Web Appendix

Automation Assemblages in the Internet of Things: Discovering Qualitative Practices at the Boundaries of Quantitative Change

Thomas P. Novak
Donna L. Hoffman

This eight-part Web Appendix provides methodological details of the paper “Automation Assemblages in the Internet of Things”:

- Web Appendix A. Learning and Validating Word Embeddings
- Web Appendix B. Applet Embeddings and Applet Similarity
- Web Appendix C. Learning Automation Practices
- Web Appendix D. Visualizing the Realized Possibility Space
- Web Appendix E. Thematic Analysis
- Web Appendix F. Practice Growth Potential
- Web Appendix G. Visualizing the Full Possibility Space
- Web Appendix H. Topic Models as an Alternative Approach to Word Embeddings

In addition, an Excel spreadsheet containing a detailed summary of the 127 automation practices discussed in the paper is available online at the Open Science Foundation (OSF): https://osf.io/vh48r/

Interactive hoverplots for the following figures from the paper and Web Appendix are available online at the Open Science Foundation (OSF): https://osf.io/vh48r/

- Interactive figure 3 - Hoverplot of Realized Applets Colored by 127 HDBSCAN Clusters (interactive version of figure 3 from the paper)
- Interactive figure 7 - Territorialized Practice Growth Over Time (interactive version of figure 7 from the paper)
- Interactive figure 8 - Boxplots of Cluster Size Growth Potential by 4 Process Groups (interactive version of figure 8 from the paper)
- Interactive figure 9a - Hoverplot of the Full possibility Space (interactive version of figure 9a from the paper)
- Interactive figures C3 through C9 - Hoverplots of seven different clustering solutions (interactive versions of figures C3 through C9 from the Web Appendix)
WEB APPENDIX A

LEARNING AND VALIDATING WORD EMBEDDINGS

A1. IFTTT Data

Users create applets through either the IFTTT app or website. The diagram in figure A1 shows the point-and-click process that requires no programming knowledge. As users create if-then applets, IFTTT stores all of the information needed to describe what the applet does. This includes the various trigger and action services (e.g., Twitter, Phillips Hue), as well as the specific trigger and action phrases (e.g., new tweet by you, turn on lights). The six fields of structured text representing the services and phrases are shown in table 2 of the paper.

FIGURE A1

PROCESS FOR CREATING AN IFTTT APPLET

IFTTT also captures whether an applet is published so that it is publicly shared vs. private (we include only published applets), and the date the applet was created by a user. This is the basic information that we use in our computational analysis of the 20,675 unique published applets. For our interpretative analysis, we use additional information from the larger set of 331,391 published applets that includes all the identical copies of the 20,675 unique applets that were created and published by multiple users. This additional information includes the number of different users that published each of the 20,675 applets, as well as the optional supplemental descriptive text that was available for 36 of the 331,391 published applets. For analyses involving the full possibility space, we use the six fields of structured text from 895,575 unrealized applets. Figure A2 diagrams the structure of all of the data we use.
FIGURE A2
IFTTT DATASETS USED IN COMPUTATIONAL AND INDUCTIVE ANALYSES

Published Realized Applets (includes copies) 
n=331,391

Realized Applets (unique) 
n=20,675

Unrealized Applets 
n=895,575

Copies of Territorialized Realized Applets 
n=278,716

36.3% of Copies of Territorialized Realized Applets have Optional Descriptive Text 
n=101,174

(Data for Inductive Analysis)

Territorialized Realized Applets (HDBSCAN identifies 127 clusters) 
n=15,417

Deterritorialized Realized Applets (HDBSCAN noise points) 
n=5,258

Territorialized Unrealized Applets (predicted to be in a cluster if realized) 
n=212,527

Deterritorialized Unrealized Applets (predicted to be noise points if realized) 
n=683,048

Data for Computational Analysis
A2. Learning Word Embeddings

Our text corpus was the six fields of structured text for the 20,675 realized applets. We removed punctuation, converted all text to lowercase, removed stop words, lemmatized, and tokenized. We only removed extraneous stop words that would not alter the meaning of a phrase. Our stop word list included “a”, “an”, “the”, “and”, “is”, “to”, “by”, “of”, “in”, “any”, “at”, “from”, “with” and “has”. We followed the same logic for lemmatizing, recoding only those applet variants that in our judgment would have the same meaning lemmatized or not. This involved recoding past and present to the infinitive (e.g., armed and arming lemmatized to arm), and plural to singular (e.g., displays lemmatized to display). Following these text cleaning steps, we tokenized each text string.

The six fields of text in table 2 designate the broader category of each trigger and action component (e.g., Smart Hubs and Systems), the components themselves (e.g., Amazon Alexa), and the capacity that each component exercises as either a trigger or an action (e.g., say a specific phrase). For each applet, the text includes the words that appear in the six fields, with the fields ordered by what we term an “outside-in” (i.e., broad to narrow) sequence. By outside-in, we mean that the beginning and end of the ordered text use a broad level of text description (the general trigger and action categories), and the interior portion of the text uses a narrow level (the specific trigger and action event). In contrast, there could be a reversed “inside-out” (i.e., narrow to broad) sequence. Table A1 provides an example of outside-in and inside-out sequences for the Amazon AlexaIFTTT applet discussed in the paper. The outside-in sequence is a more natural fit with an assemblage theory view of interaction because the juxtaposition of text for trigger and action capacities in the outside-in sequence contextualizes, rather decontextualizes, the role of paired capacities. However, we test these two text sequence orders below in an external evaluation of the embeddings.

We use the gensim API for the word2vec model in Python (Řehůřek and Sojka 2010) to learn embeddings for each of the 1383 words in our corpus. This computational linguistics method is a highly efficient recurrent neural network language model that represents words based on the idea of the distributional hypothesis (Harris 1954, Mikolov, Yih, and Zweig 2013). This hypothesis argues that words that appear in similar contexts are likely to have similar meanings, so that we can “know a word by the company it keeps” (Firth 1957, p. 11). The model architecture for word2vec is straightforward (Mikolov, et.al. 2013a). Briefly, word2vec predicts context words given a target word based on the dot products between the vectors for the context and target words. The higher the dot product between the two vectors, the more similar they are. This reveals the intuition of using cosine similarity as the preferred similarity metric since cosine similarity is simply the normalized dot product.

Word2vec requires choosing several hyperparameters such as context window size and number of features, a process known as “hyperparameter tuning.” Since the sequence in which structured text fields are ordered can also affect the results, we tuned the model by learning word embeddings for 24 different combinations of three factors: 1) context window size (two levels: 5 and 10); 2) number of features extracted for the word embeddings (six levels: 10, 25, 50, 100, 200, and 300 dimensional vectors); and 3) text sequence. For the text sequence factor, six fields of text were concatenated in either an “outside-in” (broad to narrow) or an “inside-out” (narrow to broad) sequence.
TABLE A1

OUTSIDE-IN VERSUS INSIDE-OUT SEQUENCES OF TEXT FIELDS

A) “Outside-In” Sequence of IFTTT Applet Text Fields

<table>
<thead>
<tr>
<th>Trigger Category</th>
<th>Trigger Service</th>
<th>Trigger Event</th>
<th>Action Event</th>
<th>Action Service</th>
<th>Action Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Hubs and Systems</td>
<td>Amazon Alexa</td>
<td>Say a Specific Phrase</td>
<td>Turn on a Color Loop</td>
<td>Philips Hue</td>
<td>Lighting</td>
</tr>
</tbody>
</table>

1The outside-in sequence contains adjacent words for the exercised capacities of the trigger and action components (i.e., “say a specific phrase, turn on a color loop”); this gives a clear indication of what the applet is doing and what experience is likely to emerge. Because the trigger and action capacities are contextualized by the adjacency of their respective text, the outside-in sequence is consistent with an assemblage theory view of interaction occurring through paired capacities.

B) “Inside-Out” Sequence of IFTTT Applet Text Fields

<table>
<thead>
<tr>
<th>Trigger Event</th>
<th>Trigger Service</th>
<th>Trigger Category</th>
<th>Action Category</th>
<th>Action Service</th>
<th>Action Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Say a Specific Phrase</td>
<td>Amazon Alexa</td>
<td>Smart Hubs and Systems</td>
<td>Lighting</td>
<td>Philips Hue</td>
<td>Turn on a Color Loop</td>
</tr>
</tbody>
</table>

2The inside-out sequence contains adjacent words for the categorical properties of the two components (i.e., “smart hubs and systems, lighting”); this does not as clearly indicate what the applet is accomplishing, or what experience is likely to emerge. Because the trigger and action capacities are decontextualized by the separation of their text, the inside-out sequence is not consistent with an assemblage theory view of interaction.

For all 24 combinations, we used the skip-gram formulation of word2vec (Mikolov, et al. 2013b) trained with a negative sampling parameter of 5, and a down-sampling threshold of .001. Window sizes of (plus or minus) 5 and 10 tested different context sizes. After removing stop words, the number of words in an applet ranged from 7 to 23, with a median of 12. For a median length applet, a window size of 5 captures about half of the text for a word at the beginning or end, and nearly all the text for a word in the middle. A window size of 10, on the other hand, captures nearly all the text for a word in a median length applet, regardless of the word’s position. The number of features (dimensions) extracted were 10, 25, 50, 100, 200, and 300. Once learned, the word embeddings, v, are normalized to a length of one (i.e., ||v|| = 1). While 300 feature embeddings are typical for large corpuses such as text from Wikipedia or Google News, we expected that fewer features would be needed for our relatively small and highly structured text corpus of IFTTT applets.

From the 24 combinations, we used an extrinsic evaluation, described below, to determine that the best word embeddings were based on a 25-feature word2vec model using a context window size of 10, applied to an outside-in text sequence. We then performed an intrinsic validation to assess the face validity of the best solution.
Before discussing validation, we note that there are other approaches to creating word embeddings besides word2vec, including GLoVe (Pennington, Socher and Manning 2014), FastText (Joulin, Bojanowski, Douze, Jégou and Mikolov 2016), ELMo (Peters et al. 2018) and BERT (Devlin et al. 2018); for comparisons of these approaches see Khattak et al. (2019) and Si et al. (2019). While such approaches may prove advantageous in an extremely large unstructured natural language corpus, we do not expect significant differences in substantive outcomes for our fairly small corpus of highly structured text. In fact, section A5 below shows that embeddings obtained from word2vec and FastText are very similar for our data. Nevertheless, this is a technical topic for future research. At the same time, in contrast to our bottom-up approach of building applet embeddings as the sum of individual word embeddings for an applet’s trigger and action, we could directly use approaches like doc2vec to learn embeddings for a sentence - in our case an applet - as a whole, without considering the embeddings of the words in the sentence (Le and Mikolov 2014). However, there are two reasons why word embeddings are preferable to sentence embeddings for our data. First, unlike the sentences in a Wikipedia article or product review, which have an unambiguous ordering, applet “sentences” do not have a fixed contextual relationship to the other sentences in the corpus. Second, using word embeddings allows us to create predictions for the 98% of applets that are unrealized, which would not be possible using sentence embeddings.

A3. Extrinsic Evaluation of Word Embeddings

We used extrinsic evaluation (e.g., Schnabel et al 2015) to identify which of 24 combinations of three factors (window size, number of features, and text sequence) produced the best word embeddings in our corpus of 1383 unique words occurring in 20,675 IFTTT applets. Extrinsic evaluation uses word embeddings as input features to a downstream task, and measures metrics specific to that task. Our downstream task was how well a reduced two-dimensional UMAP representation of the high-dimensional embeddings (McInnes, Healy and Melville, 2018) predicted the proportion of time each word appeared in an applet’s trigger versus in its action (see figure A3 for the distribution of this proportion). Of the 1383 unique words in the corpus, 45% appeared exclusively in an applet’s trigger, 18% appeared exclusively in an applet’s action, and 37% appeared in both an applet’s trigger and action. For each of the 24 combinations of three factors described above, we fit 18 different UMAP models varying the n_neighbors parameter from 5, 10, 20, 30, 40, and 50, and the min_dist parameter from .05, .10, and .20. These 18 UMAP models produced a family of low dimensional solutions that varied by whether they emphasized local or global aspects of the data. All UMAP models produced two dimensional solutions based on the cosine distances among the embeddings.

For each of the 24 combinations of three factors (window size, number of features, and text sequence), figure A4 plots a boxplot showing $R^2$ from Random Forest Regression (top panel) and Support Vector Regression (bottom panel), both with 10-fold cross-validation. All models were fit with Python’s scikit-learn machine learning package. Models used the two UMAP coordinates to predict the proportion of time each word appeared in an applet’s trigger versus in its action. In both analyses, the highest maximum and medium $R^2$ was obtained using UMAP coordinates from an outside-in text sequence with a window size of 10, and 25 feature word embeddings.
FIGURE A3

PERCENT OF TIMES THE 1383 UNIQUE WORDS IN CORPUS APPEAR IN AN APPLET’S TRIGGER

Horizontal axis: The percent of times a word in the corpus appears in the trigger of an applet.

Vertical axis: For each value on the horizontal axis, the percent of the 1383 words with this value.
FIGURE A4

EXTRINSIC VALIDATION OF WORD EMBEDDINGS

Random Forest Regressions

Support Vector Regressions

Boxplots show $R^2$ predicting the proportion of times words in the corpus are in the applet’s trigger vs. action, from coordinates from different UMAP solutions. Each boxplot represents 18 UMAP solutions (for different combinations of n_neighbors and min_dist hyperparameters), with a separate boxplot for each of 24 combinations of parameters (2 text orders, 2 word2vec window sizes, and 6 levels of embedding features).
A4. Intrinsic Evaluation of Word Embeddings

To illustrate the validity of the meanings of the words as learned by word2vec, we performed an intrinsic evaluation (Schnabel et. al. 2015) to assess the relatedness and coherence of words that are similar, based on the cosine similarity of their embeddings. A typical approach for intrinsic evaluation is to visualize the high-dimensional word embeddings in a low-dimensional space and subjectively assess the extent to which words which should be related are close together. We use UMAP, a new technique for nonlinear dimensionality reduction (McInnes, Healy and Melville, 2018) to produce the low-dimensional representation of the 25-dimensional feature space of the word embeddings. Figure A5 plots the 1383 words according to the two dimensions of the UMAP solution. A number of select words are labeled. Words will be close to each other in the two-dimensional plot if they have a high cosine similarity (similar meaning) based on their 25-dimensional vectors. Words that are close together, for example “search”, “shopping”, “product”, and “buy”, provide evidence of the face validity of the learned meanings of the words.

A5. Comparison of word2vec to FastText

To ensure the embeddings are not highly dependent on choice of methodology, we compared our word2vec embeddings with FastText embeddings (Joulin, Bojanowski, Douze, Jégou and Mikolov 2016). Gensim 4.0.1 was used to learn FastText embeddings. To maintain comparability with our solution, we used the same FastText parameters that we used for word2vec: window size=10, vector size=25, skip-gram formulation, negative sampling parameter = 5, and minimum word count = 1.

We then created a cosine distance matrix for the 1383 words using the word2vec embeddings, and a cosine distance matrix for the 1383 words using the FastText embeddings. The Mantel correlation (Mantel 1967) between distance matrices based on word2vec and FastText embeddings was .831, indicating a very high degree of correspondence between the semantic structure of the two solutions.

One concern may be that since our minimum word count was one, embeddings for infrequently appearing words may not be as stable as those for frequently appearing words. Therefore, we also calculated cosine distance matrices for the subset of 515 words appearing 50 or more times in the corpus, and the subset of 868 words that appeared less than 50 times. The Mantel correlation between distances based on word2vec and FastText embeddings for words occurring 50 or more times was .841. The Mantel correlation for words less than 50 times was .825. This suggests that the structure of word similarity based on word2vec and FastText embeddings is not greatly affected by word frequency.
The 1383 words in the text corpus of IFTTT applets are plotted using two dimensions from a UMAP solution of 25 feature embeddings, window size of 10, and outside-in text order. Circle size is proportional to the number of times the word appears in the corpus. Circle color indicates the proportion of times the word appeared in the applet’s trigger (blue) or action (red).
WEB APPENDIX B

APPLET EMBEDDINGS AND APPLET SIMILARITY

B1. Calculating Applet Embeddings

As described in Web Appendix A2, 25-feature word embeddings, \( \mathbf{v} \), were learned for each of the 1383 words in our corpus. We then used these word embeddings to calculate embeddings for each applet’s trigger and action. These trigger and action embeddings are, respectively, the normalized average of the embeddings \( \mathbf{v}_i \) for each of the \( n_t \) words in the applet’s trigger, and the normalized average of the embeddings \( \mathbf{v}_j \) for each of the \( n_a \) words in the applet’s action:

\[
(1a) \quad \mathbf{t} = \frac{\mathbf{t}^*}{\| \mathbf{t}^* \|}, \text{ where } \mathbf{t}^* = \frac{\sum \mathbf{v}_i}{n_t}, \text{ so that } \| \mathbf{t} \| = 1
\]

\[
(1b) \quad \mathbf{a} = \frac{\mathbf{a}^*}{\| \mathbf{a}^* \|}, \text{ where } \mathbf{a}^* = \frac{\sum \mathbf{v}_j}{n_a}, \text{ so that } \| \mathbf{a} \| = 1
\]

The embedding for each applet was then obtained by concatenating the trigger (\( \mathbf{t} \)) and action (\( \mathbf{a} \)) embeddings. We represent this concatenation as (\( \mathbf{t}, \mathbf{a} \)). Web Appendix B3 explains our rationale for representing applets as the concatenation of their trigger and action embeddings.

B2. Derivation of Applet Similarity in the Compositional Approach

In the compositional approach, the cosine similarity of the concatenated representation of two applets, \( \cos( (\mathbf{t}_1, \mathbf{a}_1), (\mathbf{t}_2, \mathbf{a}_2) ) \), is equal to the average of the cosine similarities of their trigger and action embeddings, \( \frac{\cos(\mathbf{t}_1, \mathbf{t}_2) + \cos(\mathbf{a}_1, \mathbf{a}_2)}{2} \). This is derived as follows. Since \( \mathbf{t} \) and \( \mathbf{a} \) are unit length vectors, \( \| (\mathbf{t}, \mathbf{a}) \| = \sqrt{2} \). Then, cosine similarity between two applets represented as (\( \mathbf{t}_1, \mathbf{a}_1 \)) and (\( \mathbf{t}_2, \mathbf{a}_2 \)) can be re-expressed as:

\[
(2) \quad \cos((\mathbf{t}_1, \mathbf{a}_1), (\mathbf{t}_2, \mathbf{a}_2)) = \frac{( (\mathbf{t}_1, \mathbf{a}_1) \cdot (\mathbf{t}_2, \mathbf{a}_2) )}{\| (\mathbf{t}_1, \mathbf{a}_1) \| \cdot \| (\mathbf{t}_2, \mathbf{a}_2) \|}
\]

\[
= \frac{ ( (\mathbf{t}_1, \mathbf{a}_1) \cdot (\mathbf{t}_2, \mathbf{a}_2) ) }{2}
\]

\[
= \frac{ (\mathbf{t}_1 \cdot \mathbf{t}_2 + \mathbf{a}_1 \cdot \mathbf{a}_2) }{2}
\]

\[
= \frac{(\cos(\mathbf{t}_1, \mathbf{t}_2) + \cos(\mathbf{a}_1, \mathbf{a}_2))}{2}
\]
B3. Problematic Interpretation of Applet Similarity When Applet Embeddings are Constructed Using a Non-Compositional Approach

At first glance, it may appear simpler and reasonable to take a non-compositional approach and define an applet embedding, $x$, as the normalized average of all of the embeddings, $v$, for each of the $n$ words in the applet:

$$x = x^* / \|x^*\|, \text{ where } x^* = \sum v / n,$$  \hspace{1cm} (3)

so that $\|x\| = 1$

One problem with doing so is that expression (3) unequally weights the words embeddings, $v$, by the number of words in the trigger, $n_t$, and the action, $n_a$. This can be seen by re-expressing as:

$$x^* = \sum v / n = (\sum_{i=1, n_t} v + \sum_{i=1, n_a} v) / n \hspace{1cm} (4)$$

Instead, we can produce an applet representation, $x_{EQ^*}$, that equally weights the contributions of word embeddings in trigger and action as:

$$x_{EQ^*} = (n/n_t \sum_{i=1, n_t} v + n/n_a \sum_{i=1, n_a} v) / n \hspace{1cm} (5)$$

Thus, creating an applet embedding as the sum of trigger and action embeddings, $t + a$, is a weighted variation of the non-compositional approach of creating applet embeddings.

Expression (5) shows that an equal weighting of word embedding by trigger and action corresponds to the sum of non-normalized embeddings $t^*$ and $a^*$. To simplify exposition, and in the same way that word embeddings, $v$, are normalized to a length of one, we can use the sum of normalized trigger and action embeddings, $t + a$. In this normalized non-compositional approach, when we represent an applet by the sum of its trigger and action embeddings, the cosine similarity between two applets becomes:
\[(6) \cos(\mathbf{t}_1 + \mathbf{a}_1, \mathbf{t}_2 + \mathbf{a}_2) =
\]
\[
\frac{(\mathbf{t}_1 + \mathbf{a}_1) \cdot (\mathbf{t}_2 + \mathbf{a}_2)}{||\mathbf{t}_1 + \mathbf{a}_1|| \cdot ||\mathbf{t}_2 + \mathbf{a}_2||} =
\]
\[
\frac{(\mathbf{t}_1 \cdot \mathbf{t}_2 + \mathbf{a}_1 \cdot \mathbf{a}_2 + \mathbf{a}_1 \cdot \mathbf{t}_2 + \mathbf{t}_1 \cdot \mathbf{a}_2)}{\sqrt{(\mathbf{t}_1 \cdot \mathbf{t}_1 + 2(\mathbf{t}_1 \cdot \mathbf{a}_1) + \mathbf{a}_1 \cdot \mathbf{a}_1) \cdot (\mathbf{t}_2 \cdot \mathbf{t}_2 + 2(\mathbf{t}_2 \cdot \mathbf{a}_2) + \mathbf{a}_2 \cdot \mathbf{a}_2)}} =
\]
\[
\frac{(\cos(\mathbf{t}_1, \mathbf{t}_2) + \cos(\mathbf{a}_1, \mathbf{a}_2) + \cos(\mathbf{t}_2, \mathbf{a}_1) + \cos(\mathbf{t}_1, \mathbf{a}_2))}{\sqrt{2}} \cdot \sqrt{1 + \cos(\mathbf{t}_1, \mathbf{a}_1) \cdot \cos(\mathbf{t}_2, \mathbf{a}_2)} =
\]

As shown in expression (6), the interpretation of cosine similarity for applets represented as \(\mathbf{t} + \mathbf{a}\) using the non-compositional approach is considerably more complex than the interpretation of cosine similarity for applets represented as \((\mathbf{t}, \mathbf{a})\) in expression (B2) using the compositional approach. Let’s consider the three terms identified in expression (6). Numerator term 1 corresponds to the sum of cosine similarities of trigger and action embeddings of the two applets. This is exactly the compositional cosine similarity that we are using in our paper. Numerator term 2, however, represents the cosine similarity of the trigger of one applet with the action of the other applet, that is, \(\cos(\mathbf{t}_2, \mathbf{a}_1)\) and \(\cos(\mathbf{t}_1, \mathbf{a}_2)\). It is unclear whether these cosine similarities are substantively meaningful. Last, denominator term 3 is a complex function of the within-applet similarity of trigger and action for the two applets: \(\cos(\mathbf{t}_1, \mathbf{a}_1)\) and \(\cos(\mathbf{t}_2, \mathbf{a}_2)\), and their cross product \(\cos(\mathbf{t}_1, \mathbf{a}_1) \cdot \cos(\mathbf{t}_2, \mathbf{a}_2)\). In the non-compositional approach, we wind up with cosine similarities that combine three terms in a complex and unclear manner. For this reason, we follow the compositional approach and represent applets with their concatenated trigger and action embeddings \((\mathbf{t}, \mathbf{a})\), since this produces a much more straightforward interpretation of cosine similarity.
WEB APPENDIX C
LEARNING AUTOMATION PRACTICES

C1. Identification of Territorialized and Deterritorialized Automation Practices

A total of 13 HDBSCAN solutions were run, varying the min_cluster_size parameter from 5 to 500. Note that in the paper, we only report parameter values from 50 to 350, since before 50, the solutions did not meet our repetition with difference assumption and after 350 there were no further changes in how the higher-order practices grouped together. All solutions set a second hyperparameter, min_samples, to 1, to minimize the number of applets classified as noise points (i.e., unassigned to any cluster). The distance between applets was calculated as the Euclidean distance between the concatenated trigger and action embeddings. Since squared Euclidean distance for unit length vectors is proportional to their cosine distance, Euclidean distance and cosine distance produce nearly identical distance matrices, with a Mantel correlation between the two matrices of .992. The two metrics also provide nearly equivalent HDBSCAN solutions, with an Adjusted Rand Index (Hubert and Arabie 1985) of .995 between cluster solutions using cosine and Euclidean distance. The choice of Euclidean distance was made for a technical reason. HDBSCAN does not natively support the cosine distance metric, although one can use a precomputed distance matrix based on any metric as input. However, since only the natively supported metrics allow HDBSCAN to produce the full range of analyses of which it is capable, we need to use Euclidean distance.

Table C1 presents the number of HDBSCAN clusters and noise points for each of the values of the min_cluster_size parameter. Generally speaking, as the minimum cluster size increases, the number of clusters decreases, and the number of noise points increases. We assume that at least 50 unique variations of an applet are required for the automation practice assemblage to be territorialized. Note that if we only require 25 unique variations (min_cluster_size=25), then the number of clusters more than doubles from 127 to 278. On the other hand, increasing the minimum number of unique variations to 100 cuts the number of clusters in half, to only 58.

In interpreting the HDBSCAN clusters, we begin with the 127 clusters found from the solution with min_cluster_size = 50. This solution has a fairly fine degree of granularity that allows us to see nuanced differences between automation practice assemblages. Then, we use solutions with larger values of min_cluster_size to identify higher-order practice assemblages composed of various groups of these 127 clusters. In essence, we dial the knob higher to require more repetition with difference and evaluate each solution to visualize whether and how automation applets in the 127-cluster solution join together at each progressively higher level. We found no advantage to dialing the knob lower to require less repetition with difference, since the 127 clusters were generally very homogeneous. But, in other applications, doing so may make sense.

Figure C1 plots the HDBSCAN solutions using coordinates from a two-dimensional UMAP. Interactive versions of these plots were used to examine how the cluster structure changed as min_cluster_size was varied from 50 to 350. Figure 4 in the paper displays the hierarchical structure.
<table>
<thead>
<tr>
<th>Min_cluster_size parameter value</th>
<th>Number of clusters</th>
<th>Number of noise points</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1469</td>
<td>3440</td>
</tr>
<tr>
<td>10</td>
<td>662</td>
<td>4151</td>
</tr>
<tr>
<td>25</td>
<td>278</td>
<td>4949</td>
</tr>
<tr>
<td>50</td>
<td><strong>127</strong></td>
<td><strong>5258</strong></td>
</tr>
<tr>
<td>75</td>
<td>79</td>
<td>5238</td>
</tr>
<tr>
<td>100</td>
<td>58</td>
<td>5452</td>
</tr>
<tr>
<td>150</td>
<td>40</td>
<td>6388</td>
</tr>
<tr>
<td>200</td>
<td>30</td>
<td>5941</td>
</tr>
<tr>
<td>250</td>
<td>22</td>
<td>6936</td>
</tr>
<tr>
<td>300</td>
<td>19</td>
<td>7755</td>
</tr>
<tr>
<td>350</td>
<td>14</td>
<td>7921</td>
</tr>
<tr>
<td>400</td>
<td>12</td>
<td>8681</td>
</tr>
<tr>
<td>500</td>
<td>11</td>
<td>8205</td>
</tr>
</tbody>
</table>
FIGURE C1

HDBSCAN SOLUTIONS FOR MIN_CLUSTER_SIZE FROM 50 TO 350
FIGURE C1 (CONTINUED)

HDBSCAN SOLUTIONS FOR DIFFERENT VALUES OF MIN_CLUSTER_SIZE
We compared the 127-cluster solution we obtained from HDBSCAN with solutions from KMeans, Wards, single linkage, complete linkage, and average linkage, all commonly used clustering algorithms in marketing (Punj and Stewart, 1983), using the Python scikit-learn 0.22.1 package. Unlike HDBSCAN, which identifies unclustered noise points, all these other methods force assignment of every observation into a cluster. Also, unlike HDBSCAN, which identifies the optimal number of clusters, these other methods require that the number of clusters be specified in advance. In addition to the marketing comparisons, we compared HDBSCAN to Affinity Propagation, a recently developed graph-based approach (Frey and Dueck, 2007), using the scikit-learn package. Like HDBSCAN, Affinity Propagation does not require the number of clusters to be specified in advance, although it does force the assignment of all observations into a cluster.

As with HDBSCAN, for each method we clustered the 50-dimensional representations of the 20,675 unique realized applets using either the Euclidean metric (for KMeans, Wards, and Affinity Propagation) or cosine metric (or single, complete, and average linkage), recalling that for our vector embeddings, Euclidean and cosine metrics produce comparable results. For comparability with our HDBSCAN solution, we obtained 127 clusters for KMeans, Wards, single linkage, complete linkage, and average linkage. Using the default preference and damping parameters, Affinity Propagation identified 941 clusters. To visualize the seven different clustering solutions, we plotted them using the coordinates from a two-dimensional UMAP solution of the 20,675 realized applets, as described in the paper and in Web Appendix D. Our common UMAP solution was based on the cosine distances of the 50-dimensional representations of the realized applets.

For simplicity, we present only the region of the realized possibility space corresponding to clusters for the “long finger control” personal automation processes, as highlighted on the center right of figure C2. The long finger control region of the possibility space is identified in figure 5 of the paper. Figures C3 through C9 respectively plot the long finger control clusters from HDBSCAN, KMeans, Wards, single linkage, complete linkage, average linkage, and Affinity Propagation. Interactive figures C3 through 3.9 (available online at OSF), provide parallel interactive hoverplots showing all 20,675 applets, for each of the seven clustering solutions.

There are no objective criteria that can be used to compare the different clustering solutions. Therefore, we compared the seven solutions based on their ability to visually isolate clusters of applets in the UMAP plots according to how well 1) a cluster corresponding to an automation practice assemblage has clear boundaries that define its shape, 2) the solution clearly separates one automation practice from another, and 3) whether the cluster boundaries conform to whatever the shape of the practice boundaries might be, as opposed to forcing the cluster boundaries into fixed, globular, shapes. Note that the UMAP and clustering solutions are independent of each other. This means we are not clustering the 20,675 applets on the two UMAP coordinates, but instead determining the UMAP and seven clustering solutions separately from the 50-dimensional representations of the 20,675 realized applets. Since we are assuming that the two-dimensional UMAP solution is a reasonable approximation of the manifold of the
realized possibility space, a good clustering solution should be clearly represented using the UMAP plot.

The HDBSCAN solution in figure C3, showing the long finger control clusters, shows clear and distinct separation of automation practices. The shape of the boundaries varies from compact and globular to irregular and elongated. Because HDBSCAN has identified some applets as unclustered noise points, there is a clear separation from one practice to another. This is not the case in figures C4 through C9. Because these other methods force every applet into a cluster, the boundaries between clusters are often not clearly defined, even overlapping in some cases. Single linkage (figure C6) is problematic, placing most observations into a single large cluster with a number of smaller clusters containing a few as one applet. In most of the solutions, the groups are more partitions than clusters, with abrupt dividing lines from one cluster to another. Affinity Propagation (figure C9) produces clearer separation of groups in some cases, but in other cases the dividing lines are abrupt. Affinity Propagation also produces a very large number of small globular-shaped clusters, including many clusters with only a single applet. For our data, it is clear that the HDBSCAN solution produces the best set of clearly defined boundaries reflecting territorialized automation practices.

A caveat is in order. We are not claiming that HDBSCAN is the only clustering method capable of clearly revealing the boundaries of automation practices. Other clustering methods that do not require every data point to be clustered could also be explored. These include Mean Shift, the earlier DBSCAN, and OPTICS, all available in scikit-learn, as well as robust single linkage (e.g., McInnis 2016). We did try Mean Shift but found that it assigned all of our applets into a single cluster. While DBSCAN and OPTICS could potentially provide reasonably satisfactory clusters, their parameters do not afford clear substantive interpretations like HDBSCAN’s parameters.
FIGURE C2
LONG FINGER CONTROL AUTOMATION PRACTICES
IN THE REALIZED POSSIBILITY SPACE
FIGURE C3
HDBSCAN 127 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS ARE SHOWN)

FIGURE C4
KMEANS 127 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS ARE SHOWN)
FIGURE C5

WARDS 127 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS ARE SHOWN)

FIGURE C6

SINGLE LINKAGE 127 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS ARE SHOWN)
FIGURE C7
COMPLETE LINKAGE 127 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS ARE SHOWN)

FIGURE C8
AVERAGE LINKAGE 127 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS SHOWN)
FIGURE C9

AFFINITY PROPAGATION 941 CLUSTERS (ONLY LONG FINGER CONTROL CLUSTERS ARE SHOWN)
WEB APPENDIX D

VISUALIZING THE REALIZED POSSIBILITY SPACE

D1. Riemannian Manifolds

*What is a Manifold?* Simply put, a Riemannian metric means that the way distance is defined can differ throughout the space. A Riemannian manifold is then a smooth surface with complex nonlinear structure that is embedded in the higher-dimensional space. Despite this complexity, if you take a small enough neighborhood around a point on the manifold, you can map it into Euclidean space. Then, these small local regions can be patched together into a map. As a simple illustration, consider the two-dimensional curved manifold of the Earth’s surface. We can represent the Earth’s curved surface as a flat two-dimensional map, recognizing that while local distances on the flat map will be accurate, global distances will be distorted.

*Why Manifolds are Important for Assemblage Theory.* Imagine a two-dimensional surface that curves and twists through an even higher dimensional space. For example, consider a two-dimensional surface that contains our 20,675 realized applets. This surface lies in the 50-dimensional space of our applet embeddings. But we can focus our attention on the two-dimensional manifold, rather than the 50-dimensional space, because all of the information we need to understand the possibility space is contained on the manifold, and so the higher-dimensional space can be ignored. In this way, a manifold is said to be a surface “that is a space in itself” (DeLanda 2002, p. 4). Since the manifold “is intrinsically defined, without external reference or recourse to a uniform space in which it would be submerged” (Deleuze 2004, p 231), it has a direct assemblage theory interpretation as a Deleuzian multiplicity (Deleuze 2004; Deleuze and Guattari 1987; DeLanda 2002). Further, the manifold of the possibility space of automation assemblages, is immanent and not transcendent. This means the external space that contains the manifold is not required to understand the points of attraction that guide the formation of automation assemblages. All we need to understand the automation assemblages is the manifold. This is why DeLanda (2002, 2016) interprets the possibility space in assemblage theory as an N-dimensional Riemannian manifold, with points of attraction as topological features.

D2. Comparison of Six Methods for Dimensionality Reduction

Each automation applet is represented by a vector of 50 features (25 trigger and 25 action features), derived from embeddings of the 1383 words contained in the text describing the applet’s trigger and action. Cosine distances for all pairs of applets were obtained as one minus cosine similarity obtained from expression 2 in the paper. The upper triangle of the matrix of cosine distances among the 20,675 realized automation applets contains nearly 214 million values. This is appreciably larger than distance matrices that have been considered in applications of scaling methods to market structure problems in marketing (e.g., Ringel and Skiera (2016) scale about a thousand products). The complexity of a distance matrix with nearly 214 million pairwise distances presents significant challenges for conventional data reduction and representation techniques.
In practice, UMAP has been shown to outperform other techniques for manifold learning in terms of speed and scalability, as well as the ability to both preserve local structure while at the same time reveal aspects of global structure (McInnes 2020; McInnes, Healy and Melville 2018; Sainburg, Thielk and Gentner 2019). We compared six different machine learning methods for dimensionality reduction that are commonly used in practice: PCA, MDS, ISOMAP, Spectral Embedding, t-SNE and UMAP. All models were fit in Python using PCA and manifold learning algorithms from scikit-learn 0.19.1, as well as umap-learn 0.3.10. Figure D1 plots the 20,675 automation applets for each of the six methods. The applets in Figure D1 are colored by the 127 HDBSCAN clusters. As noted in the paper, the human eye cannot distinguish all these individual colors in the plot, but the colors still provide some guidance.

In all the methods except for MDS, similar colors generally go together in the plot, indicating that applets in each of the 127 clusters are close together in the plot. However, the six methods vary considerably in their ability to clearly represent the structure of the automation applets. MDS performs the worst, failing to recover any clear structure. PCA, Spectral Embedding, and especially Isomap all reveal aspects of global structure, but do not help us understand local structure. Of the six methods, only t-SNE and UMAP provide a clear understanding of local structure. This is not surprising, given that t-SNE, a manifold learning algorithm for nonlinear dimensionality reduction (van der Maaten and Hinton 2008), preserves local similarity structure, as does UMAP. However, in addition to revealing local structure, UMAP also reveals aspects of global structure (McInnes, Healy and Melville 2018). This can be seen in the way that many small groupings in the UMAP plot are seen to group into broader, higher-level groups. With t-SNE, however, the many small groupings are fairly uniformly distributed throughout the plot, without a clear representation of higher-level global structure. For this reason, we use UMAP to perform dimensionality reduction for visualizing the possibility space of automation assemblages.

We note that recent research has shown that the superiority of UMAP over t-SNE is because UMAP uses Laplacian eigenmaps to initialize the embeddings, while t-SNE uses a random initialization (Kobak and Linderman 2019). While t-SNE can achieve similar performance as UMAP by custom programming a Laplacian eigenmap initialization, at this point in time t-SNE packages in R and Python do not support this option. However, even if t-SNE packages offered this initialization option, UMAP’s hyperparameters (especially min_cluster_size) have a clearer and more direct interpretation from an assemblage theory perspective.
FIGURE D1

COMPARISON OF SIX DIMENSIONALITY REDUCTION METHODS
D3. UMAP Parameters

We tested a range of values of the two UMAP parameters, n_neighbors (10, 25, 50, 100) and min_dist (.05, .15, .25). Figure D2 plots the 20,675 applets on two UMAP dimensions, for each combination of these two parameters. The n_neighbors parameter controls how UMAP balances local (smaller values of n_neighbors force UMAP to concentrate on local structure) versus global structure (larger values lose fine detail but provide a broader view) when estimating the structure of the manifold. For a given value of n_neighbors, the min-dist parameter specifies the minimum distance points can be apart in the embedding, and controls whether points are packed together tightly or loosely.

For each row of figure D2, the columns show a progression from relatively tightly to more loosely packed points. For each column, the rows show a progression from a focus on purely local structure at the top, to an increasing focus on global structure on the bottom. Global structure only starts to emerge with values of n_neighbors equal to 25 or more. At the same time, min_dist=.25 introduces a degree of looseness in the solution that obscures the global structure. Since our data have both local and global structure, we sought a compromise that represented both, and thus selected the solution based on n_neighbors=50 and min_dist=.15. This subjective decision is driven by our need for sufficient detail combined with our desire to see aspects of the big picture.

Fortunately, this choice of UMAP hyperparameters is not critical for the purpose of displaying HDBSCAN clusters. This is because the HDBSCAN clustering was independently performed on the 50-dimensional embeddings, not on the two dimensions from the UMAP solution. Other combinations of UMAP, such as n_neighbors=50 and min_dist=.05, would produce essentially similar interpretations since the topological structure of the clusters is invariant under continuous deformations of the space.

---

1 For further discussion of UMAP parameters see: https://umap-learn.readthedocs.io/en/latest/parameters.html.
FIGURE D2

PLOTS FOR VALUES OF UMAP HYPERPARAMETERS N_NEIGHBORS AND MINDIST

mindist  For a given value of n_neighbors, the mindist parameter gives the minimum distance points can be apart in the embedding, and controls if points are packed together tightly or loosely

0.05  0.15  0.25
10  25  50  100
WEB APPENDIX E

THEMATIC ANALYSIS DETAILS

E1. Three Data Levels

Our thematic analysis includes only territorialized realized applets and the clusters to which these applets were assigned. Deterritorialized realized applets (i.e., noise points) were ignored. The derived and observed data we have available for thematic analysis are at three different levels, as shown in table E1. At the highest level are clusters, of which 127 were identified. At the middle level are the unique realized applets, 15,417 of which were assigned to a cluster (i.e., territorialized). At the lowest level are copies of the unique realized applets. A total of 278,716 copies of the 15,417 unique applets were published by different IFTTT users. Thus, copies of applets are nested within unique applets, and unique applets are nested within clusters.

Different data types of data are available at each level. At the highest level (cluster level), the average cosine similarity of all pairs of applets in that cluster.

At the middle level (unique applet level) we have both observed and derived data. The observed data are the six text fields of ordered structured text describing each of the 15,417 territorialized unique applets. The derived data are as follows: 1) the HDBSCAN probability that the applet is a member of the cluster to which it was assigned; 2) the HDBSCAN outlier score; 3) an interactive version of the two-dimensional visualization from figure 3 of the 15,417 unique applets from UMAP assigned to the 127 clusters with hoverable annotations for each applet; and 4) the two-dimensional visualization of the nested cluster boundaries as depicted in figure 4. See Web Appendix C1 for further details of how the nested boundaries were determined.

At the lowest level (copies of unique applets), we have observed data that we used to interpret the expressive roles, but which were not used in the computational analysis. For each of the 15,417 unique applets, there are a varying number of copies, with a total of 278,716 copies. Observed data is available for 36.3% of these copies (101,174 copies). This observed data is a text description that consumers had the option of providing that described the applet’s function in their own words. This optional text, consisting of anywhere from a few words to a few sentences, gave consumers the opportunity to express why they created the applet. The remaining 63.7% of applet copies did not contain usable descriptive text. Across clusters, this amounts to 30% with no optional text, 8.4% containing gibberish characters, 3.3% containing non-English characters, and 22% simply duplicating some or all of the applet text fields themselves, providing no additional information beyond those six required text fields. On average, each of the 127 clusters had 2200 applet copies, with approximately 798 applet copies per cluster containing optional text. However, the amount of optional data available for each cluster varied widely. For example, cluster 35 contains 219 unique applets and 342 applet copies, with 92.4% of applet copies (316 applet copies) containing usable descriptive text. We also constructed word clouds for each cluster from the available optional text descriptions using the Python WordCloud V1.5.0 library. We plotted the word clouds using the PyPlot routine from matplotlib.
E2. Expressive Roles

The inductive process we employed to interpret the expressive roles of the automation assemblages is diagrammed in figure E1. It graphically summarizes the two-stage highly iterative reading and visualization process we employed. Our goal was to understand the emergent meaning in automation practices by considering the interconnecting relationship between the parts and the whole. The last column in table E1 connects the three levels of data (the parts) to each stage of analysis.

**Stage One.** We began by familiarizing ourselves with the diversity in the six fields of text describing the applets in each cluster and the optional descriptive text. Then, we constructed a “super-applet” text description from the trigger and action components of all applets in that cluster to make sure we understood the nuance in each automation practice’s materials and competences. This super-applet description was constructed as an if:then statement of the format “if [list of all trigger objects for all applets in that cluster] (list of highest frequency trigger events in descending order for all trigger objects for all applets in that cluster), then [list of all action objects for all applets in that cluster] (list of highest frequency action events in descending order for all action objects for all applets in that cluster).” For example, the super-applet description for cluster 69, consisting of 124 applets, is “if new switch, light, env, weather, security, hub event (e.g., switched on/off, nest set to away, say specific phrase, alarm goes off, motion detected, sunset, temp drops below threshold), then control lights, switch security, hub (e.g., change color, start recording, voice announcement, lock, toggle on/off, turn off plug). The complete list of super-applets and material roles labels are listed in an Excel spreadsheet, available online at OSF.

We independently coded the 127 practices through an iterative reading and visualization process, generating codes we believed captured the underlying themes in each cluster. The data we used primarily included the material roles of each cluster, the 101,174 optional text descriptions available from the applet copies, and the nested boundaries in figure 4. We also secondarily used the applet HDBSCAN cluster probability of membership and outlier metrics, and the average cosine similarity for each cluster.

As we read the descriptive text to identify themes, we used several metrics to help orient our attention to specific cluster applet descriptive text. Applets with higher cluster membership probabilities and lower cluster outlier scores were used to focus attention on descriptive text from applets that were deemed most representative of that cluster. Clusters with higher average cosine similarities among all applet pairs were considered more homogeneous with respect to descriptive text, while clusters with lower average cosine similarities were considered more diverse and potentially requiring additional effort to interpret in terms of descriptive text. In general, because applets within a cluster are more similar to each other than applets across clusters, we assumed that applets in a cluster reflected a common theme.

We applied inductive reasoning to methodically infer the themes based on our independent judgment of the underlying meanings expressed by optional descriptions provided for the applet copies in each cluster. Our domain expertise provided important context. We looked for patterns and differences across the different text descriptions in a cluster, and across clusters within and across the bounded regions of the realized possibility space using the
interactive visualizations to locate individual applets and clusters. From an interpretive standpoint, we considered each optional description as a mini-narrative conveying a plot. As we evaluated each description in a cluster, we looked for similar expressions across these narratives. Even if the actual words consumers used were not exactly the same, we took these similar expressions to imply a common theme. These expressions helped us understand the metaphors in the narratives.

As a brief example of our process, consider cluster 35, containing 219 unique applets. The automation assemblages in this cluster automate a smart device event based on a mobile device event. For example, the IFTTT rule “Mobile Devices and Accessories > Android Device > Disconnects from a specific WiFi network || Lighting > Philips Hue > Turn off lights,” turns off lights in the home when the phone is no longer on the home network. The short-hand notation for the material role is “Phone/Text: Smart Devices.” The super-applet for this cluster describes its material role:

if new notification or mobile device event (e.g., text received/sent matches search, tap button, device disconnects), then control lights, env, switches, hub, security (turn on/off, set temp, blink lights, switch off/on)

From figure 4, we can see that cluster 35 is territorialized with cluster 27 (phone/text: smart devices) and nested within Group F (smart home automation assemblages), which itself contains other nested groups of clusters. Material roles of other territorialized automation assemblages in F involve DIY/QS: smart devices, and smart devices: smart devices. This understanding of the cluster’s material role provides important insight into its functional meaning and creates the foundation on which we can interpret the data to understand the symbolic meaning.

Over 92% of the 343 applet copies in this practice had optional text descriptions. We focused our attention on the nearly two thirds of applet copies in the practice that had high HDBSCAN membership probabilities and low outlier scores. The average cosine similarity for this cluster is .775, suggesting some diversity among applets. Reviewing the super-applet suggests that this diversity likely arises because of the different types of triggers (text, button taps) and actions (turn on, blink) that can be enacted by connecting a mobile phone to a smart object. We read the optional text descriptions with an eye toward understanding the story they were trying to tell, giving more interpretive weight to the descriptive text from applets with higher probabilities of membership and lower outlier scores. Here are several illustrative optional text descriptions from cluster 35 that reflect many of the applets in this cluster:

“A warm welcome. A warm and skin flattering color for your cosy home and bedroom activities ;-)”

“can't have a hue mood without a little ‘‘love shack’’”

“unplugging my phone every morning turns up temp of house so it's nice and warm when i wake up”
These mini-narratives, and dozens of others like them in cluster 35, provided insight into why applets were created to connect smart phone triggers to smart device actions. In our readings of the text, we interpreted these narratives as revealing that people created these applets to create a particular ambient mood in their homes. We believe that the story being told is one of consumers automating the creation of unique ambient environments that convey pleasure or comfort. As we continued to come across similar expressions in the narratives, we began to recognize the metaphors that helped us understand the meaning expressed in that cluster. Even if the words consumers used were not identical, we took them to imply a common theme. As we reflected on this insight in the context of our domain expertise, we developed an interpretation of the applets in cluster 35 representing the theme of ambient awareness. Ambient awareness involves a peripheral awareness of the environment enabled through a device (Levordashka and Utz 2016).

Our interpretations of the optional text descriptions conceptually moved between evaluating the short snippets of text as consumer narratives of personal meaning to statements embedded in the broader context of consumer IoT. When we located themes that did not appear to fit the data well, we independently came up with new themes that fit better. Following this phase of independent theme coding, we met to resolve through discussion any further discrepancies, and continued iterating until consensus was achieved. We then connected the themes to the extant literature and defined them. This stage resulted in 14 different themes underlying the meaning of the expressive roles.

Stage Two. Once we interpreted the data for the symbolic meanings underlying the 14 themes, we were interested in integrating these themes into broader categories so as to place the practices into a larger sociomaterially relevant framework. This interpretive effort comprised stage two. We evaluated each practice theme in the context of our domain expertise in online consumer experience and the consumer IoT. We also relied on figure 4, which we labeled with each automation theme. As we iterated among the themes during interpretation, the successive visualizations revealed the spatial relationships among different themes. This led us to interrogate what underlying groupings of higher-order meaning might capture the similarities and differences among themes in regions of the realized possibility space. As part of this process, we also iterated back to the first stage, where we continued to evaluate whether the 14 interpreted themes best fit the data and adjusted as necessary. Once no additional readings or visualizations of the data suggested new or modified themes or categories (i.e., saturation was reached), we created a visualization (figure 5) with colored overlays to graphically represent the results.

Table E2 summarizes the results of our analysis in tabular form. The table includes a brief definition of each higher-order practice category, definitions for each of the 14 themes, and illustrative optional descriptive text for each theme.

Last, we note that we experimented with word clouds and topic models to see if they would support our understanding of the expressive roles. We used the Python WordCloud V1.5.0 library to create word clouds for each of the 127 clusters from the available optional text descriptions of the applet copies in that cluster. Examples of word clouds for clusters 35 and 14 appear in figures E2a and E2b, respectively. The word clouds were plotted with the PyPlot routine from matplotlib. The word cloud application indicates which words appear most
frequently. As a result, the word clouds necessarily emphasized the material, as opposed to expressive, roles of each cluster, since consumers tend to use identical words to describe applet functionality (e.g., “activate” or “turn on”), but more varied descriptions to describe the meaning underlying an applet’s material role (e.g., “nice and warm when I wake up” or “love shack”). Consequently, we did not find that the word clouds added to our understanding of the expressive roles.

We also experimented with topic models on the optional descriptive text to further understand each individual cluster. Thus, we additionally fit several topic models to the optional text descriptions for several clusters. Since the optional text was natural language text, rather than structured text, we compared topic models based on unigrams with models based on unigrams and bigrams. The Python scikit-learn 0.24.2 LatentDirichletAllocation package was used to fit these topic models. The results are summarized in figure E3 for clusters 35 and 14 for illustrative purposes. Inspection of the results shows the topics, like the word clouds, emphasize the material roles of the clusters. We did not find that the topics added to our understanding of the expressive roles.

We considered why the word clouds and topic models of the optional descriptive text were not especially helpful in deepening our understanding of expressive roles in our thematic analysis. In contrast, human reading and interpretation of the optional descriptive text provided clear insights about expressive roles, just as our computational analysis provided clear insight into the structure of automation assemblages based on their material roles. Are material roles well suited to analysis by computational methods, but expressive roles require human interpretation? Perhaps to some extent. Yet, we believe that computational methods that are designed to work together with the human researcher to more appropriately simplify a large text corpus may be a more fruitful path to explore in future research. For example, methods for extractive (Gupta and Lehal 2010) and abstractive (Moratanch and Chitrakala 2016) text summarization could be used to distill a massive text corpus to a more manageable size while fundamentally preserving key meaning, after which the human researcher could step in with an interpretive analysis.
### TABLE E1

**LEVELS OF DATA USED IN THE THEMATIC ANALYSIS**

<table>
<thead>
<tr>
<th>Data Level</th>
<th>Description</th>
<th>Data Describing Realized Applets</th>
<th>Data Used for Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>High-Level</td>
<td>Cluster Data</td>
<td>Average cosine similarity among all pairs (derived)</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>127 clusters learned from HDBSCAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mid-Level</td>
<td>Unique Applet Data</td>
<td>6 fields of ordered structured text (observed)</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>15,417 realized applets assigned to 127 clusters by HDBSCAN</td>
<td>HDBSCAN probability that applet is a member of its cluster (derived)</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HDSCAN cluster outlier score (derived)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Two-dimensional visualization of the 15,417 realized applets as nested boundaries in figure 4 (derived)</td>
<td>x</td>
</tr>
<tr>
<td>Low-Level</td>
<td>Applet Copies Data</td>
<td>Optional descriptive text describing applets for 101,174 (36.3%) of the copies (observed)</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>278,716 applets representing copies of the 15,417 realized applets assigned to a cluster</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Higher-Order Practice Assemblage Category and Definition (# practices in category)</td>
<td>Theme Title and Definition (# practices in theme)</td>
<td>Illustrative Optional Descriptive Text (Practice #, trigger: action material role)</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Social Expression (24) Automate communication in social media</td>
<td>Self-presentation (17) Project a desirable public image</td>
<td>“Because I prefer content going to my LinkedIn to be technology-focused, I only republish through IFTTT when my Technology category is used” (Practice 120, “Blog: Twitter/LinkedIn”)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Self-disclosure (7) Communicate personal information regardless of the impression</td>
<td>“put moods, impressions and more in your dropbox and let it post on your tumblr” (Practice 74, “Archive: Social”)</td>
<td></td>
</tr>
<tr>
<td>Social Connectedness (19) Automate communication and information-seeking forms of social connection</td>
<td>Sharing (3) Share content as result of an event to stay connected with online friends in real-time</td>
<td>“When you get to Antarctica, tweet that it’s cold. When you get to Antarctica, your fingers will be too cold to Tweet, but you will want to let people know you are there” (Practice 17, “Location:Notify/Post”)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Collaborative sharing (3) Share workplace content on team platforms as a result of an event.</td>
<td>“Help your coworkers know when they shouldn’t disturb you by posting a notice in a Slack channel when you have started a FocusTime session. Be sure to fill in ‘YOUR-NAME’ in the message so they can identify you. You may want to also use a second recipe to post an ‘all-clear’ message when your FocusTime session is over” (Practice 58, “Task: Collaborate”)</td>
<td></td>
</tr>
</tbody>
</table>
| Social surveillance (13) | “Get the most popular health articles from the NY Times sent to your inbox. Receive new popular Health articles from NYTimes automatically, and stay updated with the latest news and health tips” (Practice 24: “Online Content/Biz: Email”)
“Get notified immediately if you get an email from your wife/GF” (Practice 54 (“Email:Text”))
“Is there someone on Instagram you want to stalk...here is the recipe to be notified by SMS every time they post a photo” (Practice 111: “News/Online Content: Text”). |
| Digital content scanning and covert information-gathering |

| Extended Mind (38) | Transactive memory partner (24) | “YouTube to Journal. Create a notebook in Evernote called "Journal" (or change the task to point towards your chosen notebook.) Use with other [insert social media outlet] to Evernote journal tasks to keep a chronological journal of your online life” (Practice 98 “Photo/Video/Save Note”) |
| Automate extending human (or object) cognition to digital entities |
| Replace, extend, and augment minds, with the IoT serving as the “group” |

| Personal quantification (10) | “I find it really useful to have all your fitness data in one place that you use a lot. I love having all of my data on my calendar so I can reference it daily. This recipe will publish all of your WiThings Scale measurements to your Google Calendar” (Practice 4, “Online Content: Schedule”). |
| Self-archiving/Quantification of the self |

<p>| Object quantification (4) | “Add a row to spreadsheet everyday I lose the connection with my fridge. Imagine the different reasons why you can lose the connection with your fridge: a blackout, or maybe an issue on your home network...or maybe something is happening in your house” (Practice 61, “Smart Device: Cloud”). |
| Object behavior quantification |</p>
<table>
<thead>
<tr>
<th>Relational AI (46)</th>
<th>Ambient awareness (13)</th>
<th>Ambient control (4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automate repeated interactions between consumers and smart objects</td>
<td>Peripheral awareness of actions through smart objects</td>
<td>Location-based peripheral control of smart object interactions</td>
</tr>
<tr>
<td>“Sorry I missed Hue. I'll call you right back. Phone in the other room and missed a call? Have your Hue lights blink when you miss a call.” (Practice 27, “Phone/Text: Smart Devices”).</td>
<td>“Never worry about forgetting to turn your AC off when you leave home again, ifttt will do it for you when you leave” (Practice 15, “Location: Smart Devices”).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Long-finger control (12)</th>
<th>Long-finger control, routinized (4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master-servant remote control</td>
<td>Remote control on a set schedule</td>
</tr>
<tr>
<td>“This recipe is used to control a fan in my attached greenhouse. When the target temperature measured by Netatmo is reached a fan plugged into a Wemo switch is turned on circulating warm moist air through a network of pipes buried in the earth below. The air returns cool and dry maintaining ideal temperature and humidity for the plants while storing the heat to extend the growing season and reduce heating costs” (Practice 75, “Smart Devices: Smart Devices”).</td>
<td>“Disarm iSmartAlarm every weekday morning. Never again forget to disarm your alarm before you walk out the door on your way to work and disturb the entire household by setting off the alarm” (Practice 34, “Schedule: Smart Devices”).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Partners, alerts (10)</th>
<th>Partners, self-control (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart object as consumer proxy for alerts</td>
<td>Smart object as consumer support</td>
</tr>
<tr>
<td>“Last winter my pipes kept freezing because I wouldn't leave the water trickling when it was cold out. This recipe will send a reminder when tomorrow's forecast is below 30F to help avoid situations like this” (Practice 25, “Online Content: Save Note”).</td>
<td>“A recipe that gives you an extra prodding when you haven't met your FitBit goal for the day, and sends you a text. Saves me from being a lazy bum on the couch some nights!” (Practice 49, “QS: Text”).</td>
</tr>
</tbody>
</table>
FIGURE E1

DEPICTING THE THEMATIC ANALYSIS PROCESS USED TO INTERPRET THE AUTOMATION ASSEMBLAGES AS PRACTICES

What meanings underlie material connections? (Initial context for the "whole" is top-down)

Higher-order Practice Categories
Bottom-up understanding of automation practices based on hermeneutic reading of why triggers and actions are connected

"Parts" for Understanding Meanings of Expressive Roles
- Material roles of automation assemblages based on 6 fields of structured text
- Optional descriptive text
- Nested boundaries (fig 5)
- HDBSCAN metrics and avg cos sim

Iterative reading and visualizing

Automation Practice Themes
Expressive roles of automation assemblages interpreted as symbolic meanings of practices

"Parts" for Understanding Higher-Order Meanings
- Material roles and nested boundaries (fig 5)
- Optional descriptive text
- Automation practice themes (fig 6)
FIGURE E2A

WORD CLOUD FOR CLUSTER 35

FIGURE E2B

WORD CLOUD FOR CLUSTER 14
FIGURE E3

TOPIC MODEL ANALYSIS

*Topic Models - Cluster 35 (unigrams only)*

CountVectorizer(analyzer = 'word', max_df = 0.95, min_df = 2, ngram_range =(1,1))

Solution based on both minimum perplexity and decreasing rate of perplexity change.

Topic 0: reminder in light color is list reminders add harmony charge
Topic 1: your lights you when from wifi turn off home phone
Topic 2: to home wifi when my you your if on connects
Topic 3: onhub device with to control on turn wemo when switch
Topic 4: android your wear on turn to with button watch off
Topic Models - Cluster 35 (unigrams and bigrams)

CountVectorizer(analyzer = 'word', max_df = 0.95, min_df = 2, ngram_range =(1,2))

Solution based on decreasing rate of perplexity change.
Topic 0: to onhub when you with turn device if off then
Topic 1: your android on wear turn lights android wear to off light
Topic 2: on smarthings my to lights with door your hue reminder
Topic 3: wifi home your to when you from home wifi network if

Solution based on minimum perplexity.
Topic 0: to reminder you is in it when light off turn
Topic 1: lights on turn your you turn on when off your lights phone
Topic 2: it your to hue my on network reminder with lights
Topic 3: your wifi home to when you home wifi network phone your home
Topic 4: color light to your light color lights with change warm android
Topic 5: wear android android wear to your set on temperature from your android
Topic 6: wifi when home from disconnects home wifi disconnects from specific on from home
Topic 7: onhub with device control with onhub when device on switch wemo when
Topic 8: to your android my use watch arm disarm your android off
Topic 9: to my if wifi when this you for connects wifi then
Topic 10: garage door garage door on with turn wemo task in list
Topic 11: then if turn switch then turn off you network then any ac
Topic 12: recipe this recipe my this pod wattio pod wattio when to need
**Topic Models - Cluster 14 (unigrams only)**

CountVectorizer(analyzer = 'word', max_df = 0.95, min_df = 2, ngram_range =(1,1))

Solution based on decreasing rate of perplexity change.
Topic 0: to home you set when your area thermostat nest if
Topic 1: you your when turn area on an leave home to

Solution based on minimum perplexity.
Topic 0: home to when your thermostat set you nest get for
Topic 1: to home this for will use your thermostat work you
Topic 2: when you your turn area to location leave an enter
Topic 3: you your house in when it caleos up town will
Topic 4: turn when on you your home off leave air heating
Topic 5: home to you your when if schedule thermostart resume leave
Topic 6: you area an if set then enter to exit for
Topic 7: to set away your when home thermostat you ecobee leave
**Topic Models - Cluster 14 (unigrams and bigrams)**

`CountVectorizer(analyzer = 'word', max_df = 0.95, min_df = 2, ngram_range =(1,2))`

Solution based on decreasing rate of perplexity change.
Topic 0: your you heating turn in when away leave off when you
Topic 1: home your to when nest turn on set you ac
Topic 2: when you when you your home to area leave set you leave
Topic 3: to home your set when thermostat away you ecobee location
Topic 4: you area an an area if if you then set area then exit

Solution based on minimum perplexity.
Topic 0: your you in away your home caleos your caleos when you when town
Topic 1: home nest to your when turn set ac location close
Topic 2: your you when to when you home leave you leave air on
Topic 3: home your when to near set on your home near home arrive
Topic 4: you if if you area then an an area enter exit you enter
Topic 5: you set area if an area an to then set area then then
Topic 6: to home thermostat resume when schedule you away program thermostat to
Topic 7: when you home on location when you your turn to for
Topic 8: turn off an area you area an when when you your smart
Topic 9: to you home your leave when away you leave thermostat when you
Topic 10: when set to area home ecobee your when you you away
Topic 11: nest summer enter enter summer nest enter exit do arrive at run warm house
WEB APPENDIX F

PRACTICE GROWTH POTENTIAL

As described in the paper, using the HDBSCAN approximate_predict() function, a total of 212,527 of the 895,575 unrealized applets, or 23.7%, are predicted to belong to one of the 127 existing territorialized automation practices. The remaining 683,048 unrealized applets were not predicted to belong to an existing practice. The number of unrealized applets either assigned or not assigned to a practice are reported in cells C and D in the second row of table F1. For comparison, cells A and B in the first row of table F1 report the number of realized applets that were assigned, or not assigned, to a practice.

\[
\begin{array}{|c|c|c|}
\hline
\text{Territorialized} & \text{Deteritorialized} & \text{Total} \\
\text{Classified into one of 127} & \text{Not classified into one of} & \\
\text{automation practices} & \text{127 existing automation} & \\
& \text{practices.} & \\
\hline
\text{Realized} & \text{(A)} & \text{(B)} & \text{20,675} \\
\text{Automation} & 15,417 & 5,258 & \\
\text{Applets} & \text{Realized applets assigned} & \text{Realized applets not} & \\
& \text{to a practice by} & \text{assigned to a practice by} & \\
& \text{HDBSCAN} & \text{HDBSCAN} & \\
\text{Unrealized} & \text{(C)} & \text{(D)} & \text{895,575} \\
\text{Automation} & 212,527 & 683,048 & \\
\text{Applets} & \text{Unrealized applets assigned} & \text{Unrealized applets not} & \\
& \text{to a practice by} & \text{assigned to a practice by} & \\
& \text{the HDBSCAN} & \text{the HDBSCAN} & \\
& \text{approximate_predict() function} & \text{approximate_predict() function} & \\
\hline
\end{array}
\]
If every one of the 212,527 unrealized applets that are predicted to be in one of the 127 existing automation practices were realized, the number of applets that are part of an existing practice would grow 14.79 times, from 15,417 to 227,944. In the same way that we estimated the overall potential of automation practices across all 127 practices, individual cluster size growth potential can be calculated for each of the 127 automation practices. That is, growth potential for automation practice \( i \) = (number of realized applets in cluster \( i \) + number of unrealized applets predicted to be in cluster \( i \)) / (number of realized applets in cluster \( i \)). Across the 127 automation practices, these cluster size growth potentials range from 1.4 to 158.

The paper reports that means of log growth rates were significantly different across the four categories of higher-order practices (see also figure 8 in the paper). Tukey HSD post-hoc tests of the four higher order practices show that the mean of the log growth potential for Relational AI practices is significantly different from each of the other three higher-order practices (\( p < .001 \) for each of the three tests); all other pairwise comparisons were not significant (\( p > .05 \) for all comparisons). The automation practices with the highest and lowest growth potential, within each of the four process groups, along with the average cosine similarity of all pairs of applets in the practice, are shown in table F2.
## TABLE F2

PRACTICES WITH HIGHEST AND LOWEST GROWTH POTENTIAL IN EACH HIGHER-ORDER PRACTICE GROUP

<table>
<thead>
<tr>
<th>Practice Group</th>
<th>Highest Growth Potential Practice</th>
<th>Lowest Growth Potential Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational AI</td>
<td>Practice 13 (Ambient Control)</td>
<td>Practice 25 (Partner - Alerts)</td>
</tr>
<tr>
<td></td>
<td>Growth Potential = 157.75</td>
<td>Growth Potential = 2.78</td>
</tr>
<tr>
<td></td>
<td>Average Cosine Similarity = .773</td>
<td>Average Cosine Similarity = .879</td>
</tr>
<tr>
<td></td>
<td>if connected car event (e.g., ignition turned on/off in area), then control lights, hub, env, security, switch, (e.g., set temp, blink lights, arm security panel)</td>
<td>if new env, weather, security, switches, light event (e.g., temp rises above threshold, motion detected), then email me</td>
</tr>
<tr>
<td>Extended Mind</td>
<td>Practice 11 (Personal Quantification)</td>
<td>Practice 100 (Transact. Memory Partner)</td>
</tr>
<tr>
<td></td>
<td>Growth Potential = 43.40</td>
<td>Growth Potential= 2.80</td>
</tr>
<tr>
<td></td>
<td>Average Cosine Similarity = .803</td>
<td>Average Cosine Similarity = .931</td>
</tr>
<tr>
<td></td>
<td>if fitness/wearable event (e.g., daily step goal achieved, new sleep logged), then archive in personal journal (e.g., create journal entry, add a note)</td>
<td>if new bookmark, news/info content (e.g., public bookmark tagged, favorite item, article saved for later, article tagged), then organize/archive (e.g., create note, create link note, create page from link)</td>
</tr>
<tr>
<td>Social Connectedness</td>
<td>Practice 84 (Sharing)</td>
<td>Practice 29 (Social Surveillance)</td>
</tr>
<tr>
<td></td>
<td>Growth = 21.53</td>
<td>Growth = 1.40</td>
</tr>
<tr>
<td></td>
<td>Average Cosine Similarity = .714</td>
<td>Average Cosine Similarity = .948</td>
</tr>
<tr>
<td></td>
<td>if weather, finance, env, hub, security, appliance event (e.g., today's/tomorrow's weather report, current condition changes to, price at close, new sale), then create new social media, collaboration, blog content (e.g., post tweet, status msg, post to channel, link post)</td>
<td>if new biz/dev, shopping, finance event, news/info, social media, photo/vid, bookmark, blog content (e.g., visit to specific doc, trending story, sale, public bookmark, check-in, photo), then email (email me, add to weekly/daily digest)</td>
</tr>
<tr>
<td>Social Identity</td>
<td>Practice 82 (Strategic Self-Presentation)</td>
<td>Practice 123 (Strategic Self-Presentation)</td>
</tr>
<tr>
<td></td>
<td>Growth = 21.36</td>
<td>Growth = 1.72</td>
</tr>
<tr>
<td></td>
<td>Average Cosine Similarity = .859</td>
<td>Average Cosine Similarity = .940</td>
</tr>
<tr>
<td></td>
<td>if new audio, social media content (e.g., liked video, saved track), then create audio content (e.g., upload public track, add track to playlist)</td>
<td>if new social media content (e.g., photo by you, pin on your board, link by you), then upload a photo from URL to Facebook Pages/Facebook</td>
</tr>
</tbody>
</table>
WEB APPENDIX G

VISUALIZING THE FULL POSSIBILITY SPACE

To visualize the full possibility space of realized and unrealized applets, we used UMAP to learn a two-dimensional representation of the 916,250-by-916,250 matrix of cosine similarities among both realized and unrealized applets. UMAP readily scales to enable analysis of such very large matrices. We used the cosine distance metric with hyperparameters n_neighbors=300 and min_dist=.10. To incorporate our prior knowledge of the 127 learned automation practices into the full possibility space, we used a semi-supervised UMAP (McInnes, Healy and Melville, 2018). The 127 automation practice clusters to which the 15,417 realized applets that were assigned to an automation practice served as partial labels. These partial labels were used as categorical targets by semi-supervised UMAP (McInnes 2020), maximizing the quality of the representation of the existing automation practices in the context of the unrealized applets.

The full possibility space plot of all 916,250 possible applets in figure 9a of the paper provides a different representation of the 20,675 realized applets (i.e., the red and orange points in figure 9a) than the realized possibility plot of 20,675 applets in figure 3 of the paper. While both plots can be used to represent the points of attraction underlying the realized applets, figure 3 provides a clearer picture of the points of attraction because it emphasizes what has been created rather than what could be created. However, figure 9a is useful for conceptually understanding the portions of the possibility space that have not yet been realized.

As intuition for understanding these ideas, think of a manifold as a surface, and consider the example of the surface of the Earth and what happens when you put the round Earth on a flat map. Figure G1a shows a Mercator projection that provides a good representation of the Earth’s land masses. Locally, distances are Euclidean, but globally distances and sizes are distorted. Analogously, UMAP provides a two-dimensional projection of the manifold of the realized automation applets (the “land masses”), although the Riemannian manifold of the applets lies in a 50-dimensional space of word embeddings, and its metric can vary through this 50 dimensional space. The question may arise as to why we are using the 20,675 realized applets to represent the possibility space, rather than all 916,250 possible applets. Since the realized applets are guided by points of attraction, these points of attraction are what we seek to visualize. Following our analogy, if the realized applets are the land masses, then the unrealized applets are a vast ocean. Figure G1b shows a Spilhaus projection of the Earth which provides a good representation of the oceans, but at the expense of the land masses. Since points of attraction are revealed through the land masses and not the ocean, it is more reasonable to use the approximately 2% of realized automation applets to represent the possibility space, than to allow the nearly 98% of unrealized applets to dominate the structure.
FIGURE G1

PROJECTIONS OF THE MANIFOLD OF THE EARTH

a. Mercator Projection Represents Land Masses (analogy: realized applets)

b. Spilhaus Projection Represents Oceans (analogy: unrealized applets)
WEB APPENDIX H

TOPIC MODELS AS AN ALTERNATIVE APPROACH TO WORD EMBEDDINGS

In Web Appendix H, we consider whether topic models based on Latent Dirichlet Allocation (LDA) (e.g., Tirunillai and Tellis 2014) might provide an alternative to our approach based on applet embeddings using word2vec. We identify three ways in which we believe it is appropriate to apply LDA to our text corpus of 20,675 realized applets to obtain clusters. First, we performed a standard LDA of the applet text, and assigned each applet to a cluster based upon the dominant topic with the highest probability. Second, we used the document-topics matrix of probabilities from this standard LDA to create a distance matrix and clustered the distance matrix using HDBSCAN. Third, to parallel our compositional approach more closely to similarity, we obtained separate LDA solutions of the trigger text and the action text, created distance matrices for each solution, and averaged the two distance matrices which we then clustered using HDBSCAN.

Considering this, we organize Web Appendix H as follows. In section H1 we present the results of the standard LDA of the applet text. Section H2 uses HDBSCAN to cluster the 20,675 applets based upon the topic probabilities in the document-topic matrix from section H1. In section H3, we obtain HDBSCAN clusters based on a compositional LDA of the trigger and action text. Section H4 compares the two LDA-based HDBSCAN solutions with our word embedding-based solution. The results are summarized in section H5. As these analyses show, topic models and word embeddings provide very different clustering solutions and are not interchangeable techniques.

H1. Approach 1 - Dominant Topic from Standard LDA of the Applet Text

We fit a standard LDA topic model to our text corpus of 20,675 applets, following the same data cleaning protocols that we used for word2vec (punctuation removed, converted to lowercase, removed same stop words, stemming, and minimum word count). The Python scikit-learn 0.24.2 LatentDirichletAllocation package was used to fit all topic models. Following Berger et. al’s (2020) recommendation, we selected the number of topics by combining a statistical approach using minimum perplexity with researcher judgment. Figure H1 reports perplexity for LDA models from 10 to 600 topics, increasing in steps of 10. Perplexity is minimized for 350 topics. As this is appreciably higher than our 127-cluster solution based on word2vec embeddings, we visually inspected figure H1 and also identified a lower number of 180 topics based upon the heuristic of a decreasing rate of perplexity change (Zhao et. al 2015).

In many applications of LDA, topics are directly interpreted using the topics-keyword matrix, followed by a visualization of topic structure using MDS or tSNE (e.g., pyLDAvis: https://pyldavis.readthedocs.io/en/latest/ ). While this provides an understanding of the topics, it does not provide an understanding of the applets. A simple way to understand the applets is to identify the dominant topic for each applet. The dominant topic solution identifies, for each of the 20,675 applets, the topic with the highest probability in the document-topic matrix. This produced a higher-dimensional solution of 350 clusters for the 350-topic solution, and a lower-dimensional solution of 180 clusters for the 180-topic solution.
We used the Adjusted Rand Index (Hubert and Arabie 1985) to compare the dominant topic solutions with our 127-cluster embedding solutions. The Adjusted Rand Index ranges from 0 to 1, with 0 indicating a random relationship between two clustering solutions, and a 1 a perfect relationship. Recognizing that noise points do not define a homogeneous cluster, we excluded applets identified as noise points by our HDBSCAN embedding solution and considered only the 15,417 applets assigned to one of the 127 clusters. For these clustered applets, the Adjusted Rand Index of the embedding solution with the 350-cluster dominant topic solution was .152, and for the 180-cluster dominant topic solution was .162. Both indices reflect a very low similarity of clustering solutions.

To visualize some examples of this low similarity, figure H2 uses the UMAP plot of the embedding solution from figure 3 of the paper to highlight the four dominant topic clusters that have the largest number of applets (topics 174, 148, 41 and 34), using the 180-topic LDA. The spread of the applets in each of these four clusters throughout the UMAP plot shows a general lack of correspondence of the dominant topic solution and our 127-cluster embedding solution. Together, the very low Adjusted Rand Indices and the illustrative plots in figure H2 show that there is little correspondence between our HDBSCAN embeddings solutions and clustering based on the dominant topic.
FIGURE H2

TOP 4 DOMINANT TOPIC CLUSTERS FROM 180 TOPIC LDA PLOTTED USING UMAP FROM OUR 127 CLUSTER EMBEDDING SOLUTION
H2. Approach 2 - HDBSCAN Clusters Based on a Standard LDA of the Applet Text

A limitation of the dominant topic solution is that it forces each applet into a cluster defined by a single topic. In actuality, each applet is defined by a number of topics, some with higher probability than others. Unlike our word2vec embedding solution with 127 clusters and 5258 noise points, the dominant topic approach also does not permit interpretation in terms of territorialization and deterritorialization.

To capture the multi-topic nature of each applet and to detect deterritorialized noise points, our second approach uses HDBSCAN to cluster the 20,675 applets based upon the topic probabilities from the 350-topic solution (minimum perplexity from figure H1) and 180 topic solution (decreasing rate of perplexity change from figure H1) from Approach 1. In effect, we are substituting LDA topic probabilities for word2vec embeddings as a method for quantifying our text corpus.

One question is which metric to use in calculating the distance matrix among applets. In our embeddings approach, we showed in Web Appendix C1 that cosine distance and Euclidean distance produce essentially equivalent distance matrices (Mantel correlation = .992) and HDBSCAN cluster solutions (Adjusted Rand Index = .995). As noted earlier, this is because applet embeddings are unit length, and squared Euclidean distance for unit length vectors is proportional to their cosine distance. However, this is not the case for distance matrices calculated from topic probabilities. Thus, we obtained HDBSCAN solutions based on 180 and 350 topic probabilities using both cosine distance and Euclidean distance.

Before comparing the HDBSCAN solutions for LDA topic probabilities with our HDBSCAN solution from word2vec embeddings, we outline our third approach using a compositional approach to topic models.

H3. Approach 3 - HDBSCAN Clusters Based on Compositional LDA of the Trigger and Action Text

A limitation of using a distance matrix calculated from topic probabilities of an LDA of applet text is that the LDA topics are obtained from text in the entire applet, without considering the separate roles of trigger and action components. Thus, LDA does not directly allow us to separate the effects of triggers and actions as does our embedding approach using compositional similarity. Therefore, to apply topic models in a manner consistent with the spirit of our compositional similarity approach, our third approach performs separate LDA analyses on the trigger text and the action text, and then averages the resulting two distance matrices.

For trigger text and action text, we fit models from 10 to 250 topics, increasing in steps of 10. Figures H3 and H4 plot perplexity for these topic models. Based on minimum perplexity, we selected 190 trigger topics and 170 action topics as our higher dimensional solutions. Visually inspecting figures H3 and H4 for decreasing rate of perplexity change, we also identified lower dimensional solutions of 120 trigger topics and 80 action topics. We then produced both cosine and Euclidean distance matrices for both triggers and actions and obtained an average cosine
distance matrix and an average Euclidean distance matrix. HDBSCAN solutions were obtained for both average distance matrices.

FIGURE H3
PERPLEXITY FOR LDA OF TRIGGER TEXT, 10 TO 250 TOPICS

FIGURE H4
PERPLEXITY FOR LDA OF ACTION TEXT, 10 TO 250 TOPICS
H4. Comparing LDA-Based HDBSCAN Solutions with Our Word Embedding-Based Solution

*Mantel Correlations Between Distance Matrices.* To choose which of the different LDA-based clustering solutions to compare with our 127 word2vec embedding clusters, we examined the degree of similarity between the distance matrix from each higher or lower dimensional LDA solution and the distance matrix from our word2vec embeddings. If the distances among applets based on LDA topic probabilities are similar (dissimilar) to distances among applets based on word2vec embeddings, then HDBSCAN clusters based on these distances should be similar (dissimilar). Thus, similarity of distance matrices is an indicator of the degree to which cluster solutions based on these distance matrices are likely to be similar. We calculated the Mantel correlations (Mantel 1967) to assess similarity between our word2vec embedding cosine distance matrix and the various distance matrices resulting from LDA. Table H1 reports the results. Table H1 shows that the lower-dimensional LDA solutions (second column) are more highly correlated with the word2vec embeddings than the higher-dimensional LDA solutions (first column). Therefore, our comparisons will proceed with the lower-dimensional LDA solutions.

**TABLE H1**

**MANTEL CORRELATIONS BETWEEN WORD2VEC EMBEDDING COSINE DISTANCE AND LDA DISTANCE**

<table>
<thead>
<tr>
<th>Approach</th>
<th>Higher Dimensional LDA 350 Applet Topics, 190 Trigger and 170 Action Topics</th>
<th>Lower Dimensional LDA 180 Applet Topics, 120 Trigger and 80 Action Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approach 2 (Standard):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applet Cosine LDA</td>
<td>.298</td>
<td>.336</td>
</tr>
<tr>
<td>Approach 2 (Standard):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applet Euclidean LDA</td>
<td>.155</td>
<td>.185</td>
</tr>
<tr>
<td>Approach 3 (Compositional):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average Cosine LDA</td>
<td>.306</td>
<td>.364</td>
</tr>
<tr>
<td>Approach 3 (Compositional):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average Euclidean LDA</td>
<td>.218</td>
<td>.278</td>
</tr>
</tbody>
</table>

We also note that for both cosine and Euclidean distance matrices obtained from LDA topic probabilities, approach 3 (composition similarity using the average of separate LDAs of trigger and action text) has higher correlations than approach 2 (standard LDA of applet text). This makes sense given that our word2vec embedding approach is based on compositional similarity. In addition, the cosine distance matrices from LDA have a higher correlation with the...
word2vec embedding cosine distance matrices than the Euclidean distance matrices from LDA. Thus, we identified the average LDA cosine distance from the 180-topic solution as most likely to generate an HDBSCAN solution comparable to our solution using word2vec embeddings.

**HDBSCAN of Compositional Cosine Similarity from 180 Topic LDA.** We performed an HDBSCAN clustering of the average (e.g., compositional) cosine distance matrix from the 180 topic LDA solution. To maintain comparability with our HDBSCAN solution based on word2vec embeddings, we used the same hyperparameters.

The HDBSCAN of the average cosine distance matrix from the 180 topic LDA solution produced 79 clusters with 9,660 applets classified as noise. In comparison, our 127-cluster solution based on word2vec embeddings had only 5,258 noise points. Across both of these solutions, a total of 6,582 applets were jointly assigned to a cluster (i.e., non-noise). For these 6,582 applets, the Adjusted Rand Index was .265. This is larger than the Adjusted Rand Index for the LDA dominant topic solution, but still represents a low similarity between HDBSCAN solutions based on word2vec embeddings and based on topic probabilities from the 180 topic LDA. In addition, only 32% of all applets were assigned to a cluster by both solutions.

As we did for the dominant topic LDA solution, to visualize some examples of this low similarity between cluster solutions, figure H5 uses the UMAP plot of the embedding solution from figure 3 of the paper to highlight the four HDBSCAN clusters from the average LDA cosine distance matrix that had the largest number of applets (clusters 39, 33, 21, 35). The spread of the applets in each of these four clusters throughout the UMAP plot shows a lack of correspondence of the average LDA cosine distance and our 127-cluster embedding solution. However, in line with the higher Adjusted Rand Index, figure H5 shows less dispersion of the average cosine distance clusters through the UMAP plot than in the dominant topic clusters plotted in figure H2. In fact, three of the four subplots in figure H5 (clusters 39, 21, and 35) show that the HDBSCAN clusters from the average cosine distance of LDA topic probabilities represent broader grouping of HDBSCAN clusters from word2vec embeddings. The proposed compositional LDA approach appears, at least for these clusters, to fit our embeddings solution better than a standard dominant topics clustering.
FIGURE H5

TOP 4 HDSBSCAN CLUSTERS FROM 180 TOPIC LDA PLOTTED USING UMAP FROM OUR 127 CLUSTER EMBEDDING SOLUTION.
H5. Conclusion

Taken together, these results indicate that word embeddings and topic probabilities generally lead to different and largely unrelated cluster solutions, although a compositional approach to LDA appears to bring the cluster solutions into greater alignment. While it is beyond the scope of this paper to provide an in-depth comparison of the relative performance of these two approaches, we can make a few observations.

Our embedding solution is inherently more parsimonious than the LDA solutions, being based on 25-dimensional word vectors that were added together to create trigger and action vectors. The dimensionality of our LDA solutions is much higher. It has been noted that word2vec provides a dense vector filled with numbers while LDA provides a sparse vector of probabilities. As such, LDA has a more intuitive interpretation, but word2vec has greater flexibility (torsellio 2016). For our purposes, the dense lower-dimensional word2vec vectors can accomplish more than the sparse higher-dimensional LDA vectors.

An HDBSCAN software limitation also favors word2vec. At this time, HDBSCAN can predict assignment of new data points into existing clusters using Euclidean distance, but not using cosine distance. For unit length embedding vectors, Euclidean and cosine distance can be used interchangeably. So, we were able to use Euclidean distance to predict cluster membership of unrealized applets using HDBSCAN. For topic probabilities, Euclidean and cosine distance produce different distance matrices. However, cosine distance cannot be used by HDBSCAN for prediction, providing some constraints on analysis when LDA topic probabilities are used.

Since topic models provide a familiar approach to easily identify, interpret and visualize topics, it is tempting to view topic models as an alternative to our approach. At first glance, topic models seem to spare the researcher from clustering the data with HDBSCAN and visualizing it with UMAP. However, as we have noted, understanding topics is not the same as understanding applets. Instead, topic models could be considered as an alternative to word embeddings for the purpose of quantifying text data for the purpose of subsequent clustering. In that case, our methodology using HDBSCAN and UMAP would still be needed. However, given the clear differences in clustering solutions, we would caution that researchers who are familiar with topic models should not substitute topic models for word embeddings in implementing our approach, assuming the two are interchangeable. The relative merits of word embeddings, topic models, and other approaches for quantifying text data in preparation for clustering is an important subject for further research.

However, given they differ, we believe word2vec is the more appropriate approach for our assemblage theory conceptualization. Word2vec learns embeddings for a set of words. In our highly structured corpus, only 25 dimensions are required (LDA required many more topics). Once learned, these embeddings can be added together in a very flexible manner to create vectors for triggers, actions, and complete applets. These embeddings can be created not just for existing applets, but also for unrealized applets. This “building block” process allows us to represent assemblages (applets) in terms of their components and capacities (words), a natural fit with assemblage theory.
Topic models such as LDA, on the other hand, represent applets as a mixture of topics, with probabilities for each topic. While word embeddings represent applets as a function of their parts, LDA represents applets in terms of a mixture of topics that themselves need to be interpreted. The topics do not have clear assemblage theory interpretation, themselves being a mixture of words, with probabilities for each word.

As we have noted, cosine similarity is not an arbitrary metric for word embedding-based vectors, but a metric that is required for the deeper similarity relationships among applets and the words defining these applets to hold true. While we can certainly calculate cosine similarity among topic probabilities in LDA, there is no deeper interpretation as there is with word embeddings, and many other similarity metrics would be reasonable with LDA. The unambiguous choice of a meaningful metric for our particular problem is another advantage of using word embeddings.
WEB APPENDIX REFERENCES


Le, Quoc and Tomas Mikolov (2014), "Distributed Representations of Sentences and Documents,” In International Conference on Machine Learning, 1188-1196.


McInnis, Leland (2016), “Is Any Cluster Algorithm Suitable for this Situation?” Reddit comment https://www.reddit.com/r/MachineLearning/comments/3r2och/is_any_cluster_algorithms_suitable_for_this/


torsellio (2016), “A Tale About LDA2vec, When LDA Meets word2vec;”
https://www.datasciencecentral.com/a-tale-about-lda2vec-when-lda-meets-word2vec/

