Automatic Assignment of Topical Icons to Documents for Faster File Navigation

Rishiraj Saha Roy*, Abhijeet Singh†, Prashant Chawla‡, Shubham Saxena§ and Atanu R. Sinha¶
*Max Planck Institute for Informatics, Saarland Informatics Campus, Germany. Email: rishiraj@mpi-inf.mpg.de
†Computer and Information Science, University of Pennsylvania. Email: abhsingh@seas.upenn.edu
‡Adobe Systems India. Email: pchawla@adobe.com
§BlackRock Services India. Email: shubham.saxena@blackrock.com
¶Big Data Experience Lab, Adobe Research, India. Email: atr@adobe.com

***Abstract*—Several computer users neither assign names to their documents systematically nor organize them into suitable folders, making it difficult to search for relevant files when needed. While this problem can be addressed in several ways, we explore the novel approach of automated assignment of topical icons to documents in order to cue memory for faster navigation. Specifically, we overlay the currently available generic software-oriented file association icons like Acrobat, Word or Powerpoint on documents with algorithmically assigned icons that are specific to the topical content of the documents. Our pipeline method uses document clustering, significant-phrase extraction, phrase generalization and phrase vector matching for assigning icons to documents. Experimental results show that *topical iconification* significantly speeds up document navigation time vis-à-vis content-based *file naming*, in both a controlled laboratory setup as well as in a crowdsourced study. Icons assigned by our algorithm are observed to have satisfactory inter-annotator agreement with respect to their meanings.**

I. Introduction

Motivation. Searching for files in a folder is a fundamental task in human-computer interactions [1], [2]. In this research, we focus on text-based document files, that are currently associated with icons of software that open the files – for example, Adobe Acrobat icons for *pdf* files, and Microsoft Word or Powerpoint icons for files with *docx* or *pptx* extensions. For computer users who do not name or organize their files systematically, we posit that assigning icons that are specific to the topical content of the documents (hereafter, topical icons) makes icons relevant to content and cue human memory better for improved navigation. The general benefits would also include faster browsing and navigation within the document repository, better organization, and overall, an improved user experience. We refer to an assignment of icons to documents as "iconification". While automated topical iconification can potentially address several use cases, we show that it can speed up file navigation for typical computer users.

Limitations of current technology. Potentially, thumbnails or previews may appear for files, aiding file search. But, they are usually very small snapshots of the first page that are often not sufficient for inferring the category of the document. Moreover, looking at the preview for each document in a collection is highly time consuming. Operating systems today also allow manually setting customized icons for files, but this option is rarely used [3]. While the general problem of efficient file search can be addressed in several ways, in this work we explore the effects of topical iconification. We are motivated by past research [4], [3], [5] which shows that pictures are more easily remembered than words, visual information can be processed both sequentially and in parallel, and humans resist reading text but prefer interaction with icons [6].

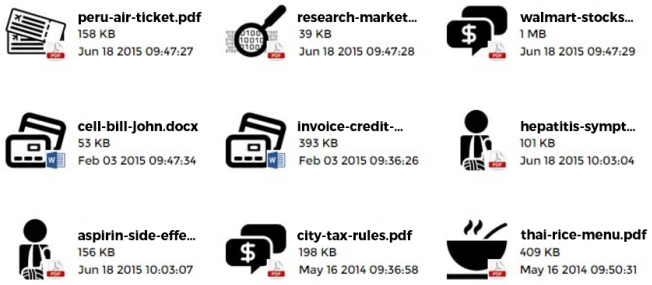

Fig. 1. Document files with topical icons assigned by our algorithm. Original software icons can be seen at the bottom-right corners of our icons.

Approach. Our method needs a set of documents, a concept taxonomy and a tagged list of icons as input, and produces iconified documents as output, where each document is associated with a topical icon. We first perform a content-based clustering of our document repository. Next, we extract significant phrases from documents in each cluster. This is followed by a generalization of the significant phrases using the concept taxonomy. Finally, we compute the cosine similarity (using word embeddings) between the generalized phrase list from a cluster and each of the icon tag lists, and assign the icon with the highest similarity to all documents from the cluster. Representative output from our algorithm is shown in Fig. 1.

Contributions. We present the first method for assigning topical icons to documents, and show that files with topical icons significantly improve file navigation over that of files with content based names and software icons. We make our evaluation dataset comprising 30 file navigation tasks, their correct answers, and the associated retrieval corpus of 60 documents, publicly available for research use[1].

Organization. The rest of the paper is organized as follows.

[1]http://people.mpi-inf.mpg.de/~rsaharo/icdar17data.zip

In Sec. II, we survey relevant research efforts from the past. This is followed by our method for document iconification (Sec. III). Next, in Sec. IV, we present details about the resources used. We desribe experiments in Sec. V, followed by a section on insights and general discussion (Sec. VI). Finally, in Sec. VII, we make concluding remarks and highlight promising avenues for future work.

II. Related Work

Icons and their usage. Haramundanis [5] and Horton [7] state that icons are useful as reminders, can aid recognition, and can assist cross-lingual communication. The authors suggest that a small number of icons used consistently across a software can help the average user. Also, icons should be accompanied by associated text to improve their efficacy [5], [8], and have been shown to be potentially useful in file search [6]. In our case, file metadata such as name, date and size may serve as such texts. Lewis et al. [3] propose a sophisticated method for generating icons (termed VisualIDs) from file names based on shape grammars. By the authors' own admission, since each file with a unique name gets a unique icon, this approach may not really help cognition for even moderately large collections. Files with similar names obtain somewhat similar icons while file content is ignored. Such assumptions may not be reasonable for several scenarios in reality. We depart from such works by explicitly recognizing the file content's primacy in generating icons.

Setlur and Mackinley [9] present a method that takes as input descriptions of categorical data such as phones, sports or even human teeth types, and finds images from the Web that can represent these classes. However, documents are not considered as inputs. Gatsou et al. [4] show that icons can improve navigation experience on mobile interfaces. Hemenway [10] and Mendling et al. [11] suggest that along with files, icons can also be associated with functions or actions in a process. Gatsou et al. [12], [4] and Koutsourelakis and Chorianopoulos [13] suggest that there is a relationship between age and icon usage comfort, and that icons need to be designed separately for different demographic segments. Kaur [14] examines effective techniques for designing animated icons for children. Icons have been analyzed from the perspective of cursor navigation effectiveness as well, with Worden et al. [15] proposing sticky icons that pull the cursor towards themselves, and Whisenand and Emurian [16] studying the effect of the angle of approach of the cursor on icon selection.

There have been significant efforts in the industry towards *icon creation*. Organizations like Flaticon, Axialis Software, Iconfinder, IconExperience, Fontastic, and Behance have created large repositories containing thousands of icons. Some of these have manually assigned textual tags associated with each icon.

Personal information management. We note that our research falls in the general area of personal information management (PIM) [17], [1], [2]. Barreau and Nardi [18] summarize early studies of user behavior in organizing and finding files on computers with different operating systems. Jones [17] and Fitchett [2] provide nice overviews, highlighting challenges, current techniques and potential research directions. Elsweiler [1] studies different file-finding tasks and proposes *task-based evaluation* of file-finding strategies, which inspires our evaluation study. Robertson [19] explores the possibility of aligning icons on a 3D desktop which may improve the accessibility of target documents.

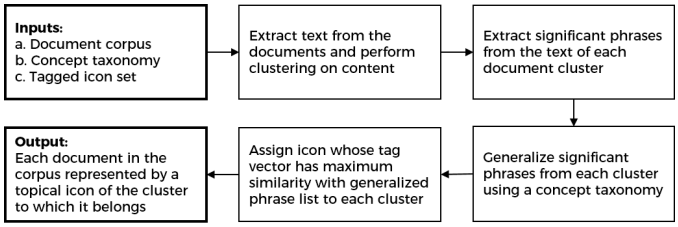

Fig. 2. An overview of our method for topical iconification.

We contribute in two ways by filling the void in current research and practice. First, we introduce the use of *topical icons* in identifying document files, which has not been explored before. Second, we propose a method for automating the icon *assignment* process from icon repositories to documents using content analysis.

III. Method

We now discuss the steps of our method, which requires a document folder where files depict currently used software icons, a concept taxonomy and a tagged icon repository as input, and produces an "iconified" folder where each document has a topically relevant icon, as output (overview in Fig. 2.). We refer to general areas of interest like *finance*, *medicine* and *travel* as topics in this work, but in reality it depends upon the context. For example, if a document collection has only financial documents, relevant topics may center around *taxes*, *stock reports* and *bank loans*. We assume that one document can be associated with a single dominant topic. Multi-thematic documents are beyond the scope of this research.

Step 1: Document clustering. Since it is not helpful to assign a unique icon to every individual document in a corpus, we first organize the dataset into document clusters. We extract textual content from each document and perform a clustering based on the words from each document. We prefer unsupervised clustering over supervised classification into a predefined set of categories, because clustering allows for *context-specific iconification*, where the icon assigned to a document can vary with the context in which it is located. Here, context is defined by the file's neighborhood, i.e, other files in the same folder.

Step 2: Phrase extraction. Every word or phrase (by phrase we refer to an n-gram) in a document cluster is not meaningful enough to play a role in icon assignment. As a result, we need to extract representative phrases from each cluster. For this purpose, every word or phrase from a cluster is first scored by its statistical significance. Any method of scoring phrases, such as those used by topic modeling algorithms like El-Kishky et al. [20] and Hulpus et al. [21], or techniques that compute

word association measures [22], may be used to obtain phrase significance scores. To rule out noise, we then extract the top-α phrases from the cluster that have a minimum support count (frequency) of β. This is done by treating a cluster as a "super-document" created by the concatenation of all its constituent documents, and running a phrase extraction process on it.

Step 3: Concept generalization. Significant phrases extracted from each cluster cannot be used directly for iconification. This is because it is not desirable to have icons that apply to only specific documents in the cluster, and we want all documents in a cluster to have the same topical icon. To generalize the phrase list to a level that applies to the cluster as a whole, we use the Wikipedia concept taxonomy, a comprehensive classification of concepts. This taxonomy is represented as a directed acyclic graph (DAG), with concepts ("parent nodes") pointing to sub-concepts ("child nodes"). A parent node can have multiple children and a child node may have multiple parents. For example, the parent `Cuisines of Italy` has children nodes `Pizza` and `Pasta`, while the child node `Pizza` has both `Cuisines of Italy` and `Flatbread` as its parent nodes.

Each significant phrase from a cluster is mapped to a "source node" in the taxonomy using exact, stemmed [23], lemmatized [24] or pluralized string matching. We then define a *parent graph* for every source node by going up a fixed number of levels (say, δ) from the source node and adding nodes and edges from the DAG to the source node. Each parent graph is therefore a subgraph of the entire taxonomy. Parent graphs area allowed to have several nodes in common.

We then score each node in each parent graph in *inverse proportion to its distance* from the source node. This is because the specificity of the topical phrase decreases as we go higher up in the taxonomy, and we want icons to be as specific as possible. Source nodes are not scored. However, a specific source node may appear in the parent graph of another source node, in which case it is scored as usual. Scores of nodes which appear in multiple parent graphs are aggregated using a simple sum. Nodes accumulating higher scores are assumed to be more representative of the cluster as a whole. A node N_δ in a parent graph (at a maximum distance of δ from the source node) is scored as in Eq. 1:

$$phraseGeneralizationScore(N) = k \sum_{p \in P, d(\cdot,\cdot) \leq \delta} \frac{1}{d(N_\delta, N_p)^\gamma} \quad (1)$$

where N_p is the source node for phrase p in significant phrase list P, $d(\cdot,\cdot)$ $(\leq \delta)$ is the number of hops (or "distance") between N_p and N_δ in the DAG, and k (constant of proportionality) and γ (penalty factor on distance from source node) are tunable parameters. We then select the top-scoring θ node labels to be the generalized phrase list representing each cluster. Since source nodes are not scored, $d(N_\delta, N_p)$ is never zero in Eq. 1.

Step 4: Icon assignment. We assign an icon i_c to a cluster c, that will be associated with every document in c. For this, we treat the generalized phrase list from c, denoted by gp_c, as a phrase vector. Also, for each icon i from the full icon repository I, we refer to the set of associated tags (tags may be single or multi-word) as the tag vector t_i for that icon. We then compute the cosine similarity between the generalized phrase list gp_c and each icon tag vector t_i using *word embeddings*. We select the icon with the highest cosine similarity as the icon for all documents in the cluster c, as in Eq. 2:

$$i_c = \arg \max_{i \in I} cos(gp_c, t_i) \quad (2)$$

TABLE I
QUERIES ISSUED THROUGH THE BING API USING EXPLICIT FILETYPE SPECIFICATIONS FOR COLLECTING DOCUMENTS FOR EACH CATEGORY.

Category	Web search queries
Medical (1762)	`"disease symptoms", "drug report", "side effects"`
Legal (1325)	`"court law", "lawsuit", "legal"`
Travel and holiday (1198)	`"bus time table", "holiday package", "hotel resort brochure"`
Food (1041)	`"food menu", "food recipe", "restaurant menu"`
Finance (3375)	`"finance", "stock report", "tax"`
Lecture notes (1414)	`"lecture", "lecture notes", "lecture slides"`
Bills and receipts (8185)	`"bill", "invoice", "receipt"`
Research papers (3396)	`"approximation algorithms", "machine learning", "molecular biology"`
Forms (1495)	`"application form", "joining form", "membership form", "passport form"`
Device manuals (1162)	`"technical specifications", "user guide", "device manual"`

IV. RESOURCES

Document corpus. "Personal" documents are inherently private assets and datasets for them are not easily available publicly. We thus created our own corpus as follows. We identified ten common categories for such documents (*medical, financial, food, travel and holiday, legal, lecture notes, bills and receipts, research papers, forms* and *device manuals*) and issued corresponding Web search queries (like `"side effects"` for *medical* and `"holiday package"` for *travel and holiday*) through the Bing Search API[2]. We fetched the top-2000 documents in PDF, Word, Powerpoint, or plaintext formats belonging to these categories. The `filetype:` advanced search operator with arguments `pdf, doc, docx, ppt, pptx` or `txt` *was always explicitly used* to exclude HTML webpages which are not representative of typical user documents. **Double quotes** were always used around the queries for *exact phrase matching*, ensuring higher precision of the search result documents. Text was extracted from each document using Apache Tika[3] and

[2]https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/, Accessed 22 August 2017.

[3]https://tika.apache.org/, Accessed 22 August 2017.

only ASCII documents with more than 30 words were retained. We randomly sampled $1,000$ documents from each category to build our final corpus of $10k$ documents. Exact queries used and the number of documents collected, for each category, are provided in Table I.

Concept taxonomy. We used the Wikipedia taxonomy[4], represented as a DAG with $473,639$ nodes and $995,122$ directed edges from parent to child. The largest (weakly) connected component (LCC) in the DAG has $466,142$ nodes and $989,949$ edges (an average node degree of 4.247, considering an undirected graph). The LCC has a clustering coefficient of 0.033, suggesting the Wikipedia concept graph to be quite sparse [25].

Icon repository. A set of $35,287$ freely available tagged icons from Flaticon[5] forms our icon repository. Each icon has between three and nine description tags with mean as 6.67, indicating rich annotations in general. Sample tags include `leisure`, `supermarket`, `bar graph`, `fast food` and `health clinic`.

V. Evaluation and Insights

We now present our experimental setup, followed by the results and insights obtained from our analysis. Typical document files represented by topical icons as assigned by our algorithm are shown in Fig. 1 (Sec. I). Since a user may also want to use his/her *recollection of the filetype* (`PDF`, `Word`, `Powerpoint`) during the document search process, we **overlay** topical icons with software icons (bottom-right corners of our icons in Fig. 1).

A. Evaluating navigation efficiency

Setup. We define **two scenarios**: **(1)** files having **content-based names** (with usual software-relevant icons), and **(2)** files having content-based names with **topical icons** assigned by our algorithm. Files in scenario 1 would have names such as those in Fig. 1 with only the usual software icons of *PDF* or *Word*, while the figure shows scenario 2 as is (content-based names and topical icons, with software icons overlayed at the corners). This technology is perhaps more suited to users who do not systematically rename their files. However, choosing default file names that documents have on the Web (which are often arbitrary) could heavily bias the results in our favor. In other words, under scenario 1, file search tasks could become much more difficult to solve if default file names are used. Unfortunately, every user has their own convention of naming files, and renaming files ourselves would again entail biases in file naming. Thus, as a *fair* alternative, we assigned **algorithmic names** based on *content* by the hyphenated concatenation of the top three phrases from the document extracted by the *state-of-the-art topic modeling algorithm* ToPMine [20]. This serves as an aid to the search process and may also be perceived as a comparison baseline for **automated document tagging**. We hypothesize that topical iconification with content-based names improves file navigation efficiency (**visual + textual search space**) over the baseline of content-based names with software icons (essentially a **text-only search space**).

To show improvements in file navigation efficiency, we devised 30 file searching tasks for the two scenarios, through crowdsourcing on Amazon Mechanical Turk (AMT). We also conducted pilot experiments in a lab setup, as discussed later. The participants had to search for a unique file from a corpus of 60 document files in both scenarios (simulating a *My Downloads* folder). It is important to note that, while a computer user may have many more files in a typical usage scenario, having more files in an experimental setup could easily lead to very long task completion times and task fatigue, especially for scenario 1, biasing results in our favor. These 60 documents were sampled randomly from our $10,000$ document corpus that had been iconified by our method. Thirty documents were picked at random from this 60-document subset. Then, the 30 file searching tasks were designed keeping in mind these 30 documents as target answers.

For example, two tasks were: (a) *"Your cook is not coming today. So you want to make some tasty lunch by following the Mr. Food eCookBooks recipe that you know you have in your Downloads folder, but do not remember other details. Find the document containing this recipe."*, and (b) *"You are planning to buy financial shares of Morris Habitat. For that, you need to find the shareholder policy of Morris Habitat, that you know you have in your Downloads folder, but do not remember other details."*. For each of the 30×2 task-scenario pairs, we elicited six annotations (each annotation is a solution, i.e., a navigation to the correct file), resulting in a total of 30 (tasks) $\times$ 2 (scenarios) $\times$ 6 (annotations) $=$ 360 annotations. Each user was asked to complete six such task-scenario pairs (three tasks, and in each task, the two scenarios) to allow for familiarization with the new interface. We thus required $360/6 = 60$ human annotators (Turkers). For task realism, all participants were provided two *sort buttons*: `sort-by-name`, and `sort-by-icon`.

We use four **metrics** for each provided task to *objectively measure navigation efficiency*: (a) total time taken (seconds) to find the correct file, (b) number of files opened for examination in the process, (c) number of screen scrolls performed, and (d) the number of files hovered on during the search. While the total time taken is an aggregate measure of file search success, the other metrics reflect the relative reduction is the search space for an user.

The AMT HIT. Each unit task on AMT is referred to as a HIT (Human Intelligence Task), and the details for our setup are presented in Table II. Since the AMT interface is not very amenable to a desktop-browser-based study like ours', we provided the training instructions on AMT but hosted the documents and the experiment on internal servers at the authors' organizations. On successful completion of the HIT, Turkers were provided with a unique code which they had to insert into an HTML form after being redirected to an AMT page. Those who exceeded the maximum time allotted, or violated any of the guidelines (not opening a file before

[4]http://wikicategory.sourceforge.net/, Accessed 22 August 2017.

[5]http://www.flaticon.com/, Accessed 22 August 2017.

TABLE II
DETAILS OF THE HIT AS POSTED ON AMT.

Parameter	Details
Title	Study on File Search Behavior
Task description	The purpose of this simple study is to understand how people search for files on their computers. You will have to find the relevant file for a question among a small collection in different scenarios.
Task keywords	File search, File icons, Computer-based study
Alloted time per HIT	60 minutes
Actual time per HIT	38 minutes (mean)
Turker qualification	HIT approval rate $\geq$ 95%; No. of HITs approved $\geq$ 1000
Turker location	Any
Reward per HIT	\$2.25
#Questions in a HIT	6

TABLE III
EVALUATION OF FILE NAVIGATION EFFICIENCY.

Metric → / Scenario ↓	Time taken	Screen scrolls	File opens	File hovers
Lab Pilot				
(1) Content-based names	101.55	16.37	**1.41**	12.22
(2) Topical icons	**77.39***	**12.22**	1.71	**11.35***
Crowdsourced (AMT)				
(1) Content-based names	97.33	15.55	1.99	11.34
(2) Topical icons	**90.21***	**13.41**	**1.47**	**9.56***

For each setup, the lower value in a column is marked in **boldface**. * marks statistical significance of topical icons over content-based names ($p < 0.05$).

submission, or made more than five incorrect submissions for a single question) were not provided with the unique code. The training video[6] was about nine and a half minutes long, and was hosted on YouTube. Since the instructions for a task like ours' were different from a typical AMT annotation task, we used a *screening quiz* about the guidelines. Turkers had to get all the questions right before they could begin.

Avoiding biases. Besides other bias-avoiding measures described above, the order of presentation of the scenarios and the tasks were randomized to control order-bias. As an additional precaution, *the order-position of the scenario was used as a covariate in subsequent analyses.* To prevent bias arising out of familiarity with the corpus, no annotator performed the same task in more than one scenario.

Lab pilot. To test our experimental setup, we conducted a small pilot inside Adobe India. The same 30 tasks in the two scenarios were used, and we obtained 96 annotations from 47 volunteers, each annotation being a solution for a task-scenario pair. Some task-scenario pairs thus received more than one annotation (at least $30 \times 2 = 60$ annotations were requested in total). All volunteers were undergraduate Computer Science interns. Same measurements and analyses were performed. We now present results of lab and AMT studies.

Results and analysis. Results on the evaluation of navigation efficiency, aggregated over all Turker annotations by scenario, are presented in Table III. Topical icons are observed to perform better than content-based names across all the four **dependent measures** (metrics): total time taken (90.21 seconds vs. 97.33 seconds), in number of scrolls (13.41 vs. 15.55), in number of files opened and examined on the way to the solution (1.47 vs. 1.99), and in number of file hovers required for skimming file properties (9.56 vs. 11.34). Users extensively used the *sort-by-icon functionality* that brings documents of the same topic adjacent to each other. We believe that this substantially reduces the search space and is the primary cause behind the increased navigation speed for the topical icons' scenario. We also present the lab pilot results in the same table, which showed the desired trends and gave us strong confidence in the efficacy of the proposed iconification algorithm and the evaluation setup. The lab pilot helped us understand several issues with the interface and subsequently strengthen the task guidelines on AMT.

[6]https://goo.gl/vd3W1k, Accessed 23 August 2017.

TABLE IV
MANOVA FOR TOPICAL ICONS VERSUS CONTENT-BASED NAMES.

MANOVA Statistics	Value	F-Value	Num DF	Den DF	*p*-value ($Pr > F$)
Lab Pilot					
Wilks' Lambda	0.9335	1.74	4	98	0.1463
Pillai's Trace	0.0665	1.74	4	98	0.1463
Hotelling-Lawley Trace	0.0712	1.74	4	98	0.1463
Roy's Greatest Root	0.0712	1.74	4	98	0.1463
Crowdsourced (AMT)					
Wilks' Lambda	0.8872	3.66	4	115	0.0077
Pillai's Trace	0.1128	3.66	4	115	0.0077
Hotelling-Lawley Trace	0.1272	3.66	4	115	0.0077
Roy's Greatest Root	0.1272	3.66	4	115	0.0077

Num = Numerator χ^2 dist., Den = Denominator χ^2 dist., DF = Degrees of Freedom

Since four dependent measures were generated for each task, a *multivariate analysis of variance (MANOVA)* was used. Furthermore, since each participant compared the two scenarios (for different tasks), the *within-subject* nature of the comparison was recognized within the MANOVA by using the appropriate error term for the test statistics. The analysis was run on SAS using *Proc. GLM*. The results are shown in Table IV. Each of the four quantities (row headers) tests the multivariate hypothesis of no difference in the dependent measures, between the two scenarios. More on these test statistics can be found in Morrison [26]. MANOVA shows that for the four dependent measures, there is a significant overall difference in task performance between the two scenarios ($p = 0.0077$, Table IV), *favoring topical icons over content-based names*. Separate analyses for each of the dependent measures show that there are statistically significant ($p < 0.05$) improvements with regard to total time taken and number of hovers ($p = 0.02, 0.0004$ respectively). In terms of number of scrolls and number of files opened, there is no significant difference ($p = 0.07, 0.17$). This statistical significance is marked with an asterisk (*) in Table III.

We expect that benefits from topical icons will be even more pronounced in realistic situations when typical users deal with a hundred documents or more in a large folder. Thus, our experimental claim of superiority is conservative. In practice, users would likely use a combination of name search and different sorting techniques to locate files in a large directory.

TABLE V
DOCUMENT CLUSTERING EVALUATION ENABLED BY THE PRESENCE OF GROUND TRUTH CATEGORY LABELS.

Algorithm	NMI	Purity	F-Score	MaxMatch
Blockclustering [27]	**0.500**	0.647	**0.627**	**0.604**
ToPMine [20]	0.091	0.300	0.292	0.290
K-Means [28]	0.463	**0.705**	0.537	0.477

The maximum value in a column is marked in **boldface**.

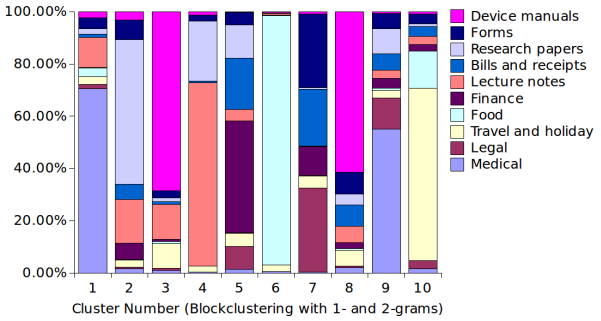

Fig. 3. Document category distribution within obtained clusters.

B. Measuring agreement on icon semantics

A good iconification algorithm is expected to produce icons that have significant agreement on their interpretations or semantics among users. We measured user agreement on the meanings of icons generated by our algorithm by providing users with icon-tag association tests, on both AMT and in the lab pilot. Users were presented with ten icons (one for each cluster) generated by our algorithm, and a set of tags corresponding to our ground truth category labels. They were asked to mark (possibly multiple) tags they deemed relevant for each icon. We found that for seven (corresponding to clusters *medical, legal, travel and holiday, food, finance, lecture notes,* and *research papers*) out of ten icons, at least 70% of the users agreed on a single tag for that icon. We believe that this is satisfactory agreement. For the remaining three icons 39% to 54% users agreed on a single tag. This poor agreement most likely corresponds to the impure clusters of non-topical documents (*bills and receipts, forms*, and *device manuals*). While these are indeed document categories per se, they may contain all types of words (*bills*, say, may come from several sources like hospital visits, groceries or telephone usage) and hence in general difficult to cluster based on textual content. This in turn causes semantic ambiguity in the retrieved icons. *Structural cues* are expected to detect such categories better than content, which we plan to do as future work. Results were very similar in the lab pilot ($\geq 70\%$ users agreed on a single tag for $8/10$ icons). Some of these icons can be seen in Fig. 1.

C. Experiment specifics

We now present the specific design choices in our iconification algorithm, that provide insights into the efficacy of each of our building blocks.

Document clustering. Since we defined ten document classes or categories to begin with (Sec. IV), the input *number of clusters* were set to ten. In practice, the number of clusters (and hence icons, since there is a one-to-one mapping between the two) that can satisfactorily represent all documents in a user's personal computer, while preserving navigation efficiency, depends largely on (a) the cognitive ability of the user, and (b) the topical diversity of his/her document collection. There is essentially a trade-off between these two factors – while more icons will present a higher discriminative power between files (as in Lewis et al. [3]), they will create a higher cognitive load on the user, thus potentially degrading retrieval efficiency.

Availability of ground truth in terms of category labels of documents enabled us to perform an external evaluation of clustering with different families of algorithms. We used the well-established metrics of *normalized mutual information (NMI), Purity, F-score* and *Maximum Matching* [29]. We present the results of the following three techniques – blockclustering [27], ToPMine topic modeling [20], and traditional k-means [28]. Blockclustering is a co-clustering algorithm that clusters documents and phrases simultaneously and associates each document cluster with a unique phrase cluster. Topic modeling algorithms provide a distribution of topics for each document, and we obtain a hard clustering by assigning a document to its most dominant topic.

An evaluation of clustering is shown in Table V. We observed that blockclustering produces the best results on three out of the four metrics. Thus, we use this clustering algorithm in the iconification pipeline. The relatively poor absolute performance of all these content-based clustering algorithms (maximum NMI of 0.5, while the upper bound is 1.0) is due to the presence of non-topical categories like *bills and receipts* and *forms*. We plan to incorporate knowledge of document structure into the clustering process as future work (bills, forms and research papers are often associated with semi-structured templates). The class-wise breakup of the document clusters *as obtained using blockclustering* is shown in Fig. 3.

We found that the topical classes of *food*, *travel and holiday*, *medical*, *legal* and *finance* and a couple of non-topical classes *research papers* and *lecture notes* have dominant representations in single clusters. Successful clustering of apparently non-topical documents belonging to *research papers* and *lecture notes* is due to high skew in their topics in material available online (search results mostly include documents associated with Computer Science and allied domains).

Phrase extraction. We used the state-of-the-art topic modeling algorithm ToPMine [20] for phrase extraction. ToPMine extracts representative phrases by obtaining counts of frequent contiguous patterns, and then probabilistically reasoning about these patterns while applying contextual constraints to discover the final list of meaningful phrases. Representative phrases are not limited to unigrams, and we have expressions like `"medical insurance"`, `"fried rice"` and `"artificial intelligence"` in our final

TABLE VI
END-TO-END DETAILS FOR EACH OF THE TEN DOCUMENT CLUSTERS.

Id	$\|D\|$	$\|P\|$	$\|P^{Tot}_{Wiki}\|$	$\|GenP_{tag}\|$	Max. cos
1	874	100	38	59	0.756
2	950	100	29	75	0.812
3	619	100	45	73	0.701
4	669	100	31	75	0.639
5	1423	100	43	70	0.888
6	847	100	32	71	0.735
7	2338	100	38	76	0.680
8	732	100	44	63	0.632
9	534	100	39	59	0.809
10	927	100	31	70	0.673

lists. We set $\alpha = 100$, i.e., we extract the top-100 phrases from each cluster, according to their ToPMine significance scores. Minimum support count (frequency) of an n-gram (β) is set to five. Maximum n (as in n-gram) is chosen as three, i.e., we consider extraction of only unigrams, bigrams and trigrams.

Concept generalization. On an average, about 37 of the 100 ToPMine phrases per cluster (varying between 29 and 45) were found in Wikipedia (in exact, stemmed, lemmatized or pluralized forms), which is substantial for continuing with the generalization procedure. The number of levels to travel up the concept taxonomy δ, the constant of proportionality k, the penalty factor γ (Eq. 1) were chosen to be 3, 1 and 5 respectively. Applying the scoring function in Eq. 1, the top-100 ($\theta = 100$) nodes were thus extracted from Wikipedia as the generalized phrase list, for each cluster.

Icon assignment. On average, per cluster, about 69% (varying between 59% and 76%) of the generalized phrase lists from Wikipedia were present as icon tags in our repository (in exact, stemmed, lemmatized or pluralized forms). Cosine similarities between the generalized phrase vectors and the icon tag vectors are computed using Google Word2Vec [30] pre-trained on Google News data[7]. The average maximum cosine similarity, over the ten clusters, between the generalized phrase list and the icon tag list was found to be 0.732 (Eq. 2). This reasonably high cosine similarity implies higher confidence in the icon assignment process. End-to-end cluster details are shown in Table VI. *Id*, $|D|$, $|P|$, $|P^{Tot}_{Wiki}|$, $|GenP_{tag}|$ and *Max. cos* refer to the cluster number, number of documents in the cluster, number of phrases extracted from the cluster with ToPMine, total number of such phrases that matched in Wikipedia titles in any form, number of generalized phrases that matched with icon tags in any form, and the maximum cosine similarity between the generalized phrase list and an icon tag vector for the cluster, respectively.

Implementation details. For implementation, file icons in Windows Explorer were changed dynamically using *shell extension handlers*, specifically, icon handlers[8]. The entire system was implemented in Java with Python and R plug-ins on a single Windows desktop system with an Intel Xeon Quad-Core processor and 16 GB RAM. The end-to-end runtime was $\simeq 2.5$ hours on our corpus of $10,000$ documents.

[7]https://github.com/dav/word2vec, Accessed 22 August 2017.
[8]https://goo.gl/r5LX9b, Accessed 22 August 2017.

VI. DISCUSSION

Addition of new documents and categories to a folder. There is always a possibility that new files or documents belonging to new topics will be added to a directory. As a result, the clustering process will need to be re-run on the folder from time to time, either in batch mode with a set of new documents, or in an incremental mode with the addition of every new document. It would perhaps be cognitively preferable to the user if icons for existing documents did not change over iterations, i.e., the user may choose to exclude existing documents from newer iconification runs.

Icons and previews. To the extent previews or thumbnails are currently available, when it comes to finding a file, any user who has to hover over each potential file for a preview will find it time consuming. Also, an icon could cue one's memory more quickly. Relative efficiency between the proposed iconification versus the use of previews is an empirical issue, which needs to be evaluated. However, given our experimental demonstration, it can be well argued that icons will help over and above the availability of previews. This will happen because the mere presence of topic-relevant icons will enable the user to reduce the set of all potential files for a given search to a small subset, on which preview can be applied.

Change in a file content. We understand that a user may regularly update a file's contents. In the rare case that the change in content is big enough to alter the theme of the document, then one can argue that a different icon may be appropriate. However, since an icon's main purpose is to cue memory, it is possible that the user has already associated the current icon with that file, and the inappropriateness is no longer an issue. The other possibility would be to mark a file for re-iconification in case of change in major topical content.

Usability. Every user may not like or benefit from iconification. For some users who are more visual in their memory, having icons added to their already existing search tools will be beneficial, and for others who are comfortable with existing tools can turn iconification off. For example, the user can be given an override option to accept a topical icon, or maintain the default icon settings on his/her computer. Since many users do not name files appropriately on a regular basis suggests that they are already inflicted by unpredictability in the filenames, a common basis for search. It is hard to foresee how iconification can make matters worse for such users. The iconified files may be associated with a *legend* for disambiguating icons. The legend may be induced by automatically naming the clusters by topic aggregation methods [31], [32], [33]. The user can turn off the legend once (s)he is comfortable with icons and has developed the necessary intuition for the same. Parameters of the iconification algorithm will be hidden to the naïve user.

Advanced sorting capability. An important point to note is that while current file sorting can only be performed along one dimension (like name, filetype, last-accessed time, or size), iconification helps us perform *nested sorting* to reduce the size of our file search space substantially. As a specific example, sorting iconified files first by *icon* and then by

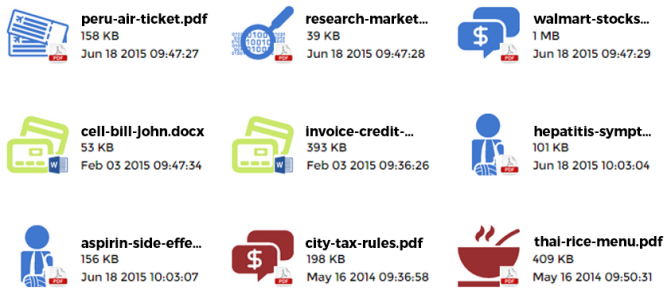

Fig. 4. Topical icons with color encoding to enable nested sorting.

time leads to files being sorted by category (all similar icons appear contiguously), and within each category, according to the time of access. Potentially, additional characteristics like file size or source can be encoded into the icon shape or texture. We performed an initial pilot investigation by *simulating* such a scenario (very old, somewhat old, and new files get different colors of icons; and tasks have artificial time constraints). Initial results in our experimental framework (Sec. V-A) looked promising, with the average navigation time reducing further from 77.39 to 73.18 seconds in the lab pilot. A small screenshot is shown in Fig. 4.

VII. Conclusions and Future Work

We have introduced the novel concept of representing documents by topical icons, and have provided the first approach for automatic assignment of such icons to documents in a folder. Using crowdsourced and lab experiments, we have shown that topical iconification can significantly improve file navigation as measured by four different metrics. Using icon-tag association tests, we have also shown that assigned icons have good agreement on their meanings. Such iconification is likely to be the most beneficial when files in a large folder are neither named systematically nor organized into neat sub-folders. We make our experimental testbed with 30 file finding tasks (with gold answers) in a corpus of 60 documents, available for public use.

Iconification is not intended to be a substitute for traditional file search, with which users have been familiar for decades. Rather, the best effects of iconification will be observed if it is suitably *complemented by content-based search*. Promising directions of future research could be to explore: (a) iconification for hierarchical file sorting, (b) incremental iconification with changes to a folder, and (c) iconification with structural inputs. As a concluding remark, we want to emphasize that while we have shown that topical icons for documents improve file navigation efficiency, iconification is likely to provide *a unique and fresh user experience* for document navigation, especially on mobile devices with limited screen size.

References

[1] D. Elsweiler and I. Ruthven, "Towards task-based personal information management evaluations," in *SIGIR '07*, 2007, pp. 23–30.

[2] S. Fitchett, "Understanding and improving personal file retrieval," Ph.D. dissertation, University of Canterbury, Department of Computer Science and Software Engineering, 2013.

[3] J. P. Lewis, R. Rosenholtz, N. Fong, and U. Neumann, "VisualIDs: Automatic Distinctive Icons for Desktop Interfaces," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 416–423, 2004.

[4] C. Gatsou, A. Politis, and D. Zevgolis, "The importance of mobile interface icons on user interaction." *IJCSA*, vol. 9, no. 3, 2012.

[5] K. Haramundanis, "Why icons cannot stand alone," *ACM SIGDOC Asterisk Journal of Computer Documentation*, vol. 20, no. 2, 1996.

[6] M. D. Byrne, "Using icons to find documents: Simplicity is critical," in *CHI '93*, 1993.

[7] W. Horton, "Designing icons and visual symbols," in *CHI '96*, 1996.

[8] S.-C. Huang and R. G. Bias, "Without context, icons are significantly worse than texts to convey meanings in terms of accuracy and efficiency," in *ASIST '12*, 2012.

[9] V. Setlur and J. D. Mackinlay, "Automatic generation of semantic icon encodings for visualizations," in *CHI '14*, 2014, pp. 541–550.

[10] K. Hemenway, "Psychological issues in the use of icons in command menus," in *CHI '82*, 1982, pp. 20–23.

[11] J. Mendling, J. Recker, and H. A. Reijers, "On the usage of labels and icons in business process modeling," *IJISMD '10*, vol. 1, no. 2, 2010.

[12] C. Gatsou, A. Politis, and D. Zevgolis, "Text vs visual metaphor in mobile interfaces for novice user interaction," *Information Services & Use*, vol. 31, no. 3-4, pp. 271–279, 2011.

[13] C. Koutsourelakis and K. Chorianopoulos, "Unaided icon recognition in mobile phones: a comparative study with young users," *The Design Journal*, vol. 13, no. 3, pp. 313–328, 2010.

[14] M. Kaur, "Designing effective animated icons for children," Ph.D. dissertation, Brunel University, 2011.

[15] A. Worden, N. Walker, K. Bharat, and S. Hudson, "Making computers easier for older adults to use: area cursors and sticky icons," in *CHI '97*, 1997, pp. 266–271.

[16] T. G. Whisenand and H. H. Emurian, "Some effects of angle of approach on icon selection," in *CHI '95*. ACM, 1995, pp. 298–299.

[17] W. Jones, "Personal information management," *Annual review of information science and technology*, vol. 41, no. 1, pp. 453–504, 2007.

[18] D. Barreau and B. A. Nardi, "Finding and Reminding: File Organization from the Desktop," *SIGCHI Bull.*, vol. 27, no. 3, pp. 39–43, 1995.

[19] G. Robertson, M. Czerwinski, K. Larson, D. C. Robbins, D. Thiel, and M. van Dantzich, "Data Mountain: Using Spatial Memory for Document Management," in *UIST '98*, 1998, pp. 153–162.

[20] A. El-Kishky, Y. Song, C. Wang, C. R. Voss, and J. Han, "Scalable topical phrase mining from text corpora," in *VLDB '14*, 2014.

[21] I. Hulpus, C. Hayes, M. Karnstedt, and D. Greene, "Unsupervised graph-based topic labelling using dbpedia," in *WSDM '13*, 2013, pp. 465–474.

[22] D. L. Chaudhari, O. P. Damani, and S. Laxman, "Lexical co-occurrence, statistical significance, and word association," in *EMNLP '11*, 2011.

[23] M. F. Porter, "Snowball: A language for stemming algorithms," 2001.

[24] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *ACL Demonstrations '14*, 2014.

[25] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, 1998.

[26] D. F. Morrison, "Multivariate statistical methods. 3," *New York, NY. Mc*, 1990.

[27] P. Bhatia, S. Iovleff, and G. Govaert, "Blockcluster: An R Package for Model Based Co-Clustering," *HAL-Inria*, 2014.

[28] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Applied statistics*, pp. 100–108, 1979.

[29] J. Han, J. Pei, and M. Kamber, *Data mining: Concepts and techniques*. Elsevier, 2011.

[30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS '13*, 2013.

[31] C. Carpineto, S. Osiński, G. Romano, and D. Weiss, "A survey of web clustering engines," *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009.

[32] U. Scaiella, P. Ferragina, A. Marino, and M. Ciaramita, "Topical clustering of search results," in *WSDM '12*, 2012.

[33] T. Mu, J. Y. Goulermas, I. Korkontzelos, and S. Ananiadou, "Descriptive document clustering via discriminant learning in a co-embedded space of multilevel similarities," *Journal of the Association for Information Science and Technology*, vol. 67, no. 1, pp. 106–133, 2016.