Hierarchical Label Propagation and Discovery for Machine Generated Email

James B. Wendt[†] Michael Bendersky[†] Lluis Garcia-Pueyo[†] Vanja Josifovski[‡*]
Balint Miklos[†] Ivo Krka[†] Amitabh Saikia[†] Jie Yang[†] Marc-Allen Cartright[†] Sujith Ravi[†]
[†]Google, Mountain View, CA, USA [‡]Pinterest, San Francisco, CA, USA
[†]{jwendt, bemike, lgpueyo, bal, krka, amitabh, yangjie, mcartright, sravi}@google.com
[‡]vanja@pinterest.com

ABSTRACT

Machine-generated documents such as email or dynamic web pages are single instantiations of a pre-defined structural template. As such, they can be viewed as a hierarchy of template and document specific content. This hierarchical template representation has several important advantages for document clustering and classification. First, templates capture common topics among the documents, while filtering out the potentially noisy variabilities such as personal information. Second, template representations scale far better than document representations since a single template captures numerous documents. Finally, since templates group together structurally similar documents, they can propagate properties between all the documents that match the template. In this paper, we use these advantages for document classification by formulating an efficient and effective hierarchical label propagation and discovery algorithm. The labels are propagated first over a template graph (constructed based on either term-based or topic-based similarities), and then to the matching documents. We evaluate the performance of the proposed algorithm using a large donated email corpus and show that the resulting template graph is significantly more compact than the corresponding document graph and the hierarchical label propagation is both efficient and effective in increasing the coverage of the baseline document classification algorithm. We demonstrate that the template label propagation achieves more than 91% precision and 93% recall, while increasing the label coverage by more than 11%.

Categories and Subject Descriptors

H.4.3 [**Information Systems Applications**]: Communications Applications — *Electronic mail*

*Work done while at Google.

WSDM 2016 February 22-25, 2016, San Francisco, CA, USA

ACM ISBN 978-1-4503-3716-8/16/02.

DOI: http://dx.doi.org/10.1145/2835776.2835780

Keywords

Machine-generated email, structural template, hierarchical label propagation.

1. INTRODUCTION

Email is one of the most popular and frequently used web applications. Even with the more recent advent and widespread adoption of social networks, email continues to be the most pervasive form of online communication. Furthermore, it remains inextricably tied to user online identity. For example, the vast majority of account-based online services require an email address for validation purposes. This practice has effectively designated the inbox as a central and personal hub for the majority of online user activity, and will continue to act so for the foreseeable future.

According to the Radicati group [23], there are just over 2.5 billion email users in the world. This number is predicted to increase by 12% in four years time. The average consumer currently sends and receives an average of about 61 emails per day while business users send and receive an average of about 121. All together, about 196 billion emails are sent and received per day and is predicted to increase to 228 billion by 2018. This proliferation of email has led to the notion of *email overload*, in which users become overwhelmed when they can no longer organize, manage, and process their emails as quickly as they receive them [10].

The most widely adopted tools employed to manage this influx of emails—filters and labels—have not changed in years. Typical filters allow the user to sort mail based on a combination of header information and search terms. An example filter might automatically move all incoming emails received from "auto-confirm@amazon.com" to a *Shopping* folder. While filters can be powerful tools for managing emails from specific senders or sender domains, this is also largely the extent to which they are practical. For example, if a user wishes to label all incoming order confirmation emails to a *Shopping* folder, the user would need to manually generate filters for each potential sender. Given the vast number of e-commerce sites accessible on the web today, creating filters for each potential sender would require an insurmountable effort of creating relevant filters.

It is also important to note that email is no longer used solely for communication. It is used for information archiving and task management as well [11]. Automated machine-learned categorization of email can be an exceptionally powerful tool to improve user efficiency in inbox management by

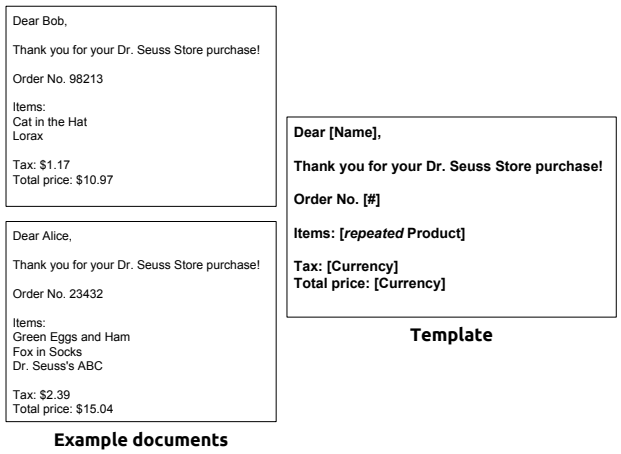


Figure 1: Fictional example of two documents that were generated from the same structural template.

categorizing the email by logical folders or labels and reducing the inbox overload.

Categorization also produces an additional layer of semantic information that can be used to benefit the user experience. For example, consider an email that is automatically categorized as a flight itinerary. In this case, it would be beneficial to the user that the email service highlights this document and brings it to the forefront or alert the user of its importance near the relevant time of departure. On the other hand, a document labeled as a sales promotion might be skipped past the inbox altogether and archived for potential lookup later. Furthermore, email categorization can be used to help extract data from the email that is most relevant for a particular category.

In this paper, we present a novel technique for the categorization of machine-generated emails. We use *structural template* representations to create a scalable, robust, and accurate classifier that increases labeling coverage over a high-precision baseline classifier that does not employ structural template information.

We specifically target machine-generated emails for the following reasons: (a) they constitute a majority percentage of all emails [2], (b) machine-generated emails are very often instantiations of pre-defined structural templates. Figure 1 demonstrates a simple example of a structural template from a fictional *Dr. Seuss* store that is used to generate multiple documents by instantiating the template with the user-specific information (user name, purchased items, prices and order number).

Structural templates provide two important advantages for clustering and categorization. The first is that they scale better than emails since one template represents numerous emails. The second is that since templates represent structurally similar documents, they effectively capture common topics between documents while filtering out potential noise, such as personal information.

We use these advantages to formulate and explore three techniques for efficient and effective categorization. The first technique labels emails based on the majority label among the underlying emails that their corresponding template was inferred from. This simple classifier is a first step towards utilizing templates for email classification and does not take into consideration similarities or relationships between different templates.

The second technique utilizes inter-template similarities to build a centroid-based classifier. Templates are represented using either term-frequencies extracted from their fixed text or as a distribution of topics as inferred by Latent Dirichlet Allocation [8]. Fixed centroids are calculated for each label and are comprised of only those templates whose underlying email samples all correspond to a single label. Templates consisting of two or more labels in their underlying emails are labeled according to their closest centroid.

However, single centroids are often too coarse to adequately classify templates, which belong to the same label but may be vastly different from one another. Thus, we present a third technique which naturally follows the centroid approach, classifying templates using graph-based label propagation [24, 27]. Again, templates are represented using either bag-of-words or topic distributions, and distances or edges between templates are measured in terms of the similarity between their representations. Seed nodes are assigned from the same set used to calculate the centroids in the previous technique. These nodes are the first to emit their labels to their neighbors, then those neighbors emit them to their neighbors, and so on until the entire graph is labeled.

Once the label propagation process over the template graph is complete we propagate the label distributions of the templates to their underlying emails. We refer to this process as *hierarchical label propagation*.

Note that while the template-based categorization techniques discussed in this paper are focused on email data, they are general enough to be applied to any machine-generated documents including web pages, call data records, or government forms, among many others. To the best of knowledge, this is the first publicly available study on leveraging structural templates for the purpose of improving categorization of machine-generated documents in general, and specifically, machine-generated emails.

2. MODEL

In this section, we present the formal description of the hierarchical label propagation algorithm over a given machine-generated corpus. The algorithm operates in three main stages.

First, documents in the corpus are grouped by their underlying structural templates. Second, we construct a partially labeled graph over the template representations. Finally, we run a hierarchical label propagation algorithm over the template graph, which propagates the derived template labels to the individual document level.

The remainder of this section is organized as follows. In Section 2.1 we describe template representation methods for a machine-generated email corpus. In Section 2.2 we describe how these template representations can be organized as a graph. Finally, in Section 2.3, we describe several methods for label assignment and propagation over this template graph.

While in this section we focus on templatization and label propagation over an email corpus, the presented algorithms are general enough to be applied to any templatizable machine-generated document collection. Therefore, henceforth, we use the terms *document* and *email* interchangeably.

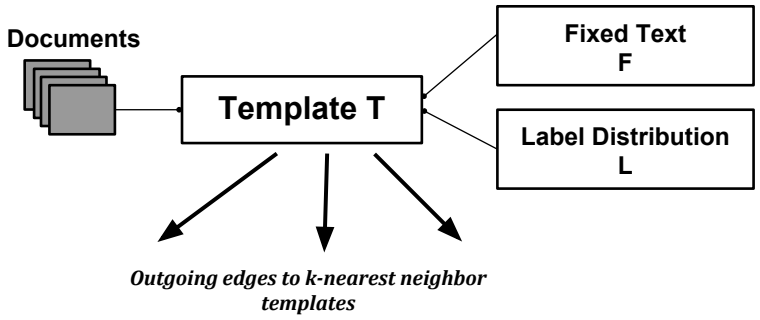


Figure 2: Template representation for graph construction.

2.1 Template Representations

There are multiple ways to derive structural templates from a stream of machine-generated documents. In smaller, specialized collections, the relation between the generating template and the document may be given a priori. For larger, web scale collections, techniques like locality sensitive hashing of document header text have been recently proposed [16].

There has been some previous work on template representation of machine-generated documents for email collections as well. Specifically relevant to this paper is work by Ailon et al. [2]. We use the approach proposed in their work, and map each incoming email document to a *template* that is uniquely identified by a

$$<\textit{sender, subject-regexp}>$$

tuple. Subject regular expressions are obtained by mining frequent patterns and removing unique identifiers and personalized information, like names, from email subjects, as described by Ailon et al. [2].

In contrast to Ailon et al. [2], our approach goes beyond associating an email with a subject regular expression. For the purpose of template representation, we are interested in deriving (a) a textual representation of a template based on the content of the documents matching the template, and (b) a label distribution of the template.

This dual representation is then used to construct a template graph. Figure 2 summarizes this representation process, which is formally described next.

2.1.1 Textual Template Representation

Formally, we define template $\mathbf{T}$ as a set of documents $\mathbf{D^T} = \{D_1, \ldots, D_n\}$ matching a template identifier, as expressed by the <*sender, subject-regexp*> tuple.

To derive a textual representation of a template, we tokenize the set of documents $\mathbf{D^T}$, and derive a set of unique terms per template. Generally, this corresponds to a bag-of-words textual representation of the template.

Given a template term x, we define its *support* S_x as a number of documents in $\mathbf{D^T}$ that contain the term, or formally:

$$S_x^{\mathbf{T}} = |\{D | D \in \mathbf{D^T} \wedge x \in D\}| \quad (1)$$

We define the template *fixed text* as a set of terms for which the support is greater than some (large) fraction of the number of template documents, or formally:

$$\mathbf{F^T} = \{x | \frac{S_x^{\mathbf{T}}}{|\mathbf{D^T}|} \geq \tau\}, \quad (2)$$

where $0 < \tau < 1$ is set to a large fraction to remove personal information from the resulting template fixed text representation.

We then use the template fixed text representation $\mathbf{F^T}$, as a representation of a single node in a template graph. Note that this template representation addresses several important issues.

First, we only keep the fixed text of the template underlying any of the documents in the set $\mathbf{D^T}$. We remove noise and variabilities that are a result of introducing personal information into the documents. In this way, the template graph represents the true aspects of the underlying collection, and not those of its particular instantiations.

Second, note that Equation 2 ignores the actual number of documents used to construct the template. Therefore, each template is given the same weight, with no regard to the number of documents that were used to construct the template. This is beneficial for document collections that follow power law distributions. For instance, for email collections, document representation would be dominated by emails from more popular email senders, while template representation avoids this problem.

Finally, the resulting representation is significantly more compact than document representation, since a single template may cover hundreds or thousands of documents.

2.1.2 Topic Models for Template Representation

In addition to using single terms for template representation, we experiment with modeling templates using topic models [8]. We run a standard Latent Dirichlet Allocation topic modeling over the fixed text of the templates (see Equation 2). We then use the resulting topics along with their weights as template representations instead of the bag-of-words template representation.

2.1.3 Label Distribution

Each document associated with a template may be labeled by some classifier. Given some labeling scheme with m distinct labels, we associate each template with a label distribution:

$$\mathbf{L^T} = \{p(L_1|\mathbf{T}), \ldots, p(L_m|\mathbf{T})\}, \quad (3)$$

where the probability of a label given a template is simply a fraction of documents in the set $\mathbf{D^T}$ containing the label. As

Method	Similarity Metric	Edge Type
Cosine Similarity	$\frac{\sum_{x \in \mathbf{F}^{\mathbf{T}_i}, \mathbf{F}^{\mathbf{T}_j}} w(x,\mathbf{T}_i) w(x,\mathbf{T}_j)}{\sqrt{\sum_{x \in \mathbf{F}^{\mathbf{T}_i}} w(x,\mathbf{T}_i)^2} \sqrt{\sum_{x \in \mathbf{F}^{\mathbf{T}_j}} w(x,\mathbf{T}_j)^2}}$	Undirected
KL Divergence	$\exp(-\sum_{x \in \mathbf{F}^{\mathbf{T}_i} \cap \mathbf{F}^{\mathbf{T}_j}} p(x\|\mathbf{T}_i) \log \frac{p(x\|\mathbf{T}_i)}{\widetilde{p}(x\|\mathbf{T}_j)})$	Directed

Table 1: Template graph construction methods. We use either cosine similarity or KL divergence to define a weighted edge between a pair of templates $(\mathbf{T}_i, \mathbf{T}_j)$.

document labeling may be incomplete, we assign an empty label $L_\emptyset$ to documents that were not assigned any labels by the classifier.

2.2 Template Graph Construction

In this section we explain how the template graph is constructed given the fixed text representation described in the previous section.

Graph edges are constructed by selecting the k-nearest neighbors per node by similarity. We calculate edge weights using two metrics, cosine similarity and KL divergence. The two metrics used for graph construction are widely used for text similarity in information retrieval literature. In particular, KL divergence has been shown to be useful for graph construction over textual corpora [17]. The final edge weights are then computed by normalizing the weights of all outgoing edges from a node to sum up to one.

The weight of term x in template $\mathbf{T}$ is denoted by $w(x, \mathbf{T})$. For bag-of-words terms, this is a binary weight, which avoids over-weighting repeated fixed terms in the template (e.g., repetitions of the word *price* in receipts). For LDA topic representations, this is a topic weight assignment.

For further derivations we also define term probability

$$p(x|\mathbf{T}) = \frac{w(x, \mathbf{T})}{\sum_{x \in \mathbf{F}^{\mathbf{T}}} w(x, \mathbf{T})}, \tag{4}$$

and its smoothed version

$$\widetilde{p}(x|\mathbf{T}) = \frac{w(x, \mathbf{T}) + \epsilon}{\sum_{x \in \mathbf{F}^{\mathbf{T}}} w(x, \mathbf{T}) + |\mathbf{F}^{\mathbf{T}}|\epsilon}, \tag{5}$$

where ϵ is a small constant used for Laplacian smoothing.

Table 1 provides details on both the cosine similarity and the KL divergence metrics for graph construction.

2.3 Label Assignment and Propagation

Recall, from Section 2.1 that we associate a label distribution with a template. In this paper, we assume an existence of an underlying high-precision / low-recall classifier that assigns a single label to each document. Template label distribution is based on these document-level labels. The description of the document-level classifier is out of the scope of this paper, and for the remainder of this section, we assume that this classifier may be based on either hand-written rules or a supervised model that is specifically optimized for document-level precision. Such classifiers are common in critical user-facing tasks, where the cost of false positives is very high. An example of such a task is email spam detection, where marking something as spam may prevent a user from ever seeing an email.

In such high-precision scenarios, the classifier will err on the side of caution. Our approach enables grouping classifier decisions by a structural template, and improving the classifier recall, without sacrificing precision. Therefore, in the models described in the next sections we develop methods to assign a single optimal label $L^{\mathbf{T}}_{OPT}$ to a template based on the label distribution of its documents and the label distribution of its neighboring templates in the template graph.

Given that all the documents come from the same underlying template, the classifier decision should be unanimous for the template, which is the property we leverage in the *majority label* approach described in Section 2.3.1.

In addition, similar templates should have similar labels, which is addressed by the *centroid similarity* and the *hierarchical label propagation* approaches described in Section 2.3.2 and Section 2.3.3, respectively.

2.3.1 Majority Label

The majority labeling classifier is our most direct method for labeling documents by way of their corresponding templates. We use this classifier as a baseline for comparison against more advanced graph-based techniques.

As explained in Section 2.1, a template is assoicated with a label distribution $\mathbf{L}^{\mathbf{T}}$. The majority label classifier thus labels a template according to the label which is assigned to more than 50% of the documents in the template. Formally, the assigned label for template $\mathbf{T}$ is determined as:

$$L^{\mathbf{T}}_{OPT} = \begin{cases} L_i & \text{if exists } L_i \text{ s.t. } p(L_i|\mathbf{T}) > 0.5 \\ L_\emptyset & \text{otherwise} \end{cases}$$

For instance, if template $\mathbf{T}$ is comprised of 100 emails, 80 of which are labeled *Receipt* and 20 of which are labeled *Finance* by the baseline document classifier, then $L^{\mathbf{T}}_{OPT}$ = *Receipt* and all 100 emails in $\mathbf{D}^{\mathbf{T}}$ will be relabeled as *Receipt*.

On the other hand, if template $\mathbf{T}$ is comprised of 100 emails, with ten labels and ten emails assigned to each label, $L^{\mathbf{T}}_{OPT} = L_\emptyset$, and the emails in $\mathbf{D}^{\mathbf{T}}$ will not be relabeled.

Note that the majority label approach utilizes solely intra-template information for classification. It thus ignores the template graph structure, which is addressed by the approaches discussed in the next sections.

2.3.2 Centroid Similarity

The centroid similarity approach takes advantage of inter-template relationships for template label assignment. Templates are represented using their fixed text $\mathbf{F}^{\mathbf{T}}$, as discussed in Section 2.1. We then derive a set of *seed templates* for each label L_i (excluding the empty label) such that

$$\mathbb{S}^{L_i} = \{\mathbf{T} | p(L_i|\mathbf{T}) = 1\}. \tag{6}$$

In other words, seed templates are templates for which label

assignment is already provided with 100% confidence by the underlying high-precision document classifier.

For each seed template set $\mathbb{S}^{L_i}$, we compute its centroid vector by averaging the fixed text vectors $\mathbf{F^T}$ of its templates. Then, for every non-seed template $\mathbf{T}$ with label distribution $\mathbf{L^T}$, we compute its similarity to the centroids of the labels in $\mathbf{L^T}$. Similarity is computed using either one of the metrics shown in Table 1. We then assign the label of the *closest* centroid to template $\mathbf{T}$ and relabel all the documents in the template accordingly.

For instance, for a template comprised of 20 *Receipt* emails, 20 *Finance* emails emails, and 20 unlabeled emails, we will compute its distance to the *Receipt* and *Finance* centroids. If the *Receipt* centroid is the closest to the template, we will relabel all the 60 emails in the template as *Receipt*.

Note that unlike the majority labeling approach, the centroid similarity approach can assign labels to templates with uniform label distributions. It can even assign labels to templates in which the majority of the emails are unlabeled.

Centroid similarity leverages to some degree the structure of the template graph, however it only considers template aggregates (centroids) and not individual templates. If centroids represent clusters that are very broad, this could lead to a loss of precision in labeling, which we address by introducing the *hierarchical label propagation* approach.

2.3.3 Hierarchical Label Propagation

While the centroid similarity technique generally improves over majority labeling, it is built on the very broad assumption that a label has a single global centroid from which all templates of that label can be derived. However, this may not be the case for all labels.

Consider for example a *Travel* label which is comprised of subcategories such as flight itineraries, hotel reservations, and car rental information. While the intra-subcategory templates may be closely clustered together (i.e. have high similarities between one another in terms of shared vocabulary), it is not clear if the same holds for the inter-subcategory templates as well. Hence, we must consider the possibility that a single label will in fact have multiple clustering centers and cannot be represented by a single centroid.

Label propagation [27, 26] is a well known graph algorithm which propagates labels between neighboring nodes weighted by similarity. In our case, nodes are represented by templates and the graph is constructed as described previously in Section 2.2.

The label propagation algorithm that we use in this work is simple, yet effective. It operates in a hierarchical two-stage fashion. At the first stage, we run the label propagation on the template graph. At the second stage we propagate the labels derived on the template level to the individual documents.

Template propagation stage. We first construct a graph $G = (V, E, W)$ as described in Section 2.2. Here, $V = \{\mathbf{T}\}$ refers to the set of template nodes, E is the set of edges (between a template node and its k-nearest neighbors) and W is the edge weight matrix computed using KL divergence or cosine similarity. We use a set of labeled seed nodes $\mathbb{S} \subset V$ as defined in Equation 6 and provide this as input to the algorithm. The label set is L, where the size $|L| = m$. The goal of the template label propagation algorithm is a soft assignment of labels $\mathbf{L^T}$ to each template node $\mathbf{T}$ in the graph [7, 26, 27, 24].

Our method uses the EXPANDER [24] framework and algorithm for propagation and learning label distributions. The propagation procedure utilizes an efficient iterative algorithm which defines an approximate solution at the $(i+1)th$ iteration, given the solution of the $(i)th$ iteration. In the first iteration of template propagation, seed nodes broadcast their labels to their k-nearest neighbors. Each node that receives an incoming broadcast from at least one neighboring node updates its existing label distribution according to the weights of the incoming message and the transmitted label distribution. In subsequent iterations, all nodes which have some label distribution participate in broadcasting their label distributions, and the procedure repeats until the propagated label distributions converge. In practice, the algorithm converges within a few ($k = 10$) iterations.

Document propagation stage. After convergence of the template propagation stage, we further propagate the labels from the templates to their documents. For this stage, we simply propagate the most likely label of the template to all of its constituent documents. Formally,

$$L^{\mathbf{T}}_{OPT} = \arg\max_{L_i} \hat{p}(L_i|\mathbf{T}),$$

where $\hat{p}(L_i|\mathbf{T})$ denotes the probability of label L_i according to distribution $\hat{\mathbf{L}}$, *after* the template propagation stage.

The two-stage hierarchical label propagation has two main advantages over the previously discussed methods. First, unlike the centroid similarity approach, it leverages the initial template label distribution as a *prior distribution* in the template propagation stage. Second, it iteratively updates this prior distribution based on the evidence from the neighboring nodes in the template graph, and propagates the updates to the template constituent documents. In Section 4 we show that these advantages lead to the superior performance of the hierarchical label propagation method compared to the other baselines.

3. RELATED WORK

In this section we discuss the state of the art techniques for email classification, including threading, foldering, and ranking, as well as techniques for template induction and label propagation.

3.1 Email Organization

Popular email management techniques include threading, foldering, and ranking.

Text classification has been one of the more popular machine learning applications over the last couple decades. Applications include language identification [19], genre analysis [13], and sentiment classification [20]. Automated email management methods include email categorization, which often employ text classification techniques. Bekkerman et al. explicitly explore the challenges of applying traditional document classification techniques, such as maximum entropy classifiers, naive bayes classifiers, and support vector machines, to email foldering [6]. They present a new formulation of the winnow algorithm and conclude with a discussion on the remaining challenges and difficulties that exist in email foldering.

Email threading is a technique in widespread use today that visually groups related emails together. Specifically, emails are threaded by header information, such as sender email addresses, subject string, and reply and forward pre-

fixes. This technique is predominantly used to thread personal communications. However related machine-generated emails are often left unthreaded. For example, an e-commerce site might send multiple emails to a single user in response to a purchase. The first email might be a receipt or confirmation, followed by a shipment notification email, and followed finally by a delivery announcement or survey. Oftentimes each email in this correspondence has different subject lines and even different sender aliases which correspond to the specific notification type of the email. Current systems cannot thread these emails accurately due to their misaligned header information. Ailon et al. have presented techniques to thread these machine-generated emails through "causal threading" which leverages templates, learns causal relationships between emails from similar senders, and automatically threads incoming emails with distinct headers [2].

Automated email ranking is a relatively new technique which aims to assist in email management by classifying emails into important and unimportant categories. The goal is to reduce the amount of overhead required by the user in manually sorting through emails to find those with the highest importance. Possibly the most popular automated email ranking system in widespread use today is Google's Gmail priority inbox. This system ranks emails by the probability that a user will interact with an email in a meaningful way (e.g. open, reply, correct importance level) and within some amount of time from delivery [1]. They also go one step further by attempting to rank emails without explicit labeling from the user, although they do account for explicit labeling in their linear logistic regression models, which were explicitly chosen due to their ability to scale.

Along with the increased popularity of email over the last few decades, came the proliferation of spam. Unfortunately, the Simple Mail Transfer Protocol (SMTP), which is a standard protocol for electronic mail transmissions, has a number of limitations that spammers are able to capitalize on, and thus it is somewhat of a self-crippling system. While proposals exist to change SMTP in order to migrate the cost of spam from SMTP receiving servers to the spammers themselves (i.e. transition from a sender-push to a receiver-pull model) [12], changes to the underlying core email infrastructure would require a worldwide overhaul. As a result of these complexities, greylists and machine learning techniques have become the default methods for spam detection, many of which utilize textual analysis methods [21, 3, 25]. Caruana and Maozhen present a brief survey on current machine learning techniques for spam filtering, exploring both text classification methods as well as emerging approaches in peer-to-peer, social networking, and ontology-based semantic spam detection [9].

Other classification techniques have also been presented with the goal of assisting in email management. Bar-Yossef et al. present a new cluster ranking technique on mailbox networks—user-centric networks consisting of contacts with whom an individual exchanges email—to discover and identify communities [5]. They assert that their framework can be applied to generic clustering tasks as well. Grbovic et al. present methods for distinguishing between personal and machine-generated email and classifying messages into a few latent categories [14].

Most of the previously proposed classification, filtering, and ranking techniques rely on individual emails. In this paper, we demonstrate that they can be further improved for machine-generated email by applying them on a structural template level.

3.2 Template Induction

Data on the web is generally formatted in such a way that a machine can render it—thus presenting the data in a human-readable format—but not necessarily extract the pertinent information from it. For example, consider an itinerary sent from an airline company to its customer via email. The body of the message might contain images and text. A number of HTML tables and cells organize the images, boilerplate text, and pertinent information (i.e. date, flight number, airport codes) in such a way that when the message is rendered the important information is presented in a visually appealing manner to the user. However, it remains a difficult task for the machine to automatically locate among the HTML and text what is actually the central and important information in the document.

It has been estimated that over 60% of overall email traffic is comprised of machine-generated email [2], some of which contain personal and important information (e.g. purchases, tracking numbers, events, flights, hotel reservations). Techniques for template or wrapper induction have been extensively studied in an effort to enable the extraction of this structured data from web pages [4, 16, 18]. However, there is very little published work on using such structural templates for processing commercial email data.

3.3 Label Propagation

Label propagation operates on a geometric framework consisting of a set of labeled and unlabeled points. The goal is to predict the labels of unlabeled data using labeled data [27, 26, 7, 24]. Edge weights correspond to the relative similarities between points. In each iteration the label distributions of labeled points are propagated to each of their neighbors. In turn, data points that receive label distributions from one or more of their neighbors update their own label distributions according to a weighted sum of distributions proportional to the incoming edge weights.

This semi-supervised technique is especially useful in the age of big data considering that labeled data often come in very small quantities while unlabeled data can be orders of magnitude larger. A nice characteristic of this algorithm is that labels naturally cluster together based on their similarity metrics which enables multiple clusters per label to emerge. Gregory presents an extension of label propagation which enables multi-labeling for communities that might overlap one another [15]. In this paper, we use EXPANDER, the scalable label propagation framework presented by Ravi and Diao [24].

4. EVALUATION

4.1 Experimental Setup

We explore the use of two template representations in the experiments in the remainder of this section: bag-of-words and topic distributions. In both cases we only use terms found in the template's fixed text, as described in Section 2.1. For both representations, we discard 100 most frequent terms that appear over the entire template corpus, terms which appear fewer than 11 times in the entire template corpus, and terms which appear in fewer than 6 unique sender domains. Additionally, we stem words using

Label	Example of email types	# Labeled Templates
Finance	Financial statements, stock reports, bank account updates.	262
Receipt	Purchase receipts, order confirmations, shipping notices.	432
Travel	Travel itineraries, hotel reservations, car rentals.	201

Table 2: Experimental labels.

	Majority Label		Centroid Similarity		Label Propagation		
	Precision	Recall	Precision	Recall	Precision	Recall	
Finance	55.00	83.97	60.73	89.30	91.49	98.47	m, c
Receipt	89.35	79.63	85.05	95.87	97.82	93.52	c
Travel	86.57	86.57	86.62	95.33	95.00	94.53	m, c

Table 3: Label propagation vs. baselines: majority label, centroid (bag of words). Significant differences with Majority Label and Centroid Similarity are marked by m and c, respectively (statistical significance measured using McNemar's test with $p < 0.05$).

the Porter stemming algorithm [22] and only consider English templates for the purposes of evaluation. While we only investigate and evaluate English templates, since our techniques are unsupervised and do not take into account any language-specific features, we expect that they should generalize to other languages without substantial loss in effectiveness.

As a corpus, we generate templates over mailboxes of company employees who voluntarily signed up in the evaluation and agreed to participate in our experiment. We use several thousands such mailboxes with a total of close to a million machine generated emails. This data was treated with utmost care: for our evaluation, we examine only the generated templates, and not any individual emails.

We manually label the templates into three predefined categories in Table 2 for evaluation purposes. In order to limit the space of labeled templates, in the majority of our experiments, we only label templates for which the existing classification system (which is outside the scope of this paper) labels at least one email in a template.

Each template is labeled by two human annotators. During evaluation we found that inter-annotator agreement was above 90%. Templates for which no agreement was established are disregarded for evaluation. Templates that did not belong to any of these predefined categories were marked as "unlabeled".

4.2 Template Classification Experiments

4.2.1 Baselines

As baselines, we compare the precision and recall for majority label and centroid similarity label assignment techniques (described in Section 2.3.1 and Section 2.3.2, respectively) to the precision and recall of the hierarchical label propagation approach (described in Section 2.3.3). In this section, we use cosine similarity and bag-of-words template representations for template graph construction.

These baseline comparisons are shown in Table 3. The comparisons show that, for most labels, centroid similarity improves both precision and recall over the simple majority label approach. In particular, recall is significantly increased, which is not surprising, given that we make use of the inter-template information.

Overall, hierarchical label propagation outperforms both of these approaches to a statistically significant degree. For instance, for the *Finance* label, it increases the recall by more than 10%, and precision by more than 50% compared to the centroid similarity approach. This demonstrates that using local template similarities is more beneficial than the global centroid similarity alone.

Due to its superior performance, in the next section, we focus on the performance of the hierarchical label propagation model. We analyze its performance with different similarity metrics and template representations.

4.2.2 Hierarchical Label Propagation

We explore four different instantiations of the template graph consisting of the combinations of topic and bag-of-words models and using cosine similarity and KL divergence as similarity metrics. Table 4 compares these combinations of similarity metrics and template representations for the hierarchical label propagation.

We find that while the hierarchical label propagation with topics representation performs better than majority label and is consistent across all labels, the technique effectively suffers from a loss of resolution due to the use of Latent Dirichlet Allocation, a form of dimensionality reduction, and cannot improve much beyond about 90% for both precision and recall regardless of the underlying similarity metric and graph fan-out.

Overall, the bag-of-words fixed text term representation has the most consistent performance across all labels[1]. Both KL divergence and cosine similarity yield very similar results, and in both cases, both precision and recall are above 95%, which empirically validates the effectiveness of the hierarchical label propagation approach.

We also investigate the impact of graph connectivity on the precision and recall of the hierarchical label propagation in Figure 3. We find that precision and recall is the highest when each node is connected to its nearest 10 neighbors. While Figure 3 depicts the bag-of-words and cosine similarity case, we observe similar results for the other instantiations of the hierarchical label propagation as well. The intuition behind this result is the iterativeness of the label propagation algorithm, due to which graph connectivity plays a central role in label propagation accuracy. For sparsely connected graphs, it becomes more difficult for the correct label to propagate to and affect the label distribution of distant

[1]However, we should note that the changes compared to topic representation are in all but one case not statistically significant.

	Topics [KL]		Topics [CS]		BoW [KL]		BoW [CS]		
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	
Finance	80.44	83.21	83.09	86.26	93.13	93.13	91.49	98.47	
Receipt	88.40	82.87	89.78	85.42	93.47	96.06	97.82	93.52	
Travel	88.61	89.05	86.79	91.54	95.24	89.55	95.00	94.53	t_{cs}

Table 4: Comparison of algorithms for template label propagation: precision/recall per label. Template graph fan-out $k = 10$. Significant differences with Topics [CS] is marked by t_{cs} (statistical significance is measured using McNemara's test with $p < 0.05$).

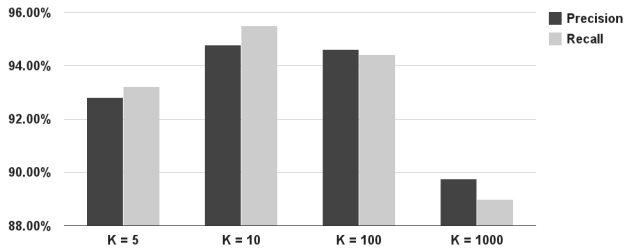


Figure 3: Precision/recall as a function of label propagation graph fan-out.

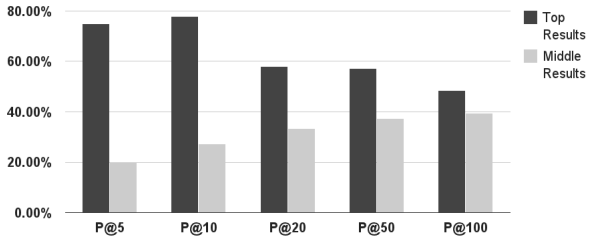


Figure 4: Precision@{1-100} for unlabeled templates. Results reported for both top ranked templates, and mid-ranked templates as comparison. P@X corresponds to the measured precision over X templates beginning at either the top or middle of the unlabelled set when sorted in descending order by most likely label weight.

	Precision	Recall	Email coverage
Finance	93.08	99.64	+9.7%
Receipt	99.13	91.39	+16.8%
Travel	95.93	93.52	+5.7%

Table 5: Coverage increase per label.

nodes. On the other hand, densely connected graphs may have many weakly weighted edges that ultimately introduce noise to the propagated label distributions.

4.3 Coverage

One of the primary goals of this work is to increase the coverage of the existing labels using the structural template graph, compared to a high-precision email-based classifier. In our first coverage experiment, we run the bag-of-words cosine similarity hierarchical label propagation on all the templates that have at least one underlying sample labeled with either of the three aforementioned labels. In this way, we can use the sample email labels as a prior distribution for the label propagation algorithm.

The coverage results of this run are shown in Table 5. Precision and recall numbers in Table 5 are computed for only the non-seed templates. Email coverage numbers demonstrate the increase in the number of emails covered by the labels when applying the propagated template labels to their constituent documents. These results show that we can achieve, on average across labels, 11% increase in email coverage, while maintaining precision and recall well above the 90% mark.

Note however that while all the templates in this experiment have some hint of being labeled by one of the three aforementioned categories, there exists a large unlabeled template set. Therefore, in our second coverage experiment we instantiate a template graph consisting of all the templates, including those which do not have emails labeled as *Finance*, *Receipt*, or *Travel*. The newly added unlabeled templates outnumber the previous set of templates 7 to 1. Again, we represent templates using bag-of-words, calculate edge weights using cosine similarity, and connect each node to the nearest 10 neighbors.

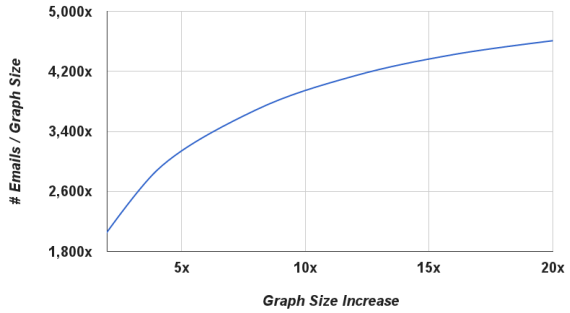
After running label propagation we order all previously unlabeled templates by the weight of their most likely label $L^{\mathbf{T}}_{OPT}$. We then conduct two evaluations on this list. First, we measure precision at x top ranks (P@x) for the first 100 templates in the list. These precision numbers are represented by the dark-grey bars in Figure 4. Note that the top 10 templates are labeled with a total precision (over all three labels) of about 75% while after the top 20 templates precision drops to below 60%. Second, we measure P@x starting at the middle of the template list sorted by the most likely label. The light-grey bars in Figure 4 represent these precision numbers, which hover around 40%.

Overall, the comparison between the dark and light grey bars in Figure 4 demonstrates that there is a correlation between the label weights assigned by the hierarchical label propagation, and their correctness. Higher weighted labels are more likely to be correct than the lower weighted labels.

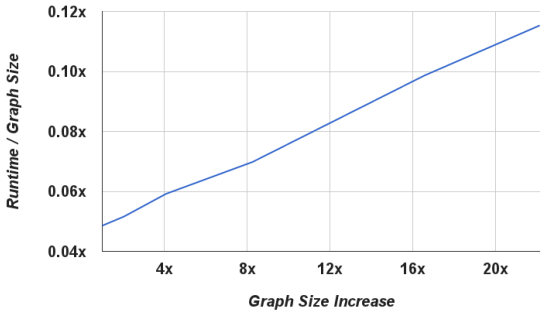
The biggest factor contributing to low measures of precision in Figure 4 is the fact that the hierarchical label propagation attempts to label the entire graph using only the labels it is provided (i.e. *Finance*, *Receipt*, *Travel*), when in fact, there are potentially a number of new labels that must be introduced to cover the newly added set. We discuss the prospect of label discovery in Section 4.5.

4.4 Scalability

As previously mentioned, one of our major motivations for using templates to classify emails is their scalability. According to Figure 5a, nearly 80% of emails in our corpus can be



(a) Email coverage.



(b) Label propagation runtime.

Figure 5: Email coverage and label propagation runtime as a function of the template graph size, as measured by the number of templates included in the graph.

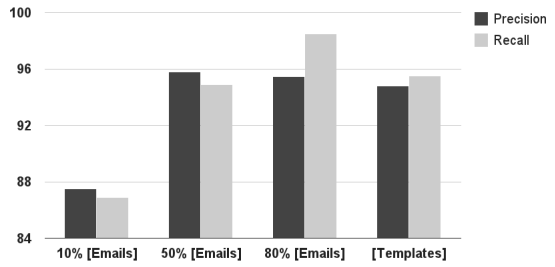


Figure 6: Comparison of sender/template level precision/recall for label propagation over email/template graphs.

covered using only 50% of the templates. Furthermore, the absolute number of templates required is up to three orders of magnitude less than the number of emails covered, since a single template generally represents 100 to 1,000 emails, as seen in Figure 5a. Since the runtime of the label propagation algorithm is approximately proportional to the size of the graph, as seen in Figure 5b, by using templates for classification instead of emails we can reduce runtime by orders of magnitude.

While templates scale better than emails, it is important to also consider the trade-offs in terms of the classifier performance. In Figure 6 we compare the effectiveness of the two approaches. Since emails do not have an equivalent label distribution that is characteristic to templates, and is used for seed selection, we cannot select seed nodes for the email graph using the same techniques used for the template graph. In place of the seed selection procedure, we experiment by randomly selecting some percentage of labeled emails as seed nodes (this percentage is labeled in Figure 6). We find that the template-based hierrarchical label propagation precision is comparable to using emails in the underlying graph structure, when we use either 50% or 80% of the emails as seed nodes for label propagation, and its recall is comparable to the 50% case.

4.5 Label Discovery

Table 6 demonstrates top topics for templates to which no existing labels were propagated. As such, these templates potentially belong to new undiscovered labels. Recall that each topic x, has a certain weight $w(x, \mathbf{T})$ for template $\mathbf{T}$. Topics are ranked for each label L (including an empty label $L_\emptyset$ for unlabeled templates) as:

$$\frac{\sum_{\mathbf{T} \in L} w(x, \mathbf{T})}{\sum_{\mathbf{T}} w(x, \mathbf{T})}.$$

Intuitively, top topics per label are the topics that are more likely to be associated to the label (including the empty label), compared to the entire collection.

The top topics for the unlabeled emails reveal interesting potential common themes for the unlabeled set. For instance, there are topics for live music shows, fashion promotions or newsletters, and business-related announcements.

These preliminary results indicate that our label propagation technique, in conjunction with template-based topic models, is useful not only for improving coverage for a prespecified label set, but also for discovering new emerging labels from the data.

5. CONCLUSION

In this paper, we have presented three new algorithms for classification of machine-generated emails using structural templates: majority label, centroid similarity, and hierarchical label propagation. We explored a number of underlying template representations including bag-of-words and topic distributions, different similarity metrics in template graph construction, and varying degrees of graph connectivity.

Our empirical evaluation demonstrates the superior effectiveness of the hierarchical label propagation technique using a template graph. Not only does it scale far better than email-based label propagation, but it is also comparable to it in terms of precision and recall. Hierarchical label propagation achieves above 90% precision and recall for all labels, while increasing the labeling coverage by more than 11%. Our experiments also demonstrate that hierarchical label propagation, along with template topic modeling, can be used for discovery of new labels in the corpus.

Unlabeled						
Business & Tech	*Music*		*Social*	*Fashion*	*Politics*	
develop	music	music	story	fashion	say	thank
technolog	world	ticket	her	deal	govern	contribut
job	love	show	his	sale	american	support
learn	artist	live	video	design	polit	campaign
compani	star	band	friend	brand	obama	donat

Table 6: Top topics for unlabeled templates. Each column shows top 5 stems per topics. Headers in italics represent a human-assigned topic label.

6. REFERENCES

[1] D. Aberdeen, O. Pacovsky, and A. Slater. The learning behind gmail priority inbox. In *NIPS Workshop on Learning on Cores, Clusters and Clouds*, 2010.

[2] N. Ailon, Z. S. Karnin, E. Liberty, and Y. Maarek. Threading machine generated email. In *Proceedings of WSDM*, pages 405–414, 2013.

[3] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, and C. D. Spyropoulos. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of SIGIR*, pages 160–167, 2000.

[4] A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proceedings of SIGMOD*, pages 337–348, 2003.

[5] Z. Bar-Yossef, I. Guy, R. Lempel, Y. Maarek, and V. Soroka. Cluster ranking with an application to mining mailbox networks. *Knowledge and Information Systems*, 14(1):101–139, 2008.

[6] R. Bekkerman, A. McCallum, and G. Huang. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. *Computer Science Department Faculty Publication Series*, page 218, 2004.

[7] Y. Bengio, O. Delalleau, and N. Le Roux. Label propagation and quadratic criterion. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*, pages 193–216. MIT Press, 2006.

[8] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[9] G. Caruana and M. Li. A survey of emerging approaches to spam filtering. *ACM Computing Surveys (CSUR)*, 44(2):9, 2012.

[10] L. A. Dabbish and R. E. Kraut. Email overload at work: an analysis of factors associated with email strain. In *Proceedings of the Conference on Computer Supported Cooperative Work*, pages 431–440, 2006.

[11] L. A. Dabbish, R. E. Kraut, S. Fussell, and S. Kiesler. Understanding email use: predicting action on a message. In *Proceedings of SIGCHI*, pages 691–700, 2005.

[12] Z. Duan, Y. Dong, and K. Gopalan. DMTP: Controlling spam through message delivery differentiation. *Computer Networks*, 51(10):2616–2630, 2007.

[13] A. Finn, N. Kushmerick, and B. Smyth. Genre classification and domain transfer for information filtering. In *Proceedings of ECIR*, pages 353–362. 2002.

[14] M. Grbovic, G. Halawi, Z. Karnin, and Y. Maarek. How many folders do you really need?: Classifying email into a handful of categories. In *Proceedings of CIKM*, pages 869–878, 2014.

[15] S. Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103018, 2010.

[16] C. Hachenberg and T. Gottron. Locality sensitive hashing for scalable structural classification and clustering of web documents. In *Proceedings of CIKM*, pages 359–368, 2013.

[17] O. Kurland and L. Lee. Pagerank without hyperlinks: Structural re-ranking using links induced by language models. In *Proceedings of SIGIR*, pages 306–313, New York, NY, USA, 2005. ACM.

[18] N. Kushmerick. *Wrapper Induction for Information Extraction*. PhD thesis, 1997. AAI9819266.

[19] B. Martins and M. J. Silva. Language identification in web pages. In *Proceedings of SAC*, pages 764–768, 2005.

[20] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86, 2002.

[21] P. Pantel et al. SpamCop: A spam classification & organization program. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, pages 95–98, 1998.

[22] M. Porter. The porter stemming algorithm, 2009. http://tartarus.org/~martin/PorterStemmer/.

[23] S. Radicati. Email statistics report, 2014-2018, 2014. http://www.radicati.com/wp/wp-content/uploads/2014/01/Email-Statistics-Report-2014-2018-Executive-Summary.pdf.

[24] S. Ravi and Q. Diao. Large scale distributed semi-supervised learning using streaming approximation. *arXiv preprint arXiv:1512.01752*, 2015.

[25] L. Zhang, J. Zhu, and T. Yao. An evaluation of statistical spam filtering techniques. *ACM Transactions on Asian Language Information Processing (TALIP)*, 3(4):243–269, 2004.

[26] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Proceedings of NIPS*, pages 321–328, 2004.

[27] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, CMU-CALD-02-107, Carnegie Mellon University, 2002.