.ac.il

Joel Chan
Carnegie Mellon University
joelchue@cs.cmu.edu

Aniket Kittur
Carnegie Mellon University
nkittur@cs.cmu.edu

Dafna Shahaf
The Hebrew University of Jerusalem
dshahaf@cs.huji.ac.il

ABSTRACT
The availability of large idea repositories (e.g., the U.S. patent database) could significantly accelerate innovation and discovery by providing people with inspiration from solutions to analogous problems. However, finding useful analogies in these large, messy, real-world repositories remains a persistent challenge for either human or automated methods. Previous approaches include costly hand-created databases that have high relational structure (e.g., predicate calculus representations) but are very sparse. Simpler machine-learning/information-retrieval similarity metrics can scale to large, natural-language datasets, but struggle to account for structural similarity, which is central to analogy. In this paper we explore the viability and value of learning simpler structural representations, specifically, “problem schemas”, which specify the purpose of a product and the mechanisms by which it achieves that purpose. Our approach combines crowdsourcing and recurrent neural networks to extract purpose and mechanism vector representations from product descriptions. We demonstrate that these learned vectors allow us to find analogies with higher precision and recall than traditional information-retrieval methods. In an ideation experiment, analogies retrieved by our models significantly increased people’s likelihood of generating creative ideas compared to analogies retrieved by traditional methods. Our results suggest a promising approach to enabling computational analogy at scale is to learn and leverage weaker structural representations.

KEYWORDS
Computational analogy; innovation; creativity; product dimensions; text mining; text embedding

1 INTRODUCTION
The ability to find useful analogies is critical to driving innovation in a variety of domains. Many important discoveries in science were driven by analogies: for example, an analogy between bacteria and slot machines helped Salvador Luria advance the theory of bacterial mutation. Analogical reasoning forms the foundation of law, with the effectiveness of an argument often dependent on its

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

1innocentive.com
2quirky.com
3OpenIDEO.com
mining and information retrieval include a variety of natural language techniques that use words, parts of speech or other language feature-based vector representations in order to calculate similarity measures (see [31]). Examples include word embedding models like Word2Vec [24], vector-space models like Latent Semantic Indexing [10], and probabilistic topic modeling approaches like Latent Dirichlet Allocation [4]. These approaches excel at detecting surface similarity, but are often unable to detect similarity between documents whose word distributions are disparate. The problem is especially acute when the source and target domains are different (for example, bacterial mutation and slot machines).

Another approach to finding analogies has been to use the structural similarity of sentences or texts, such as using coupled clustering for detecting structural correspondence of text [5, 35]. However, these approaches typically require rich data sets with clear substructures, whereas most descriptions of problems or ideas in existing online databases are short, sparse, or lack consistent structure. Other current methods focus on very narrow analogy tasks, such as four-term analogy problems (teacher:student = doctor:?), in particular with short strings (ABC:ABD = KJI:? ) [19]. In contrast, we wish to find analogies in real world data, which involve complex representations and a diverse set of analogical relations.

In this paper, we are interested in automatically discovering analogies in large, unstructured data sets. In particular, we focus on a corpus of product innovations. There are two insights behind our approach that we believe may make this problem tractable despite its longstanding status as a “holy grail” in both cognitive science and AI. First, rather than trying to solve the problem of fully structured analogical reasoning, we instead explore the idea that for retrieving practically useful analogies, we can use weaker structural representations that can be learned and reasoned with at scale (in other words, there is a tradeoff between the ease of extraction of a structure and its expressivity). Specifically, we investigate the weaker structural representation of an idea’s purpose and mechanism as a way to find useful analogies. The second insight is that advances in crowdsourcing have made it possible to harvest rich signals of analogical structure that can help machine learning models learn in ways that would not be possible with existing datasets alone.

This paper combines these two ideas to contribute a technique for computationally finding analogies from unstructured text datasets that go beyond surface features. At a high level, our approach uses the behavioral traces of crowd workers searching for analogies and identifying the purpose and mechanisms of ideas, then developing machine learning models that develop similarity metrics suited for analogy mining. We demonstrate that learning purpose and mechanism representations allows us to find analogies with higher precision and recall than traditional information-retrieval methods based on TF-IDF, LSA, LDA and GloVe, in challenging noisy settings. Furthermore, we use our similarity metrics to automatically find far analogies – products with high purpose similarity, and low mechanism similarity. In a user study, we show that we are able to “inspire” participants to generate more innovative ideas than alternative baselines, increasing the relative proportion of positively-rated ideas by at least 25%.

2 LEARNING A REPRESENTATION FOR ANALOGIES

2.1 Motivation

Much work in computation analogy has focused on fully structured data, often with logic-based representations. For example [11],

\[ \text{CAUSE(} \text{GREATER-THAN}\text{TEMPERATURE(coffee),} \]
\[ \text{TEMPERATURE (ice-cube)}, \]
\[ \text{FLOW(coffee, ice-cube, heat, bar)} \]

These representations, while very expressive, are notoriously difficult to obtain. In this section, we investigate a weaker structural representation. Our goal is to come up with a representation that can be learned, while still being expressive enough to allow analogical mining.

Analogies between product ideas are intricately related to their purpose and mechanism. Informally, we think of a product’s purpose as “what it does, what it is used for”, and a product’s mechanism is “how it does it, how it works”. The importance of a product’s purpose and mechanism as core components of analogy are theoretically rooted in early cognitive psychology work on schema induction which define the core components of a schema as a goal and proposed solution to it (e.g., [15]). More recently, the practical value of defining a problem schema as a purpose and mechanism has been demonstrated to have empirical benefits in finding and using analogies to augment idea generation (e.g., [38–41]).

Separating an idea into purpose and mechanisms enables core analogical innovation processes such as re-purposing: For a given product (such as a kitchen-sink cleaner) and its purpose, finding another way to put it to use (cleaning windows). To that end, assume (for the moment) that we have for each product i two vectors, \( p_i \) and \( m_i \), representing the product’s purpose and mechanism, respectively. Using this representation, we are able to apply rich queries to our corpus of products, such as:

- **Same purpose, different mechanism.** Given the corpus of all products \( P \), a product \( p \) with (normalized) purpose and mechanism vectors \( p_i \) and \( m_i \), and distance metrics \( d_P(\cdot, \cdot), d_m(\cdot, \cdot) \) between purpose and mechanism vectors (respectively), solve:

\[
\arg\min_{i \in P} d_P(p_i, p) \\
\text{s.t.} d_m(m_i, m) \geq \text{threshold},
\]

- **Same Mechanism, different purpose.** Solve:

\[
\arg\min_{i \in P} d_m(m_i, m) \\
\text{s.t.} d_P(p_i, p) \geq \text{threshold},
\]

The decomposition of products into purpose and mechanism also draws inspiration from engineering functional models and ontologies for describing products [18]. Although there is no set common definition of functions [25], much research on functionality has been conducted in areas such as functional representation, engineering design and value engineering. The scope of these ontologies, however, is highly “mechanistic” or engineering-oriented, while in many cases we observe in product data the purpose of a product is more naturally understood in others term – such as whether it is for entertainment, leisure, or more serious purposes, who is the target user (adults, children), and so forth.
Importantly, our dataset of product descriptions contains noisy texts, often written informally by non-professional people. In these texts product descriptions are often lacking detail or are ill-defined. To automatically describe a product in terms of a formal functional model would require an inordinate amount of meticulous data annotation and collection by professional engineers over a large number of product descriptions. We thus resort to a softer approach, hoping that a compromise on the level of detail will enable data-driven methods to automatically extract useful representations of product purpose and mechanism.

Finally, we also make note of the potentially wider applicability of automatically extracting these representations from real-word product descriptions. Identifying the key components and functions of products could conceivably improve (or augment) search capabilities in internal or external product databases, and perhaps enhance recommender systems by better understanding what a user is looking for in a product and what a product offers. This last idea is connected to a line of work on “product dimensions” [23], in which it is shown that implicitly identifying the properties of products (such as that Harry Potter is a book about wizards), helps in improving recommendations. The authors propose a method that combines ratings data with textual product reviews, hoping to implicitly recover topics in the text that inform recommendations. We too look at product dimensions, but target only two that are more abstract and broad, and directly learn them in a supervised fashion from annotated data.

2.2 Data

Innovation Corpus. We test our approach with a corpus of product descriptions from Quirky.com, an online crowdsourced product innovation website. Quirky is representative of the kinds of datasets we are interested in, because it is large (at the time of writing, it hosts upwards of 10,000 product ideas, of which our corpus included 8500), unstructured (ideas are described in natural language), and covers a variety of domains (invention categories) which makes cross-domain analogies possible. The following example illustrates the typical length and “messiness” of product ideas in this dataset:

Thirsty too
• Pet water bowl/dispenser for your vehicle cup holder.
• Over spill lip to catch water
• Has optional sleeve for larger cup holders
• Optional floor base
• One way valve so water can’t over flow from bottle
• Small reservoir
• Reservoir acts as backsplash
• Water bottle attachment
• Holds water in your vehicle cupholder for pet
• Foldable handle to get unit out of holder
• Dishwasher safe
• Optional sleeve for larger cup holders

Collecting Purpose and Mechanism Data. In addition to the Quirky innovation corpus, we needed to collect analogy-specific data to train our model. Previous approaches to creating structured representations of items for analogical computation, for example predicate logic, are extremely heavyweight and can take tens of person-hours for complex items [37]. Instead, we aim to develop a lightweight task that avoids complex structure but instead relies on the cognitive expertise and intuitions of people to be able to separate the purpose of a product from its mechanism. By doing so we can scale up the collection of purpose and mechanism labels through the use of microtasks and crowdsourcing markets [21]. Specifically, we show Amazon Mechanical Turk (AMT) crowd workers a product description, asking them to annotate the parts of the text they consider to be about the purposes of the product, and the parts related to mechanisms. We frame the problem in simple terms, guiding workers to look for words/phrases/chunks of text talking about “what the product does, what it is good for” (purposes), and “how it works, what are its components” (mechanisms). As seen in Figure 1, we juxtapose two copies of the product text side-by-side, to ease cognitive load and encourage workers not to give purpose and mechanism tags that are too similar or overlapping, thus capturing a potentially richer and more distinct signal. Our corpus consisted of 8500 products. Each product was annotated by four workers.

Collecting Analogies. In previous, preliminary work [7], we explored the use of crowdsourcing to label analogies, collecting labeled examples of analogies (product pairs) that were fed into a metric-learning deep learning model. While showing promising results, the process of collecting labeled analogies proved expensive, requiring considerable cognitive effort from workers and thus more time, limiting the number of labels that can realistically be collected. In addition, in that work, the deep learning model was blind to the rich structures of purpose and mechanism, and had no hope of recovering them automatically due to relative data scarcity. In this paper we take a different approach, focusing our resources on collecting purpose and mechanism annotations from the crowd, while collecting only a small number of curated labeled analogies strictly for the purpose of evaluation (see Section 3.1 for details).

2.3 Method

2.3.1 Extracting Purpose and Mechanism vectors. In this section, we describe our approach to learning to extract purpose and mechanism product representations. We begin with a set of $N$ training product texts $X_N = \{x_1, x_2, \ldots, x_N\}$, where each $x_i$ is a variable-length sequence of tokens $(x_{i1}^1, x_{i2}^1, \ldots, x_{iT_i})$. For each document $x_i$, we collect a set of $K$ purpose annotations and $K$ mechanism annotations, where $K$ is the number of workers who annotate each document. We define the purpose annotation to be a binary vector

$\hat{p}_{ik} = (\hat{p}_{i1}^k, \hat{p}_{i2}^k, \ldots, \hat{p}_{iT_i}^k)$

of the same length as $x_i$, with $\hat{p}_{ij}^k = 1$ if token $x_{ij}$ is annotated as purpose by annotator $k$, and $\hat{p}_{ij}^k = 0$ if not. In the same way, we denote
the mechanism annotation with

\[ \hat{m}_{ik} = (\hat{m}_{ik}^1, \hat{m}_{ik}^2, \ldots, \hat{m}_{ik}^T) \]

While on the surface this setting appears to lend itself naturally to sequence-to-sequence learning [33], there are a few important differences. A key difference is that in our setting the problem of interest is not to learn to recover the latent \( \hat{p} (\hat{m}) \) exactly for unseen products, but rather to extract some form of representation that captures the overall purpose and mechanism. That is, what we do care about is the semantic meaning or context our representation captures with respect to product purposes or mechanisms, rather than predicting any individual words. Additionally, sequence-to-sequence models typically involve heavier machinery and work well on large data sets, not suitable for our scenario were we have giving higher impact to words considered important by all annotators and weighting by TF-IDF, we naturally give more weight in the between products. Additionally, by concatenating all annotations modestly-sized data set, and for the objective of average has several advantages. As we will next see, it lends itself targeting (corresponding), and take the TF-IDF-weighted average of their data), representing of GloVe [27] word vectors (pre-trained on Common Crawl web data), representing \( x_i^T = (x_1^T, x_2^T, \ldots, x_T^T) \). We select all \( x_i \) word vectors for which \( \hat{p}_{ik}^j = 1 (\hat{m}_{ik}^j = 1) \) for some \( k \), and concatenate them into one sequence. We then compute the TF-IDF scores for tokens in this sequence, find the D tokens with top TF-IDF scores \( D = 5 \) in our experiments), and take the TF-IDF-weighted average of their corresponding D GloVe vectors. We denote the resulting weighted-average vectors as \( \hat{p}_i \in \mathbb{R}^{300} \) and \( \hat{m}_i \in \mathbb{R}^{300} \) for purpose and mechanism annotations, respectively. We consider \( \hat{p}_i, \hat{m}_i \) to be target vectors we aim to predict for unseen texts.

Embedding (short) texts with a weighted-average of word vectors (such as with TF-IDF) can lead to surprisingly good results across many tasks [2]. Furthermore, in our case this simple weighted-average has several advantages. As we will next see, it lends itself to a straightforward machine learning setting, suitable for our modestly-sized data set, and for the objective of finding an overall vector representation that can be used in multiple ways, chiefly the computation of purpose-wise and mechanism-wise distances between products. Additionally, by concatenating all annotations and weighting by TF-IDF, we naturally give more weight in the average vector to words that are more frequently annotated – thus giving higher impact to words considered important by all annotators with respect to purpose/mechanism.

2.3.2 Learning purpose and mechanism. We now have \( N \) training product texts \( X_N = \{x_1, x_2, \ldots, x_N\} \), and \( N \) corresponding target tuples \( Y_N = \{p_1, m_1\}, \{p_2, m_2\}, \ldots, \{p_N, m_N\} \). We represent each \( x_i \) with its pre-trained GloVe vectors, \( w_i \). Our goal is to learn a function \( f(w_i) \) that predicts \( (p_i, m_i) \). To this end, we model \( f(\cdot) \) with a Recurrent Neural Network as follows. The network takes as input the variable-length sequence \( w_i \). This sequence is processed with a bidirectional RNN (BiRNN) [3] with one GRU layer. The BiRNN consists of the forward GRU \( \text{GRU} \) which reads the sequence \( w_i^T \) from \( w_i^1 \) to \( w_i^T \), and a backward GRU \( \text{GRU} \) which reads from \( w_i^1 \) to \( w_i^T \) thus in practice capturing the neighborhood of \( w_i^j \) from “both directions”:

\[
\vec{h}_i^j = \text{GRU}(w_i^j), \quad \vec{h}_i^j = \text{GRU}(w_i^j), \quad \vec{h}_i^j = [\vec{h}_i^j, \vec{h}_i^j],
\]

where we concatenate forward and backward GRU hidden states to obtain \( \vec{h}_i^j \), our representation for word \( j \) in product \( i \). In our case, we are interested in \( \vec{h}_i^j \) which captures the entire product text.

Next, let \( W_p \) and \( W_m \) be purpose and mechanism weight matrices, respectively. \( h_i^j \) is a shared representation of the document, which we now transform into two new vectors, \( \hat{p}_i, \hat{m}_i \), forming our purpose and mechanism predictions for product \( i \):

\[
\hat{p}_i = W_p h_i^T, \quad \hat{m}_i = W_m h_i^T.
\]

Parameters in this network are then tuned to minimize the MSE loss averaged over \( (p_i, m_i) \). In some scenarios, we may care more about predicting either purpose or mechanism, and in that case could incorporate a weight term in the loss function, giving more weight to either \( p_i \) or \( m_i \).

2.3.3 Purpose and Mechanism vector interpretations. Here, we give intuition about the kinds of representations extracted and the ability to interpret them with very simple tools. We first compute \( \hat{p}, \hat{m} \) for held-out product texts. Then, in the first approach to interpreting purpose and mechanism predictions, we find the top 10 GloVe word vectors \( w \) most similar to each of \( \hat{p}, \hat{m} \), among all vectors that appear in our vocabulary.

In the second approach, we aim to recover a set of 10 word vectors such that their sparse linear combination approximately gives \( \hat{p} \) or \( \hat{m} \). More formally, in the spirit of the sparse coding approach in [1], consider the collection of all word vectors in our vocabulary \( V \), \( w_1, w_2, \ldots, w_{|V|} \). We stack them into a matrix \( W \). We aim to solve the following optimization problem:

\[
\arg\min_a \|p_i - Wa\|^2 \quad s.t. \quad \|a\|_0 \leq 10,
\]

where \( a \) is a weight vector. Optimization can be done with the Orthogonal Matching Pursuit (OMP) [6] greedy algorithm.

In Table 1, we display some examples of applying these two simple methods, to product texts in test data (not seen during training). The first product is a yogurt maker machine, used for concentrating yogurt under heat, and to reduce time and energy. We observe that words selected as most related to our purpose vector representation include food, produce, concentrate, making, energy, reduce and also words that are typical in the language used in describing advantages of products in our data, such as especially, whole, enough, much. Mechanism words, are indeed overall of a much more “mechanical” nature, including liquid, heat, cooling, pump, steel, machine. In the other examples too, we observe the same pattern: Words selected as most closely-related to purpose or mechanism representations, using simple techniques, empirically appear to reflect corresponding properties in the product text, both in language and in deeper meaning.
We crowdsourced analogy with potential data quality issues, we recruited 3 workers for each. As typically done in the context of learning document representations, the key approach to quantitative evaluation is a down-stream task such as document classification. We now evaluate the predicted $p_i$, $m_i$ in the context of their ability to capture distances that reflect analogies, which is the primary focus of this paper. To do so, we first create a dataset of analogies and non-analogies.

### 3.1 Collecting analogies via crowdsourcing

We crowdsourced analogy finding within a set of about 8000 Quirky products. AMT crowd workers used our search interface to collect analogies – pairs of products – for about 200 seed documents. The search task is powered by a simple word-matching approach. To deal with word variants, we added lemmas for each word to the bag-of-words associated with each product. Each search query was also expanded with lemmas associated with each query term. Search results were ranked in descending order of number of matching terms. Median completion time for each seed was 7 minutes (workers could complete as many seeds as they wanted). Further, to deal with potential data quality issues, we recruited 3 workers for each seed (to allow for majority-vote aggregation).

Pairs that were tagged as matches became positive examples in our analogy dataset. However, coming up with negative examples was more difficult. Borrowing from information retrieval, we assume that people read the search results sequentially, and treat the implicitly rejected documents (i.e., documents that were not matches, despite appearing before matches) as negatives. It is important to remember that these documents are not necessarily real negatives. To further increase the chance that the document has actually been read, we restrict ourselves to the top-5 results.

**Challenges.** Getting workers to understand the concept of analogies and avoiding tagging products that are superficially similar (e.g., “both smartphone-based”) as analogies proved a challenge. To address this, we scaffolded the search task by first requiring workers to generate a schema (or “pattern”) to describe the core purpose and mechanism of the product, first in concrete terms, and then in more abstract terms (see an example pattern Figure 2). Workers were then instructed to find other products that matched the abstract schema they created. We found that this scaffolded workflow reduced the number of superficial matches; yet, a non-negligible portion of the pairs labeled as positive were either superficial matches or near analogies (i.e., analogies with many shared surface features), likely due to the strong tendency towards surface features in analogical retrieval [14]. Further, because products were multifaceted, search results may have been implicitly rejected even if they were analogous to the seed if the matching schema was different from the one initially identified by the worker.

### 3.2 Quantitative results

In Table 2, we present precision and recall @ K results. We rank all pairs in the test data (N = 2500, with training done on about 5500 products) based on their distances, according to various metrics, including our own. In summary, across all levels our approach outperformed the baselines, despite a challenging noisy setting. A considerable portion of test product pairs were tagged by workers as analogies despite having only surface similarity, creating mislabeled positive examples that favor the surface-based baselines. In addition to ranking by purpose-only and mechanism-only, we also concatenate both representations in a vector [p, m] for product $i$, and observe an overall improvement in results, although the “one-dimensional” use of either purpose or mechanism alone still beats the baselines. Using m only led to considerably better results when looking at precision @ top 1%, perhaps indicating a tendency by workers to find more mechanism-based analogies.

### 4 EVALUATION: IDEATION BY ANALOGY

Since a major application of the enhanced search and retrieval capabilities of analogy is enhanced creativity, we now evaluate the usefulness of our algorithms. We examine the degree to which our model’s retrieved output improves people’s ability to generate

---

<table>
<thead>
<tr>
<th>Product</th>
<th>Purpose words</th>
<th>Mechanism words</th>
</tr>
</thead>
<tbody>
<tr>
<td>A small yogurt maker machine for concentrating yogurt under heat and vacuum. Has a round base in drum with customized scoop, washable stainless steel drum parts. Reduce time and energy used.</td>
<td>Top similar: concentrate, enough, food, even, much, especially, reduce, produce, whole</td>
<td>Top similar: liquid, heat, cooling, pump, steel, machine, water, heating, electric</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A leash accessory with removable compartments for phone, cards cash, keys, poop bags, treats, bowl. Walk your dog and carry your essentials without pockets or a purse bag.</td>
<td>Top similar: bags, purse, wallet, carry, leash, backpack, pocket, dog, luggage</td>
<td>Top similar: leash, pouch, purse, pocket, bags, pockets, strap, compartment, backpack</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Product</th>
<th>Purpose words</th>
<th>Mechanism words</th>
</tr>
</thead>
<tbody>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Product</th>
<th>Purpose words</th>
<th>Mechanism words</th>
</tr>
</thead>
<tbody>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
</tbody>
</table>

---

### Table 1: Purpose and Mechanism vector interpretation examples. Descriptions shortened. Sparse coding shows only words with $|α| \geq 0.1$

<table>
<thead>
<tr>
<th>Product</th>
<th>Purpose words</th>
<th>Mechanism words</th>
</tr>
</thead>
<tbody>
<tr>
<td>A small yogurt maker machine for concentrating yogurt under heat and vacuum. Has a round base in drum with customized scoop, washable stainless steel drum parts. Reduce time and energy used.</td>
<td>Top similar: concentrate, enough, food, even, much, especially, reduce, produce, whole</td>
<td>Top similar: liquid, heat, cooling, pump, steel, machine, water, heating, electric</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
<tr>
<td>A cover placed on a car truck to protect from hail. Elastic perimeter to prevent wind from blowing under cover. Snap or velcro slits to open door without removing cover. Strong attachment so it won’t blow away. Inflatable baffles that cover the top, front windshield, side.</td>
<td>Top similar: storm, hail, rain, roofs, doors, wind, front, winds, walls</td>
<td>Top similar: roof, cover, lining, zipper, bottom, hood, plastic, flap, rubber</td>
</tr>
</tbody>
</table>
To generate inspirations for the redesign task, we start by using the learned purpose and mechanism representations $p_i, m_i$ for each document $i$ (in the test set) to apply rich queries to our corpus of products. In particular, assuming all vectors are normalized to unit euclidean norm, we can find pairs of products $i_1, i_2$ such that $d_p(p_{i_1}, p_{i_2}) = p_{i_1} \cdot p_{i_2}$ is high (near purpose), while $d_m(m_{i_1}, m_{i_2}) = m_{i_1} \cdot m_{i_2}$ is low (far mechanism). This type of reasoning, as discussed above, is a core element of analogical reasoning.

We take this idea one step forward by clustering by purpose and diversifying by mechanism. In more detail, we take a set of 2500 products not seen during training, and follow a simple and intuitive procedure as follows. Let $P_T$ denote our corpus of test-set products. Let $S$ denote the number of seed products we wish to use in our experiment. Let $M$ denote the number of inspirations we wish to produce for each seed $(1, \ldots, P)$.

### Clustering by purpose.
First, we find groups of products with similar purpose by clustering by our purpose representation.
- Run K-means ($K = 50$), based on vectors $p_i, \forall i \in P_T$. (Note that when all vectors are normalized, the euclidean norm on which K-means is based is equivalent to the cosine distance).
  - For each cluster $k \in \{1, \ldots, K\}$, compute an intra-distance measure (purpose homogeneity) $d_k$. We use the $MSE$. Prune clusters with less than $M$ instances. Rank clusters by $d_k$ in descending order, pick top $P$. Call this set of clusters $K_{top-purpose}$, with corresponding cluster centers $\hat{p}_1, \ldots, \hat{p}_P$.
  - For each cluster $k$ in $K_{top-purpose}$, select the product $i$ whose vector $p_i$ is nearest to the cluster center $\hat{p}_k$. This is our $k^{th}$ seed product, denoted by $s_k$.

### Result diversification by mechanism.
We now have a set of seed products, each with a corresponding cluster of products with similar purposes. Next, we need to pick $M$ inspirations per seed.

For each seed $s_k$, we have a set of candidate matches $U_{s_k}$, all from the same purpose cluster. We empirically observe that in...
the purpose-clusters $\mathcal{K}_{\text{top-purpose}}$ we generate, some vectors are highly similar to the seed with respect to mechanism, and some less so. In order to generate far-mechanism results for each seed from candidate set $\mathcal{U}_{sk}$, we now turn to diversification of results.

The problem of extracting a well-diversified subset of results from a larger set of candidates has seen a lot of work, prominently in the context of information retrieval (which is closely related to our setting). In our case, we assume to have found a set of relevant results $\mathcal{U}_{sk}$ according to purpose metric $d_p(\cdot, \cdot)$, and diversify by mechanism metric $d_m(\cdot, \cdot)$.

There are many ways to diversify results, mainly differing by objective function and constraints. Two canonical measures are the MAX-MIN and MAX-AVG dispersion problems [28]. In the former, we aim to find a subset $\mathcal{M} \subseteq \mathcal{U}_{sk}$ such that $|\mathcal{M}| = M$, and

$$\min_{m_i, m_j \in \mathcal{M}} d_m(m_i, m_j)$$

is maximized. In the latter, we aim to find a subset $\mathcal{M} \subseteq \mathcal{U}_{sk}$ such that $|\mathcal{M}| = M$, and

$$\frac{2}{M(M - 1)} \sum_{m_i, m_j \in \mathcal{M}} d_m(m_i, m_j)$$

is maximized. In other words, in the MAX-MIN problem we find a subset of products $\mathcal{M}$ such that the distance between the two nearest products is maximized. In the MAX-AVG problem, we find a subset such the average distance between pairs is maximized. Both problems admit simple greedy algorithms with constant-factor approximations [28]. We choose the MAX-MIN problem, since we want to avoid displaying too-similar results even once to a user (who may become frustrated and not proceed to read more inspirations).

We solve the problem using the GMM algorithm mentioned in [28]. Each iteration of GMM selects a candidate $m \in \mathcal{U}_{sk} - \mathcal{M}$ such that the minimum distance from $m$ to an already-selected product in $\mathcal{M}$ is the largest among remaining candidates in $\mathcal{U}_{sk} - \mathcal{M}$, where we measure distance according to our mechanism metric $d_m(\cdot, \cdot)$.

In our experiments, we set $P = 12, M = 12$, for 12 seeds and 12 matches each, respectively.

### 4.2 Experiment design

We recruited 38 AMT workers to redesign an existing product, a common creative task in design firms [36]. To ensure robustness of effects, the experiment included 12 different “seed” products. Participants were paid $1.5 for their participation.

To maximize statistical power, we utilized a within-subjects design with a single manipulated factor, inspiration_type:

- **ANALOGY**: participants receive 12 product inspirations retrieved by our method detailed above, using near-purpose far-mechanism clustering and diversification.
- **BASELINE: SURFACE**: participants receive product inspirations retrieved using TF-IDF, by finding the top 12 products similar to the seed. This baseline is meant to simulate current search engines.
- **BASELINE: RANDOM**: participants receive 12 product inspirations sampled at random from our product corpus.

Since we used a within-subjects design, participants completed the redesign task under each of the 3 inspiration_type conditions.

The order of conditions was counterbalanced to prevent order effects. To ensure unbiased permutations, we used the Fisher-Yates shuffle to assign seeds to conditions, so that every seed would be seen in all conditions (by different users).

Since prior work has shown that people benefit more from analogies if they receive them after ideation has begun [34], the ideation task proceeded in two phases: 1) generating ideas unassisted for one minute, then 2) receiving 12 inspirations and generating more ideas for 6 minutes. The inspirations were laid out in four pages, 3 inspirations per page, and the users could freely browse them.

Figure 3 provides an overview of the experiment and an excerpt from the data. The original task was to redesign an existing product, in this case a cell phone charger case. The SURFACE baseline retrieves products that are very phone-related (or case related). In contrast, our algorithm retrieves diverse results such as a human pulley-powered electricity generator suit. The bottom of the figure shows ideas generated by users in each condition. Interestingly,
the user exposed to our approach suggested a case that generates power using movement, potentially inspired by the suit.

4.3 Results

Measures. We are interested in the ability of our approach to enhance people’s ability to generate creative ideas. Following [29], we measured creative output as the rate at which a participant generates good ideas. We recruited five graduate students to judge each idea generated by our participants as good or not. Our definition of “good” follows the standard definition of creativity in the literature as a combination of novelty, quality, and feasibility [30]. Each judge was instructed to judge an idea as good if it satisfied all of the following criteria: 1) it uses a different mechanism/technology than the original product (novelty), 2) it proposes a mechanism/technology that would achieve the same purpose as the original product (quality), and 3) could be implemented using existing technology and does not defy physics (feasibility).

Agreement between the judges was substantial, with a Fleiss kappa of 0.51, lending our measure of creativity acceptable inter-rater reliability. The final measure of whether an idea was good or not was computed by thresholding the number of votes, so that good = 1 if at least K judges rated it as good. We report results for both liberal and strict settings K = 2, 3.

Evaluation. For K = 2, out of 749 total ideas collected, 249 ideas were judged as good by this measure. As mentioned above, we use the Fisher-Yates shuffle to assign seeds to conditions. To take a conservative approach, as a first step we look only at seeds that appeared across all three conditions (9 such seeds), to put the conditions on par with one another. By this slicing of the data, there were 208 good ideas. The proportion of good ideas in our condition was 46% (N = 105). Next was the random baseline with 37% (49), and finally the TF-IDF baseline achieved 30% (N=54). These results are significant by a $\chi^2$ proportion test (p < .01). We thus observe that both in terms of the absolute number of positively-rated ideas and in terms of proportions, our approach was able to generate a considerably large relative positive effect, leading to better ideas.

For K = 3 (the majority vote), out of 749 total ideas collected, 184 ideas were judged as good. Again, we start by looking only at seeds that appeared across all three conditions (9 such seeds). This leaves 154 good ideas. The proportion of good ideas in our condition was 38% (N = 118). Next-up was the random baseline with 22% (68), and finally the TF-IDF baseline achieved 21% (N=63), with p < .01. By looking at the more conservative majority-vote threshold, the observed effect of our method only increases.

Looking only at seeds that appeared across all conditions was a basic way to make sure we cancel out possible confounding factors. A more refined way is attempting to model these effects and condition on them, as follows.

We are interested in the likelihood that a given idea is good, or pr(good), as a function of inspiration condition. However, ideas are not independent: each participant generated multiple ideas, and ideas were proposed for different seeds. Failing to account for these dependencies would lead to inaccurate estimates of the effects of the inspirations: some participants may be better at generating ideas than others, while some seeds might be more easy/difficult than others. Therefore, we used a generalized linear mixed model, with a fixed effect of inspiration condition, and random effects of participant and seed (to model within-participant and within-seed dependencies between ideas).

For K = 2, our resulting model (with fixed effect of inspiration condition) yields a significant reduction in variance compared to a null model with no fixed effects, Likelihood ratio $\chi^2(3) = 67.96$, p < .01. The model also yields a reduction in Akaike Information Criterion (AIC), from 682.28 in the null model to 620.32, indicating that the improved fit to the data is not due to overfitting.

For K = 3, the model also yields a significant drop in variance compared to a null model, Likelihood ratio $\chi^2(3) = 92.38$, p < .01, with AIC dropping from 682.28 in the null model to 595.90.

As Figure 4 shows, our method led to a significantly higher probability for good ideas. For K = 2, pr(Good) = 0.71, 95% confidence interval = [0.48, 0.87] in our condition. TF-IDF had pr(Good) = 0.28 [0.16, 0.44], and random had pr(Good) = 0.27 [0.16, 0.41]. The advantages of the analogy condition over each baseline are both substantial and statistically significant, B = −1.81, p < .01 vs. TF-IDF, and B = −1.88, p < .01 vs. random. For K = 3, we had pr(Good) = 0.56, [0.36, 0.75]. TF-IDF had pr(Good) = 0.16 [0.08, 0.27], and random had pr(Good) = 0.14 [0.08, 0.24]. B = −1.94, p < .01 vs. TF-IDF, and B = −2.05, p < .01 vs. random.

Note that confidence intervals for the probability estimates are relatively wide (more so, unsurprisingly, for K = 2). Replications of this experiment, possibly with more data, could yield results somewhere in between, with more precise estimates on the true size of the effect. The main take-away of this study is that our approach yields a reliable increase in participants’ creative ability.

5 DISCUSSION AND CONCLUSION

In this paper, we sought to develop a scalable approach to finding analogies in large, messy, real-world datasets. We explored the potential of learning and leveraging a weak structural representation (i.e., purpose and mechanism vectors) for product descriptions. We leverage crowdsourcing techniques to construct a training dataset with purpose/mechanism annotations, and use an RNN to learn purpose and mechanism vectors for each product. We demonstrate that these learned vectors allow us to find analogies with higher precision than traditional information-retrieval similarity metrics like TF-IDF, LSA, GloVe and LDA.

Our ideation evaluation experiment further illustrates the effectiveness of our approach: participants had a higher likelihood of generating good ideas for the redesign ideation task when they...
received inspirations sampled by our analogy approach (tuned to be similar in purpose, but different in mechanism), compared to a traditional (TF-IDF) baseline or random sampling approach. From a psychological perspective, the benefits of our inspirations are likely due to our approach’s superior ability to sample diverse yet still structurally similar inspirations, since diversity of examples is a known robust booster for creative ability [8]. The TF-IDF approach yielded inspirations likely to be relevant, but also likely to be redundant and homogenous, while the random sampling approach yields diversity but not relevance.

While moving to a “weak” structural representation based on purpose and mechanism significantly increased the feasibility of analogy-finding, extensions may be necessary to generalize to other domains besides product descriptions. For example, our purpose and mechanism vectors did not distinguish between higher and lower level purposes/mechanisms, or core/peripheral purposes/mechanisms, and also did not encode dependencies between particular purposes/mechanisms. These are potentially fruitful areas for future work and may be especially important when moving from relatively simple product descriptions to more complex data such as scientific papers, in which purposes and mechanisms can exist at multiple hierarchical levels (e.g., “accelerate innovation” vs. “learn mechanisms, and also did not encode dependencies between partic-

Acknowledgments. The authors thank the anonymous reviewers for their helpful comments. This work was supported by NSF grants CHS-1526665, IIS-1149797, IIS-1217559, Carnegie Mellon’s Web2020.

REFERENCES


