Learning to Cluster Documents into Workspaces Using Large Scale Activity Logs

Weize Kong, Michael Bendersky, Marc Najork, Brandon Vargo, Mike Colagrosso
Google
Mountain View, CA, USA
{weize,bemike,najork,bvargo,mcolagrosso}@google.com

ABSTRACT

Google Drive is widely used for managing personal and work-related documents in the cloud. To help users organize their documents in Google Drive, we develop a new feature to allow users to create a set of working files for ongoing easy access, called *workspace*. A workspace is a cluster of documents, but unlike a typical document cluster, it contains documents that are not only topically coherent, but are also useful in the ongoing user tasks.

To alleviate the burden of creating workspaces manually, we automatically cluster documents into suggested workspaces. We go beyond the textual similarity-based unsupervised clustering paradigm and instead directly learn from users' activity for document clustering. More specifically, we extract co-access signals (i.e., whether a user accessed two documents around the same time) to measure document relatedness. We then use a neural document similarity model that incorporates text, metadata, as well as co-access features. Since human labels are often difficult or expensive to collect, we extract weak labels based on co-access data at large scale for model training. Our offline and online experiments based on Google Drive show that (a) co-access features are very effective for document clustering; (b) our weakly supervised clustering achieves comparable or even better performance compared to the models trained with human labels; and (c) the weakly supervised method leads to better workspace suggestions that the users accept more often in the production system than baseline approaches.

ACM Reference Format:
Weize Kong, Michael Bendersky, Marc Najork, Brandon Vargo, Mike Colagrosso. 2020. Learning to Cluster Documents into Workspaces Using Large Scale Activity Logs. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403291

1 INTRODUCTION

Google Drive is widely used for storing, editing and sharing personal and work-related documents in the cloud. In this paper, we describe *workspaces*[1] – a new feature that helps users to organize their Google Drive documents into sets of working files for ongoing easy access. For instance, a user may create a workspace for each task they are working on, and add related documents to this workspace. Organizing documents using workspaces does not affect file storage location or permissions – it just aggregates documents to help users find them more efficiently. Figure 1 shows an example of the workspace feature.

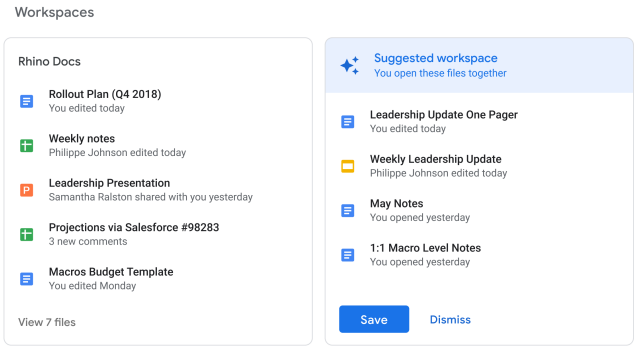

Figure 1: Two workspaces in Google Drive. The right one is automatically suggested to the user.

To alleviate the burden of creating workspaces manually, we develop a machine learning method to automatically cluster documents into workspaces and suggest workspaces to users (see the workspace on the right in Figure 1). This workspace suggestion task can be regarded as a special case of document clustering, where each suggested workspace is a cluster of documents. Different from a typical document cluster, a high-quality workspace should contain coherent documents that are also useful to the user. *Coherent* indicates that the documents inside a workspace should have topics or belong to the same work task. *Useful* indicates that the user is likely to use the documents in the near future.

We discuss a few limitations and challenges of applying existing document clustering methods for workspace suggestion. First, document clustering is typically addressed as an unsupervised problem. Clustering is often based on hand-crafted features and weights (e.g., the TF-IDF representation), or learned from an unlabeled text corpus (e.g., topic modeling). The unsupervised clustering methods could be suboptimal due to two drawbacks: (a) they may not adapt well to different clustering criteria [13], especially when the desired clustering is not solely based on document topics but also on other factors. For workspace suggestion, a user may prefer to cluster documents based on whether they belong to the same task the user is working on, regardless of the document topics; (b) it is not easy to incorporate new features and learn optimal feature weights without hand-tuning for the unsupervised methods.

[1] https://gsuiteupdates.googleblog.com/2019/03/priority-page-drive.html

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08.
https://doi.org/10.1145/3394486.3403291

Supervised clustering methods, on the other hand, are believed to be superior, since their models and the clustering results can be optimized based on available labels. Some prior work has explored semi-supervised [2, 3, 6] and supervised clustering methods [10, 13, 22]. However, these techniques have met less adoption than the unsupervised methods due to the difficulty of collecting clustering labels. As a new feature in Google Drive, prior to its production launch and wide user adoption, *suggested workspaces* suffered from the same lack of clustering labels.

Lastly, the existing methods mostly focus on using document *text* for clustering. Many techniques have been proposed to improve the text representation, from using words and phrases as features with TF-IDF weighting [25], to extracting latent semantics or topics using Latent Semantic Analysis [12] or topic modeling [9, 14]. A few prior works use search logs for search result clustering [30] and webpage clustering [7, 11, 23]. However, these methods are less suitable for private document corpora like Google Drive due to sparsity in their search logs [8, 29].

We address these limitations in this work, and propose to learn from large-scale document usage activity to cluster documents into workspaces. Different from search logs, the document usage activity logs, or *activity logs* for short, record users' document usage activities, such as opening and editing documents. Activity logs are available in much more abundant quantity since they span more than just the search component. They can be collected in desktop file systems, cloud storage systems like Google Drive, or other information systems. While the activity logs could contain several useful signals for document clustering, in this work we focus on one: we extract *co-access* signals, where co-access is defined as two documents are accessed by the same user consecutively within a short time window. For example, a co-access will occur when a user opens two documents at the same time or updates two documents in quick succession. We hypothesize that co-accessing indicates that the two documents are likely to be related, based on the assumption that they may belong to the same task, analogous to the case of query sessions in the search logs [24].

Under the assumption that the human labels are available, we propose a supervised clustering method to incorporate text features, metadata features, as well as features based on the co-access signals. The supervised method learns a model based on these features to predict the similarity between two documents using human labels of "whether two documents should be clustered together." We use a feedforward neural network to build the document similarity model to learn text representations automatically. We then cluster documents based on the predicted document similarity using, for instance, agglomerative hierarchical clustering algorithms like single-linkage clustering [4]. Our experiments show that the co-access features are very effective for document clustering when used alone as well as when combined with other features.

However, the assumption of being able to collect human clustering labels at scale does not always hold. For instance, prior to the public launch of workspaces we faced the cold-start problem and had no way to collect a realistic labeled dataset. Therefore, we also study how to derive weak labels from the activity logs to enable weakly supervised clustering. More specifically, we extract future co-access labels to train the document similarity model. The *future co-access label* is a synthetic label that indicates whether the user will co-access the two documents in the near future.

These labels are certainly not perfect as clustering labels. However, they could be good approximations for human labels, and — very importantly — can be extracted from activity logs at large scale *prior* to the product launch. Our experiments show that this weakly supervised clustering method achieves comparable performance to the supervised clustering method, or even better performance when the number of human labels is limited.

Our contributions can be summarized as follows:

- We present the workspace suggestion task – an interesting document clustering problem that requires intra-cluster documents to be coherent as well as useful to users in their ongoing tasks.
- We go beyond the text-based unsupervised document clustering paradigm to learn from large-scale activity logs for workspace suggestion. To the best of our knowledge, activity logs have not been explored in the literature for document clustering.
- We discover an effective signal in activity logs for document clustering, called co-access. The co-access signals are analogous to the clickthrough signals used for search ranking [5, 17], and can be utilized in two ways. First, *recent and historic co-accesses* can be used as features in a predictive model alongside document text or metadata. Second, *future co-accesses* can be used as weak labels in the absence of clustering labels.
- We adopt a neural document similarity model to utilize the large scale co-access signals. The model supports supervised and weakly supervised document clustering using human labels and future co-access labels respectively.
- We conduct extensive offline and live experiments to demonstrate the effectiveness of the co-access signals as well as our clustering methods. We find our weakly supervised clustering method – which does not require any human labels – can achieve comparable performance to the supervised clustering method, or even better performance when the number of human labels is limited. Lastly, the live experiments also confirm that our weakly supervised method leads to better workspace suggestions that the users accept significantly more often than the other baseline clustering approaches based on document topicality or heuristics.

2 RELATED WORK

Document clustering (or text clustering) has been extensively studied in the literature. Typically, documents are represented as feature vectors [25], using words and phrases extracted from the documents as features. Prior work has proposed to improve document representation using latent topics, such as Latent Semantic Indexing [12], Probabilistic Latent Semantic Analysis [14] and Latent Dirichlet Allocation [9]. Based on the representation, clustering algorithms can then measure document similarity (or distance) to conduct clustering, using cosine or other similarity functions. Two types of clustering algorithms are commonly used. Agglomerative hierarchical clustering algorithms, such as single-linkage and complete-linkage, are often more effective but less efficient due to their quadratic time complexity. Distance-based partitioning clustering algorithms such as k-means, on the other hand, are more efficient but can be less effective. We refer the reader to Aggarwal and Zhai [4] for a comprehensive survey on document clustering.

While clustering is typically addressed as an unsupervised learning problem, prior work has studied semi-supervised/supervised clustering when labels are partially/fully available. The labels can be pointwise, in the form of "which cluster the item belongs to" [6], or pairwise, in the form of "these two items do or do not belong together" [13]. For semi-supervised clustering, prior work modifies clustering algorithms to use labeled data as seeds in k-means [3, 6] or as clustering constraints [2]. For supervised clustering, some methods directly optimize a clustering loss defined on the labels [13, 19]. Others, including ours, learn a model to measure item similarity based on the pairwise labels, and then use the similarity model to predict pair similarity for clustering. For example, Ng and Cardie [22] train a decision tree to predict whether two noun phrases are co-referent and use it for noun phrase coreference/clustering. Cohen and Richman [10] train a classifier to predict whether two entity names are co-referent and use it for entity-name clustering. Our work follows the same idea, but is different in that we propose a weak supervision method that directly learns the similarity model from large-scale activity logs without human labels.

Our work is also related to user logs and implicit feedback. The idea of using signals from user logs as implicit feedback [20] has been extensively explored for many information retrieval problems, such as search ranking and query suggestion. A comprehensive survey can be found here [16]. Some prior work has also used search logs for webpage clustering [7, 11, 23] and search result clustering [30]. For example, Beeferman and Berge [7] proposed an agglomerative clustering algorithm for clustering similar webpages using a click-through bipartite graph extracted from search logs. They estimate similarity between webpages based on their shared neighbor nodes in the bipartite graph. Wang and Zhai [30] clustered search results around the query subtopics learned from search logs. Our approach is different from these prior works in that we explore document usage activities for document clustering in Google Drive, whose activity logs are much more abundant than its search logs.

3 PROBLEM SETTING

3.1 Workspace Suggestion

Workspace suggestion aims to automatically suggest workspaces for a given user at a given request time (see Figure 1). A workspace is a cluster of documents, and a high-quality workspace should contain coherent documents that are also useful to the user for his/her ongoing tasks.

We give an overview of our workspace suggestion process below:

(1) **Document selection**. A user may have access to tens of thousands of documents, or more. For workspace suggestion, we are only interested in the ones that are most useful to the user in the near future. For this purpose, we select up to N documents that were recently accessed by the user as the candidate documents;

(2) **Document clustering**. To cluster the selected documents into workspaces, we apply our proposed clustering method (Section 4) that learns from large-scale activity logs;

(3) **Ranking**. We rank each clustered workspace based on its *utility score*, i.e., how likely the user will use documents in the workspace in the future, estimated based on the user's recent activity. We omit the details here and leave further exploration of the cluster ranking problem as future work, since it is not the focus of this work.

3.2 Document Clustering

In this work, we focus on the document clustering problem above, which is the core part of the workspace suggestion process. More specifically, our document clustering problem is to group a given set of documents $D = \{d_i\}_{i=1}^{N}$ into clusters that are coherent internally while different from each other. In other words, documents within a cluster should be as similar as possible, and documents from different clusters should be as dissimilar as possible. Unlike other document clustering applications, the notion of similarity in workspace suggestion, may not solely depend on topicality, but also depends on whether the documents are all useful in the context of the same ongoing task.

Each document d contains text content, from which we could measure the document textual similarity. In addition to document text, Google Drive also record users' document usage activities in activity logs, such as opening, editing, uploading, and downloading documents. The activity logs can be represented as a list of events, $E = \{e\}$, where each event $e = (t, u, d)$ has a timestamp t, an acting user u, and a target document d, indicating user u used/accessed document d at time t.

Our research problem is to improve the document clustering using the activity logs. It is important to point out that while we evaluate our methods in the context of Google Drive, the problem of suggesting coherent and useful workspaces is a general one. Our weakly supervised clustering methods can apply to many cloud file and document storage scenarios.

3.3 Privacy

Google Drive contains private document corpora and requires special treatment to protect user privacy. For this reason, the data used for workspace suggestions are k-anonymized [27] and are inaccessible to individual engineers. Moreover, for our proposed methods, we limit the use of text content to the document titles, not the full content, with no word sequence information preserved. The titles are also k-anonymized, i.e., only frequent words used by sufficiently many users in the corpus are retained. Some of our baseline methods extract text features from document full content, but these methods only cluster documents on request – the document content and their extracted features are never materialized.

4 METHOD

This section describes our supervised and weakly-supervised document clustering method using activity logs.

4.1 Supervised Clustering

To facilitate developing the supervised clustering method, we first reformulate the clustering problem as a label prediction problem. That is, for each pair of documents in the given document set $d, d' \in D$, we predict a *co-cluster label* $y_{d,d'} \in \{0, 1\}$ indicating whether the two documents should be clustered together or not. The entire set of co-cluster labels can be more formally denoted as $Y_D = \{y_{d,d'} | \{d, d'\} \in \mathcal{P}_D\}$, where $\mathcal{P}_D = \{\{d, d'\} | d, d' \in D \wedge d \neq d'\}$ is the set of all the unordered document pairs for D. It is easy

to see that we can induce clusters from the co-cluster labels Y_D and vice versa. For example, from clusters $\{d_1, d_3\}$ and $\{d_2\}$, we can induce the labels as $y_{d_1,d_3} = 1$ and $y_{d_1,d_2} = y_{d_2,d_3} = 0$, and from these labels we can induce the original two clusters. Note that the labels may not produce strict partitioning clustering, i.e., each document belonging to exactly one cluster. For example, when $y_{d_1,d_2} = 1$ and $y_{d_2,d_3} = 1$, we may have $y_{d_1,d_3} = 0$, and induce two overlapping clusters $\{d_1, d_2\}$ and $\{d_2, d_3\}$.

Based on the formulation above, we do not solve the clustering problem directly; instead, we build a document similarity model $sim(d, d')$ to estimate the probability that two documents are similar to each other and should be grouped together into one cluster, i.e., $P(y_{d,d'} = 1) \equiv sim(d, d')$. Using $sim(d, d')$, we can predict $y_{d,d'}$ for all the document pairs in $\mathcal{P}_D$ by thresholding, and then induce clusters from the predicted labels. If strict partitioning clustering is required, we can apply clustering algorithms such as single-linkage clustering or complete-linkage clustering based on the predicted similarities.

This method converts the clustering problem into a simpler classification problem, i.e., predicting whether two documents are similar to each other or not. The document similarity model $sim(d, d')$ can be learned in a supervised manner using available human labels or large-scale activity logs, as described in the following sections.

4.2 Document Similarity Model

The document similarity model takes a pair of documents $\{d, d'\}$ as input, and outputs their estimated similarity $sim(d, d') \in [0, 1]$. Each document can be represented by its text content, as well as metadata such as document media type (MIME type). In addition to those two types of features, we also investigate how to extract features from activity logs to measure document similarity for clustering purpose, and we call them *activity-based features*. Table 1 lists all the features and labels used in our experiments. We defer the description of the activity-based features and labels to Section 4.4 and 4.5. We describe data anonymization in Section 3.3 above.

Table 1: Features and labels for a documents pair d, d'.

Text features
• $text(d)$: text content for document d. • $text(d')$: text content for document d'.
Metadata features
• $mime(d)$: MIME type of the document d. • $mime(d')$: MIME type of the document d'.
Activity-based features
• $recent\_co\_accesses(d, d')$: number of co-accesses between d, d' in the past 2 weeks. • $historic\_co\_accesses(d, d')$: number of co-accesses between d, d' in the past 4 weeks.
Human and activity-based labels
• $co\_cluster(d, d')$: human labels on whether d, d' should be clustered together in a workspace. • $future\_co\_accesses(d, d')$: number of co-accesses between d, d' in the future week.

To learn representations automatically for text features, we use a feedforward neural network to build our document similarity model. Figure 2 illustrates the neural network architecture.

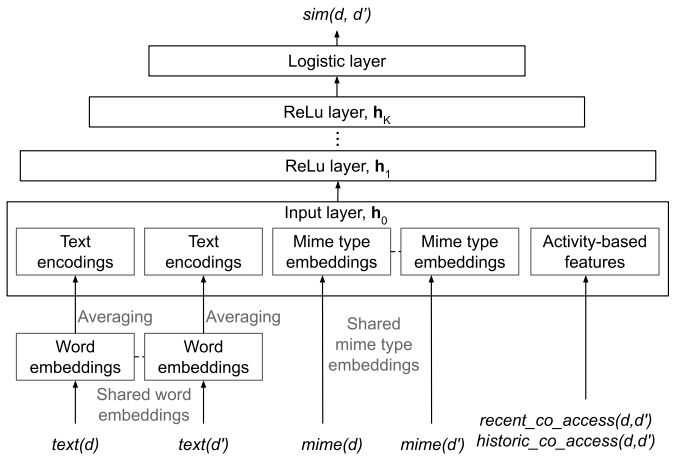

Figure 2: Document similarity model.

In the neutral network, we first build dense representations $\phi_t(text(d))$ for each document text input $text(d)$. ϕ_t could be any complex text encoder such as Transformer and BERT. However, as mentioned in Section 3.3, to protect user privacy, we are only allowed to use k-anonymized words with no sequence information for model training. As a result, models like BERT and recurrent neural networks are not applicable to our workspace suggestion problem. Thus, we choose to encode $text(d)$ by averaging its word embeddings, similar to the Deep Averaging Network [15]:

$$\phi_t(text(d)) = \frac{\sum_{t \in text(d)} emb_t(t)}{|text(d)|}, \tag{1}$$

where $t \in text(d)$ are the words inside the document text, and $emb_t(t)$ is the word embedding. Next, we build dense representations for each document MIME type features $mime(d)$ by simply embedding this categorical feature, i.e., $\phi_t(mime(d)) \equiv emb_m(mime(d))$. Note that the word embeddings and MIME type embeddings are shared between the two documents d, d'.

We then concatenate the text encoding $\phi_t(text(d))$, $\phi_t(text(d'))$ and MIME type embeddings $\phi_m(mime(d))$, $\phi_m(mime(d'))$ with the activity-based features. Let's denote the feature vector after concatenation as $\mathbf{h}_0$. The feature vector is then passed to multiple (K) hidden layers. More specifically, each hidden layer is defined as

$$\mathbf{h_k} = \phi(\mathbf{w}_k^T \mathbf{h}_{k-1} + \mathbf{b}_k), k = 1, 2, ..., K, \tag{2}$$

where $\mathbf{w}_k$ and $\mathbf{b}_k$ denote the weight matrix and the bias vector in the k-th layer and ϕ is an activation function — we use the rectified linear unit (ReLu) in our experiments. The last hidden layer output $\mathbf{h}_K$ is then passed to a sigmoid layer to compute the final similarity,

$$sim(d, d') = \sigma(\mathbf{w}_o^T \mathbf{h}_K + \mathbf{b}_o), \tag{3}$$

where $\mathbf{w}_o$ and $\mathbf{b}_o$ are the weight and bias vector for the sigmoid output layer, and σ is the sigmoid function.

Given a training dataset $\mathcal{D} = \{(d, d', y_{d,d'}) | \{d, d'\} \in \cup_i \mathcal{P}_D^{(i)}\}$, we train the entire network end-to-end by minimizing the weighted

cross-entropy loss defined as follows:

$$-\sum_{(d,d',y_{d,d'})\in\mathcal{D}} y_{d,d'}\log(sim(d,d'))+\lambda(1-y_{d,d'})\log(1-sim(d,d')). \quad (4)$$

where the co-cluster labels $y_{d,d'}$ are induced from workspaces created by users (see the label description in Table 1), $\lambda \in (0,1]$ is a hyper parameter used to down-weight the loss for negative document pairs. We down-weight negative pairs because the number of co-clustered document pairs (i.e., $y_{d,d'} = 1$) is often relatively small in practice. For example, in our data set only around 3% of the document pairs are clustered into the same workspace by the users (see data set description in Section 5.1). This data imbalance issue could lead to understatement of pairwise similarity [13].

4.3 Co-access Signals

Prior work on document clustering mostly relied on using document text signals, however there are two weaknesses to this "ad-hoc" clustering paradigm. First, models based on lexical text features, such as the TF-IDF representation [25], could face the vocabulary mismatch problem when estimating document similarity. Many techniques have been proposed to address this problem including latent semantic analysis [12], topic modeling [9, 14] and more recently word2vec embeddings [21]. However, these techniques offer little help on the second weakness. That is, the text features/representations could be effective in capturing topic similarity, but they may not adapt well to other notions of similarity or document clustering criteria [13]. In the case of workspace suggestion, a user may want to cluster documents according to whether they belong to the same task the user is working on, and each task may contain documents on a wide range of topics.

To address these weaknesses and improve document clustering, we propose to extract signals from users' activity logs (Section 3.2) for measuring document similarity. This idea is analogous to the pioneering of work from the early 2000s that leveraged search logs for improving web search ranking [5, 17]. In this earlier work, clickthrough signals are introduced, and combined with text features for ranking. Instead of *clickthrough*, *search logs* and *web search ranking*, in our work we extract *co-access signals* from *activity logs* for *document clustering*.

Two documents are considered to be **co-accessed** if a user accessed them consecutively within a short time window. Figure 3 shows a sample activity stream with a few co-access pairs using a 2-minute time window. In the figure, $(d1, d2)$ are co-accessed twice, and $(d1, d3)$ are co-accessed once. However, $d3$ and $d4$ are not co-accessed, because the time difference between their accesses is larger than the 2-minute time window. $d2$ and $d3$ are not co-accessed even though they are accessed within 2 minutes. This is because $d2$ and $d3$ are not accessed consecutively — $d1$ is accessed between $d2$ and $d3$. We believe that co-accessing two documents indicates that they are likely to be related. This is based on the assumption that within a small time window a user is likely to be working on the same task, and documents accessed for the same task are related.

The question is then, what is the optimal co-access time window? To answer this question we conduct an analysis on document *inter-access time* (IAT), i.e., the time interval between a user accessing two

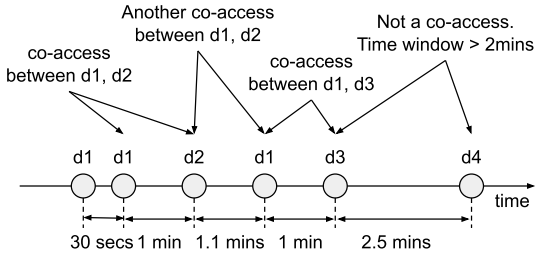

Figure 3: Documents co-accessed by a user in an example activity event stream. We use 2 minutes as the co-access time window threshold in this example.

different documents. For example, for the activity stream in Figure 3, the IATs for $d1/d3$ and $d3/d4$ are 1 and 2.5 minutes respectively. We plot the distribution of IATs over a sample of Google Drive users in Figure 4. It shows that most of the inter-access behaviors happen within a few minutes. More specifically, 58% of the IATs are within one minute. The accumulative percentage grows to 67% and 74% for IATs within two and three minutes, followed by a very long tail. We notice the number of IATs decreases dramatically for each minute after two or three, and therefore hypothesize that most of the within-task document inter-accesses should occur within this time threshold. Thus, we fix the co-access time window to two minutes in the remainder of this paper.

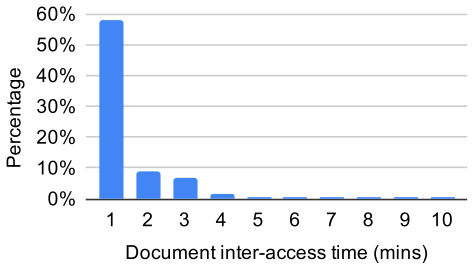

Figure 4: Distribution of document inter-access time, i.e., time interval between a user accessing two documents.

Aside from the co-access signals, other types of activity-based signals (e.g., co-session) could also be potentially useful. However, due to space constraints, in this work we solely focus on co-access, and leave the exploration of other signals to future work.

4.4 Activity-Based Features

Based on the co-access signals, we extract features from activity logs to improve document similarity prediction, and we call them *activity-based features*. More specifically, given a pair of documents $\{d, d'\}$, we extract feature *recent_co_accesses*(d, d'), which is the number of co-accesses in the past 2 weeks before the user requesting workspace suggestions, as well as feature *historic_co_accesses*(d, d'), which is the number of co-accesses in the past 4 weeks before the user requesting workspace suggestions. The two activity-based features are also listed in Table 1.

Activity logs could be potentially noisy, and so are the two activity-based features extracted from the logs. For example, when a user restarts a web browser, the browser may automatically reload all previously opened Google Drive documents, which could result

in a series of co-accesses between these documents that may not necessarily be related or similar. To test whether the two activity-based features are indeed useful in predicting document similarity, we analyze them by using each individual feature to predict the co-cluster labels (i.e., whether or not two documents are clustered in the same workspace by the user) and report the Area Under the ROC Curve (AUC) in Table 2. For this analysis, we randomly sample around 1M document pairs from our training data set $\mathcal{D}_h^{train}$ (see the data collection description in Section 5.1). The AUC results of 0.63 and 0.68 in Table 1 indicate that both of the two activity-based features perform fairly well in distinguishing the co-clustered document pairs ($y_{d,d'} = 1$) from the negative pairs ($y_{d,d'} = 0$) – much better than a random guess. Thus, we further hypothesize that they could enrich the document pair representation for predicting document similarity in addition to the document text and metadata features, and test this hypothesis in Section 6.1.

Table 2: AUC performance on predicting co-cluster labels by each activity-based features.

Feature	AUC
recent_co_accesses(d, d')	0.6360
historic_co_accesses(d, d')	0.6838

4.5 Activity-Based Labels and Weak Supervision

With the proposed activity-based features and the text/metadata features, one could train an effective document similarity model to enable supervised clustering when the co-cluster labels are available, as described in Section 4.1 and Section 4.2. However, one practical issue we encountered is the inability to collect any realistic co-cluster labels at scale before the production launch of the suggested workspaces. This is not unique to our use case – it is often challenging or expensive to collect human clustering labels at large scale, which is an important reason for the limited adoption of supervised clustering methods in practical applications. Therefore, in this section we investigate the following research question: *Is it possible to extract weak labels from activity logs and use them for effective document clustering without the need for human labels?*

Figure 5 illustrates our process of extracting activity-based labels. We first split a user's activity event stream into past and future activity segments, at the time when the user requests workspace suggestion. The past activity segment contains events before the workspace suggestion request. We use this segment for extracting activity-based features, such as the number of **recent co-accesses** (see Table 1). The future activity segment contains events after the workspace suggestion request. We propose to derive activity-based labels from this segment. More specifically, we extract the number of **future co-accesses** (co-accesses in the subsequent days after the workspace suggestion request) as labels for model training.

The notion of *recent co-access features* and *future co-access labels* is analogous to notion of *clickthrough features* [5] and *clickthrough labels* [17] used for search ranking – clickthrough information from the past search logs can be extracted as features, while the future clickthrough information can be used as labels for model training or evaluation.

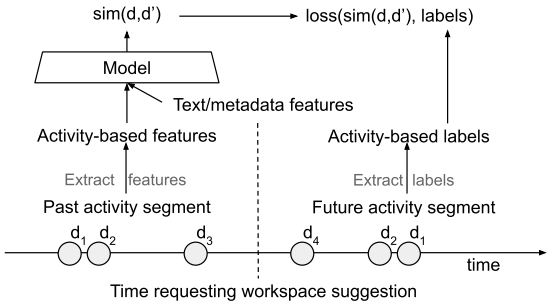

Figure 5: Past and future activity segments used for extracting activity-based features and labels respectively.

We hypothesize that future co-access could be a good weak label for training the document similarity model. We already show that co-access could indicate document relatedness and similarity (see Table 2), and will further test this hypothesis in Section 6.2. Meanwhile, we acknowledge that the future co-access labels are not perfect, but they are cheaper and easier to collect at large scale from activity logs. This is similar to implicit user feedback [20] used in other related problems. For example, for search ranking, user clicks are often collected as labels for training ranking models [17], even though clicks are known to be biased [18, 29]. In addition, by optimizing clustering models to group future co-accessed items in the same cluster, we can save users time from switching context and searching for the right next document they need in the future.

Therefore, we propose to use the future co-accesses as weak labels to train the document similarity model, as a weakly supervised clustering method. More specifically, we collect another training data set $\mathcal{D}_a = \{(d, d', y_{d,d'}) | \{d, d'\} \in \cup_i \mathcal{P}_D^{(i)}\}$, where the labels $y_{d,d'}$ are the binary future co-access labels, or more formally,

$$y_{d,d'} = \begin{cases} 1, & \textit{future\_co\_accesses}(d, d') > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Instead of binarizing the future co-access labels, we also tried to use the raw future co-access counts to train the document similarity model as a regression model, as well as to weight training examples in the cross-entropy loss (Equation 4) using the raw counts. However, we find similar performance across these model variations, and therefore, for simplicity, use the binary future co-access label definition in the remainder of this paper.

5 EXPERIMENTAL SETUP

5.1 Data Collection

We collect two Google Drive activity datasets for our work on the *suggested workspaces* feature. Before the production launch, we collect the first dataset, future co-access label data $\mathcal{D}_a$, which contains the future co-access labels. This dataset is used to train the first production model when human labels are not available. Only *after* the production launch, we are able to collect the second dataset, the human labeled data $\mathcal{D}_h$, which contains the co-cluster labels from user created workspaces. Note that we use subscript "a", which stands for *a*ctivity-based labels, and "h", which stands for *h*uman labels, to differentiate the two datasets.

To collect $\mathcal{D}_h$, we randomly sample three large, non-overlapping subsets of users who actively use Google Drive workspaces for training, validation and testing. For each user, we synthesize the datasets using the following steps:

(1) Sample 20 hypothetical workspace suggestion request timestamps from a uniform distribution within a 2-week time window. We also try to sample requests based on users' Google Drive homepage visits, but find the two methods yield very close results.

(2) For each sampled request timestamp, collect the past and future activity segments (see Figure 5). We collect a maximum of several thousand events over several weeks for each segment. We filter out cases where the past or future segments do not contain sufficient events.

(3) From each past activity segment, we collect documents $D = \{d\}$ for clustering by selecting N most recently accessed documents.

(4) From the document set D, we construct the document pair set $\mathcal{P}_D$ (Section 4.1). The size of the document pair set could be very large (up to $N \cdot (N-1)/2$ pairs in theory). To improve efficiency as well as reducing noise, we filter the document pairs by requiring them to be co-accessed at least once in the past two weeks before the request time, i.e., $recent\_co\_accesses(d, d') \geq 1$. This reduces the pair set size dramatically to ~ 20 pairs per request on average.

(5) For each document pair in $\mathcal{P}_D$, we extract all the features and labels listed in Table 1. The co-cluster human labels are defined by whether the two documents are added to the same workspace by the user within a week following the request time. The data is anonymized as described in Section 3.3.

The future co-access label data $\mathcal{D}_a$ is collected in the same way as described above for $\mathcal{D}_h$, except for three differences. First, we sample from a large population of Drive users, not limited to the ones who actively use Drive workspaces. Because of this, we can collect more data for $\mathcal{D}_a$. Second, we do not extract the co-cluster labels for $\mathcal{D}_a$, since the suggested workspaces were not available at the time of data collection. Lastly, for $\mathcal{D}_a$ we collect only a single data set for training, because we further tune and test models on $\mathcal{D}_h$'s validation and testing data set.

Table 3 shows some statistics for the collected data sets, which reports the number of requests (*#requests*), and the number of document pairs (*#pairs*). Note that $\mathcal{D}_a^{train}$ is much larger than $\mathcal{D}_h^{train}$, because $\mathcal{D}_a$ can be collected from all eligible Drive users, while $\mathcal{D}_h$ can only be collected from users who actively use and create Drive workspaces. The table also reports the percentages of positive document pairs (positive rates) according to the future co-access and co-cluster labels, which are quite consistent across different data sets. We also notice the positive rates are low, therefore we use weighted cross entropy loss (Equation 4) to down-weight the loss of the negative examples.

Table 3: Statistics for $\mathcal{D}_h$ and $\mathcal{D}_a$ date sets. The *train, vali, test* superscript indicates the data set is for training, validation or testing respectively. "#requests" and "#pairs" report the total number of requests and document pairs respectively. "co-access%" and "co-cluster%" reports the percentage of positive pairs according to the future co-access labels and the co-cluster labels respectively.

Data set	#requests	#pairs	co-access%	co-cluster%
$\mathcal{D}_h^{train}$	434K	15.6M	8.27%	2.72%
$\mathcal{D}_h^{vali}$	18.0K	385K	8.89%	2.88%
$\mathcal{D}_h^{test}$	16.9K	396K	8.96%	3.00%
$\mathcal{D}_a^{train}$	19.9M	405M	8.96%	n/a

5.2 Evaluation

We evaluate the document clustering performance according to co-cluster labels as well as the future co-access labels, and consider the co-cluster labels as the the ground-truth. We measure the clustering performance using the pair-counting F1 measure [1]. It treats document clustering as classification on whether each pair of documents are in the same cluster, and then combines the pair precision and recall using the F1 measure [28]. The F1 measure requires selecting a classification threshold; we tune the threshold in $[0.01, 0.02, ..., 0.99]$ to maximize F1 on the validation data set. In addition to the F1 measure, we also measure the area under the ROC curve (AUC) for the same document pair classification problem. The AUC tells us how well a model could distinguish the positive document pairs from the negative ones. Overall, we use the following four metrics:

- $F1_h$: F1 measure based on the co-cluster human labels.
- $F1_a$: F1 measure based on the future co-access labels.
- AUC_h: AUC based on the co-cluster human labels.
- AUC_a: AUC based on the future co-access labels.

Both the F1 measure and AUC are computed at the request level. That is, we compute one measure score for each document pair set $\mathcal{P}_D$, and then report the average measure scores. To compute a valid AUC, the document pair set $\mathcal{P}_D$ needs to contain at least one positive and one negative pair. Therefore, we filter out all invalid cases in the validation and testing data sets. Note that information-theoretic clustering metrics such as purity or mutual information are less applicable here, because our labeled dataset contains documents that do not belong to any cluster (workspace) and they would need to be treated as singleton clusters for these metrics.

We train our models using $\mathcal{D}_h^{train}$ or $\mathcal{D}_a^{train}$, tune hyperparameters on the validation data $\mathcal{D}_h^{vali}$, and report results on the test data $\mathcal{D}_h^{test}$. We perform statistical significance tests using paired t-test with 0.01 as the p-value threshold.

5.3 Implementation Details

We implement the document similarity model in Tensorflow, using AdaGrad as the optimizer. We cap the word vocabulary size at 500k by mapping less frequent words to a few out-of-vocabulary embeddings. We use pilot studies to select the embedding dimension size and batch size. We tune the hidden layer size, λ (Equation 4), learning rate, and classification threshold (Section 5.2) by grid search. We discuss implementation details for other baseline models in Section 6.3.

6 EXPERIMENTAL RESULTS

6.1 Activity-Based Features

To study the effectiveness of activity-based features, we train the document similarity model with different feature sets using co-cluster labels on $\mathcal{D}_h^{train}$. We report their test results on $\mathcal{D}_h^{test}$ in

Table 4, in which f_T, f_M, f_A stand for the text, metadata and activity-based feature sets respectively. The performance differences mentioned in the comparison below are all statistically significant (p-value < 0.01).

Table 4: Performance with/without activity-based features. Models are trained on co-cluster labels. f_T, f_M and f_A stand for text, metadata and activity-based features respectively.

Feature set	AUC_h	$F1_h$	AUC_a	$F1_a$
f_A	0.7169	0.3324	0.6986	0.3388
f_M	0.6285	0.3110	0.5810	0.2598
$f_M + f_A$	0.7353	0.3333	0.7027	0.2997
f_T	0.6590	0.2902	0.5978	0.2740
$f_T + f_A$	0.7360	0.3188	0.7045	0.2906
$f_M + f_T$	0.6883	0.3018	0.6100	0.2573
$f_M + f_T + f_A$	0.7495	0.3459	0.7099	0.3242

First, activity-based features are the most effective features when used alone. By comparing each individual feature set, f_M, f_T and f_A, we find f_A outperforms f_M, f_T by a large margin across all the metrics (+7% to +30%). f_T outperforms f_M on all metrics except for $F1_h$. **Second**, incorporating activity-based features with baseline feature sets effectively improves the performance. Comparing the baseline feature sets f_M, f_T, $f_M + f_T$ with and without f_A, we find adding f_A into the baseline feature sets improves all metrics by a large margin (+6% to +26%). These two observations clearly show the effectiveness of activity-based features.

Third, combining all the features $f_M + f_T + f_A$ achieves the best performance across all the metrics. Therefore, we only report results on all features for the other experiments below due to space limitations. **Finally**, we find the evaluation results are quite consistent between the future co-access metrics (AUC_a, $F1_a$) and co-cluster metrics (AUC_h, $F1_h$). This indicates the future co-access label could be a good approximation for the co-cluster human label. This leads to our next research question: can we train an effective model using future co-access labels as weak supervision?

6.2 Weak Supervision

To study the effectiveness of the proposed weak supervision method, we train a document similarity model with future co-access labels on $\mathcal{D}_a^{train}$ (weak supervision method) and another model with co-cluster human labels on $\mathcal{D}_h^{train}$ (supervision method). We compare their test results on $\mathcal{D}_h^{test}$ in Table 5, denoted as $\mathcal{M}_a$ and $\mathcal{M}_{100\%h}$ respectively. **First**, the weak supervision method achieves comparable performance as the supervision method. $\mathcal{M}_a$ and $\mathcal{M}_{100\%h}$ have very close performance in terms of AUC_h and $F1_h$, and the differences are not statistically significant. The p-value is 0.06 when comparing AUC_h and 0.14 when comparing $F1_h$. This indicates that the future co-access label is a good approximation for the co-cluster human label, and our model can effectively learn from the large-scale dataset of weak labels without any human labels. **Second**, the weak supervision method outperforms the supervision method by a large margin on future co-access metrics, AUC_a and $F1_a$. This is expected since $\mathcal{M}_a$ is trained by optimizing for future co-access prediction. The two observations combined indicate training with future co-access labels could optimize both future co-access prediction as well as co-cluster prediction.

Table 5: Performance for models trained with future co-access and co-cluster labels. $\mathcal{M}_a$ stands for the model trained with future co-access labels. $\mathcal{M}_{x\%h}$ stands for the model trained with co-cluster human labels using x% of $\mathcal{D}_h$ training data. "Data size" reports the number of document pairs in the used training data set. +/− indicates $\mathcal{M}_a$ is statistically better/worse than the marked baselines (p-value < 0.01).

Model	Data size	AUC_h	$F1_h$	AUC_a	$F1_a$
$\mathcal{M}_{1\%h}$	0.16M	0.7016^+	0.2878^+	0.6505^+	0.2343^+
$\mathcal{M}_{9\%h}$	1.41M	0.7298^+	0.3210^+	0.6862^+	0.2977^+
$\mathcal{M}_{10\%h}$	1.56M	0.7418	0.3391^+	0.6927^+	0.2978^+
$\mathcal{M}_{50\%h}$	7.82M	0.7429	0.3423^+	0.7030^+	0.3202^+
$\mathcal{M}_{60\%h}$	9.38M	0.7431	0.3343^+	0.7018^+	0.3147^+
$\mathcal{M}_{70\%h}$	10.9M	0.7422	0.3474	0.7048^+	0.3112^+
$\mathcal{M}_{100\%h}$	15.6M	0.7495	0.3459	0.7099^+	0.3242^+
$\mathcal{M}_a$	405M	0.7458	0.3498	0.7291	0.3457

Co-cluster labels are not available at the time of the initial product launch, and are often much more difficult to collect at large scale than future co-access labels. So next, we study the effectiveness of the weak supervision method by comparing its performance with models trained on limited co-cluster human labels. For this experiment, we randomly subsample different amounts of D_h^{train} to train the document similarity model, and compare their test results with $\mathcal{M}_a$ in Table 5. We denote the model trained with x% of D_h^{train} as $\mathcal{M}_{x\%h}$, and report the data size in terms of the number of document pairs. We find that when the amount of co-cluster labels is limited, training with future co-access labels achieves better performance. More specifically, $\mathcal{M}_a$ statistically significantly (p-value < 0.01) outperforms $\mathcal{M}_{x\%h}$ on AUC_h when $x\% \leq 9\%$ (1.41M document pairs), and on $F1_h$ when $x\% \leq 60\%$ (9.38M document pairs).

6.3 Live Experiment

To conclude the empirical study, we verify the effectiveness of our proposed weak supervision method by comparing it with several baseline clustering methods that do not require supervised labels, and were included in the original suggested workspaces launch:

Weak Supervision Our proposed weak supervision method. This method predicts $sim(d, d')$ using the model trained with future co-access labels, and clusters documents using the single-linkage clustering algorithm based on the predicted similarities.

Topicality A popular unsupervised clustering method that represents a document's topic using a feature vector and then clusters the documents using complete-linkage clustering based on the vectors' cosine similarity. We extract Knowledge Graph [26] entities and salient terms from document full text content as features, using an internal tool. The tool also produces relevance scores for each extracted term, which are used as the feature weights.

Hyperlink Same as Topicality, except that a document's topic is represented using its hyperlinks, and document similarity is estimated using the Jaccard similarity coefficient.

Calendar A heuristic method that clusters documents that are all attached to the same calendar event.

Collaborator A heuristic method that clusters documents shared from the user's close collaborators, by the collaborators.

Favorites A heuristic method that clusters the user's most frequently used documents. Favorite documents are selected based on

document interaction scores, and the score is estimated based on the number of activity events on the document weighted by the event recency.

We use pilot studies to select the above clustering algorithms (single-linkage or complete-linkage) and other hyper parameters (e.g., clustering thresholds). We rank and suggest the clustered workspaces as described in Section 3.1. As already mentioned in Section 3.3, to protect user privacy, the baseline methods process and cluster documents at request time. The document content and their extracted features are never materialized.

We compare their performance using an online experiment, and measure their acceptance rate over 28 days. *Acceptance rate* is the ratio between the number of accepted workspace suggestions and the number of displayed workspace suggestions. In Table 6, we report the relative improvement of the **Weak Supervision** method over the other baselines. We find that **Weak Supervision** outperforms the other methods by a large margin +[30%, 49%]. This shows the benefit of learning from large-scale activity logs, and testifies to the effectiveness of the proposed weak supervision method.

It is interesting to note that **Hyperlink**, **Favorites** and **Calendar** methods all outperform **Topicality** (a standard text-based unsupervised clustering method). This indicates that the document clustering criterion for workspace suggestion may not only depend on topicality, but also on other factors (e.g., the tasks that the user is working on).

Table 6: Relative acceptance rate improvements of Weak Supervision over the other clustering baselines.

Baseline	Relative Improvement
Collaborator	+49.1%
Topicality	+43.1%
Calendar	+40.9%
Favorites	+31.3%
Hyperlink	+29.9%

7 CONCLUSIONS

In this work, we develop workspace suggestion to help users organize their documents in Google Drive. A workspace is a cluster of coherent documents that are also useful for a user's ongoing task. To cluster documents into workspaces, we go beyond the text-based unsupervised clustering paradigm and propose to learn directly from large scale activity logs. More specifically, we first hypothesize that *document co-access* can be used to measure document relatedness/similarity. We then develop a supervised clustering method based on a neural document similarity model, which incorporates co-access features with sparse text features and categorical metadata features. Third, we propose to use *future co-access* – which can be automatically extracted from large scale activity logs at little cost – to enable weak supervision without any human labels.

Our offline evaluation using Google Drive logs shows that the co-access features are very effective for document clustering when used alone and provide large performance gain when combined with document text and metadata features. Our weak supervision method based on the future co-access labels can achieve comparable performance to the supervised clustering model, and even better performance when the number of human labels is limited. Our online experiment shows that the weakly supervised method leads to better workspace suggestions that the users accept 30% to 49% more often compared to other unsupervised clustering baselines.

REFERENCES

[1] Elke Achtert, Sascha Goldhofer, Hans-Peter Kriegel, Erich Schubert, and Arthur Zimek. 2012. Evaluation of Clusterings–Metrics and Visual Support. In *Proc. of ICDE*. 1285–1288.

[2] Charu C Aggarwal, Stephen C Gates, and Philip S Yu. 1999. On the merits of building categorization systems by supervised clustering. In *Proc. of KDD*. 352–356.

[3] Charu C Aggarwal, Stephen C Gates, and Philip S Yu. 2004. On using partial supervision for text categorization. *IEEE TKDE* 16, 2 (2004), 245–255.

[4] Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text clustering algorithms. In *Mining Text Data*. Springer, 77–128.

[5] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proc. of SIGIR*. 19–26.

[6] Sugato Basu, Arindam Banerjee, and Raymond Mooney. 2002. Semi-supervised clustering by seeding. In *Proc. of ICML*. 27–34.

[7] Doug Beeferman and Adam Berger. 2000. Agglomerative clustering of a search engine query log. In *Proc. of KDD*. 407–416.

[8] Michael Bendersky, Xuanhui Wang, Donald Metzler, and Marc Najork. 2017. Learning from user interactions in personal search via attribute parameterization. In *Proc. of WSDM*. 791–799.

[9] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3, Jan (2003), 993–1022.

[10] William W Cohen and Jacob Richman. 2002. Learning to match and cluster large high-dimensional data sets for data integration. In *Proc. of KDD*. 475–480.

[11] Zhuyun Dai, Chenyan Xiong, and Jamie Callan. 2016. Query-biased partitioning for selective search. In *Proc. of CIKM*. 1119–1128.

[12] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41, 6 (1990), 391–407.

[13] Thomas Finley and Thorsten Joachims. 2005. Supervised clustering with support vector machines. In *Proc. of ICML*. 217–224.

[14] Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proc. of SIGIR*. 50–57.

[15] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proc. of ACL*, Vol. 1. 1681–1691.

[16] Daxin Jiang, Jian Pei, and Hang Li. 2013. Mining search and browse logs for web search: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)* 4, 4 (2013), 57.

[17] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proc. of KDD*. 133–142.

[18] Thorsten Joachims, Laura A Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proc. of SIGIR*. 154–161.

[19] Toshihiro Kamishima and Fumio Motoyoshi. 2003. Learning from cluster examples. *Machine Learning* 53, 3 (2003), 199–233.

[20] Diane Kelly and Jaime Teevan. 2003. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum* 37, 2 (2003), 18–28.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[22] Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proc. of ACL*. 104–111.

[23] Barbara Poblete and Ricardo Baeza-Yates. 2008. Query-sets: using implicit feedback and query patterns to organize web documents. In *Proc. of WWW*. 41–50.

[24] Filip Radlinski and Thorsten Joachims. 2005. Query chains: learning to rank from implicit feedback. In *Proc. of KDD*. 239–248.

[25] Gerard Salton and Michael J McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc.

[26] Amit Singhal. 2012. Introducing the knowledge graph: things, not strings. https://blog.google/products/search/introducing-knowledge-graph-things-not/.

[27] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *Intl. Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 557–570.

[28] C.J. van Rijsbergen. 1979. *Information Retrieval*. Butterworth.

[29] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proc. of SIGIR*. 115–124.

[30] Xuanhui Wang and ChengXiang Zhai. 2007. Learn from web search logs to organize search results. In *Proc. of SIGIR*. 87–94.