Annotating Needles in the Haystack without Looking: Product Information Extraction from Emails[*]

Weinan Zhang[1], Amr Ahmed[2], Jie Yang[2], Vanja Josifovski[3], Alex J. Smola[4]
[1]University College London, [2]Google Inc.
[3]Pinterest, [4]Carnegie Mellon University
w.zhang@cs.ucl.ac.uk, {amra,yangjie}@google.com
vanja.josifovski@gmail.com, alex@smola.org

ABSTRACT

Business-to-consumer (B2C) emails are usually generated by filling structured user data (e.g. purchase, event) into templates. Extracting structured data from B2C emails allows users to track important information on various devices.

However, it also poses several challenges, due to the requirement of short response time for massive data volume, the diversity and complexity of templates, and the privacy and legal constraints. Most notably, email data is legally protected content, which means no one except the receiver can review the messages or derived information.

In this paper we first introduce a system which can extract structured information automatically without requiring human review of any personal content. Then we focus on how to annotate product names from the extracted texts, which is one of the most difficult problems in the system. Neither general learning methods, such as binary classifiers, nor more specific structure learning methods, such as Conditional Random Field (CRF), can solve this problem well.

To accomplish this task, we propose a hybrid approach, which basically trains a CRF model using the labels predicted by binary classifiers (weak learners). However, the performance of weak learners can be low, therefore we use Expectation Maximization (EM) algorithm on CRF to remove the noise and improve the accuracy, *without* the need to label and inspect specific emails. In our experiments, the EM-CRF model can significantly improve the product name annotations over the weak learners and plain CRFs.

Keywords

Structured information extraction, Business intelligence

1. INTRODUCTION

Email is arguably one of the most successful applications on the Internet. In the early days email was primarily used for short textual user-to-user communication. Today, this role has been largely overtaken by chat and social network platforms. Now email increasingly serves as a repository of transactional information such as receipts from many business-to-consumer (B2C) interactions. Email messages have several properties that make them appealing for such communication: the communication is asynchronous — it does not require both parties to be present at the same time; it is virtually infallible; email provides persistent storage; finally it is mostly private, requiring authentication for access.

[*]Work done while all the authors were at Google.

KDD'15, August 11-14, 2015, Sydney, NSW, Australia
ACM 978-1-4503-3664-2/15/08.
http://dx.doi.org/10.1145/2783258.2788580.

While email is a very suitable communication *medium* for such information exchange, it was originally designed for PC screens with manageable amount of messages. However, nowadays more and more people are checking emails from mobile devices with smaller screens. It will be much more convenient and efficient for users to track important information without having to read every email on mobile devices. Actually it is a key feature of Google Now, just as those displayed in Figure 1.

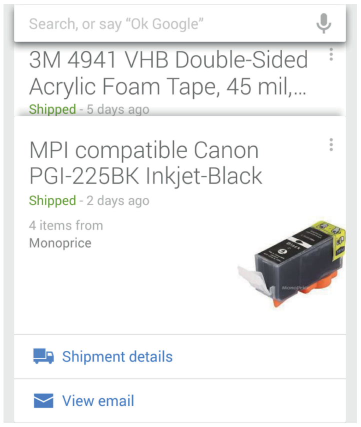
Most B2C emails are generated by filling a template with values from databases. We call it a templatized email. For example, retailers make receipt emails by populating order details (order number, items and prices, etc.) into a predefined template, and shipment details (tracking number, delivery date, etc.) into another template. Usually users only look for those details, not the templates or static contents. In other words, users only want to see the structured data from databases, which is embedded in the templates. In this paper we will propose a system to extract those data from emails, also known as "wrapper" [12] or "parser" [19] in the literature.

A simple but accurate way is to manually create a parser for each template. However, due to the privacy constraints, it will rely on the vendors to provide the templates, or hire people to craft the extraction rules using donated emails of this template. However, both approaches are not scalable and are vulnerable to changes of the template.

Another more desirable approach is to automatically extract the structured data from those templatized emails: construct the template from similar documents, extract the data, and annotate the semantic types (i.e., price, shipping address, delivery date, etc.) of those data entries. Eventually, the extracted structure data can be rendered in a different UI and served to the users, as what Google Now does. We will introduce such a system in section 3.

1.1 Challenges

Privacy. By law and by policy commercial email providers are held to a very high privacy standard. Users can assume rightfully that their messages are not read by other humans. Most major Internet companies have established protocols to ensure such guarantees automatically. This makes it impossible to generate broad training data of annotated emails, or to visually inspect the results of extractors applied to emails. In other words, unlike traditional information extraction scenarios, we have no access to *both the input and output* of the extractors, except a very small amount of donated emails.


Search, or say "Ok Google"

3M 4941 VHB Double-Sided Acrylic Foam Tape, 45 mil,...

Shipped - 5 days ago

MPI compatible Canon PGI-225BK Inkjet-Black

Shipped - 2 days ago

4 items from Monoprice

Shipment details

View email

Figure 1: Structured information presentation in Google Now (1/2014) from B2C shipping emails on Android 4.4. The product name, shipping date, quantity and retailer are extracted and annotated. The user can check shipment details via the extracted tracking number rather than needing to read the original emails.

Every design and implementation shall respect and protect user privacy. That is why it is hard for us to get more samples to analyze, find heuristics and train good models over the data we cannot see.

Diversity. Different websites have very different templates, the same site can have multiple templates (order confirmation, cancellation, etc.), and even the same template may yield different variants for personalized experience. In general, we are generating millions of templates from billions of emails. Each template has its own contents and characteristics, but we only have some donated emails from a small number of templates to train models, which in turn could yield very biased models with poor performance when apply to new templates. For example, a wrapper or model trained from Google Play receipts may not work well on Newegg.com receipts, but the problem is that we do not have donated training samples from Newegg, nor allow to view any personal emails from Newegg.

Complexity. In early days, many email templates only have one or a few simple tables, but now the HTML codes used in email become much more complex and nested for better visual presentation. In addition, a template consists of various semantic types. For instance, a purchase receipt may contain customer ID, order number, tracking number and URL, delivery date, shipping address, product name, total price, and so on, while an event email can have a totally different set of types.

Scalability. We are dealing with extremely large data, and have to process it in a very short time since people expect emails to arrive instantly. The daily volume of emails are a number of times larger than the web pages indexed per day by well known search engines, and it grows fast. Therefore, efficiency and scalability are the key issues of the system.

In summary, rigid user privacy constraints, diversity, complexity and lack of data are new challenges to wrapper techniques proposed previously for scalable email systems.

1.2 Problem Scope and Contributions

In this paper we address the problem of extracting personal structured information in the presence of reviewing restrictions and variations in emails. We take advantage of the repetitive structure of emails to induce templates. For the content that is not part of the template, i.e., the text regions changing in every email, we need annotate the semantic type so that it can be properly rendered to the user. Some types are easy to annotate, such as date, address, price, etc. We can pre-annotate them using some existing annotators. This paper focuses on non-trivial types without mature annotators, particularly the product name in email receipts, which was one of the most difficult problems in the system. Specifically, after we extract a text without any pre-annotated type, we need to predict if it is a product name or not. We can also apply the same solution to other non-trivial types.

Due to the diversity and lack of training data on all templates, general supervised learning models, such as binary classifiers, have poor performance for this task, although they can be trained with limited data. On the other hand, structure learning methods, such as Conditional Random Field (CRF), are more specific and accurate for inducing structures in templates of sufficient training samples, but it usually requires training labels of every template. Please note, here the elements of the structure learned by CRF are semantic types, not the HTML nodes or text regions. When we apply CRF on a templatized email, it can annotate the types of each node or region[1], such that we can use it to predict the type of any extracted text. If we train a CRF model from some templates and then apply to others templates, its predictions still could be poor, again due to diversity of the data and complexity of the model.

To solve this dilemma, we propose a hybrid method, which takes advantages of both the generality of classifiers and specificity of structure learners. The former can generate sufficient training labels, although it has low quality, and then we rely on the latter to remove the noise and discover high accurate structures, by applying an Expectation Maximization (EM) algorithm. In our experiments, the EM-CRF model can significantly improve the product name annotations over the weak learners and simple CRFs. In sum, our main contributions are:

- We describe the problem of extracting private structured information from email messages under privacy constraints and propose an appropriate training mechanism that does not require labels or human review.
- We propose an EM-style algorithm to compensate for the noise of the weak classifier using a CRF.
- We demonstrate the effectiveness of the framework over a large scale of emails, achieving 30% improvement in F1 measure.

2. RELATED WORK

Information extraction on the Web. Information extraction is a well-studied problem. In particular, since most commercial emails contain HTML, it is reasonable to assume that tools developed for Web pages could conceivably be extended to apply to emails, too. In fact, we will take advantage of this strategy in our model.

[7, 15] present heuristic rules to extract data from Web pages using characteristics of HTML files. [8] uses HTML tag strings to find useful patterns for extraction. In [30, 44], the data records from the Web pages are detected by visual features such as gaps between blocks, and then tagged based on the pre-detected tree structure in the HTML. In [12, 13], different extraction rules are incorporated in an automatic fashion. The above work heavily relies on the induced Web page templates and lack of the advantages of leveraging the other annotated data fields to adaptively improve the extraction/annotation performance, which can be problematic when the templates are mismatched. Therefore, learning based methods are proposed. In [46], the authors infer a hierarchical CRF model to simultaneously detect the data records and label the attributes. [45] exploits additional information about page element alignment by using a 2D-CRFs model. A more integrated model which dynamically learns the hierarchical structure of the Web page and labels the attributes is proposed in [47]. A prerequisite of these works is that the Web pages in the training data should be

[1]Region means the text of a particular node in the DOM tree [18] of the email HTML. It is the unit for labeling here.

fully labeled to train the CRFs models. This is impractical in our scenario where emails are not human accessible.

Email. Email is always among the most sensitive data on the Internet due to users' privacy concerns. Thus data availability is one of the main constraints for any of research or product work on emails [26]. Most work on emails content mining is concerned with anti-spam [4, 9], where the task is to perform binary classification to figure out whether an email is a spam or not [43, 20]. Moreover, classification [26, 24] and clustering [29, 27] techniques are explored for email message categorization into the user's predefined hierarchical folders. This task has some unique challenges [6], such as the varying of users' email filing habits, and the strong imbalance of different email folders. In previous works on email content mining, the email data used is always in the bag-of-words form [26], even if there are some sections (e.g. `from`, `to`, `subject` and `body`). This is quite different from our email data, where the HTML structure and tags can be heavily utilized to improve performance and dynamically include content (such as up-to-date advertisements). For machine generate emails, templating and threading are also studied to help re-organize the email list presenting [1]. Such work is different with our problem because it does not need to recognize the type of each text field (or region). Besides the email content mining, there are some works on email social network analysis and mining [3, 14] to infer email priority [42], to detect spam [41], and to analyze the social hierarchy [38] inherent in the data.

Semi-supervised Learning. To supplement expensive or even unavailable labeled data researchers have started devising semi-supervised learning algorithms which take advantage of additional unlabeled data. See e.g. [48] for an extensive summary and review. Usually semi-supervised algorithms take advantage of a reasonable hypothesis based on which the unlabeled data could be utilized to improve the learning model [49]. In the context of NLP, such a hypothesis can be modeled e.g. via generalized expectation criteria [31] which assumes that conditional class probabilities satisfy moment constraints; regularization via posterior constraints [17]; or by using extrinsic objectives [19]. Optimization proceeds e.g. via Lagrangian relaxation and dual decomposition [39, 40].

In our work we define a hypothesis class using structured extraction via a CRF and employ automatically generated (low-accuracy) annotations as (inconsistent) training data. In other words, we learn constraints between adjacent labels as a mechanism for extracting information about the structure of the documents. Moreover, we cannot assess the quality of the results directly since, due to privacy constraints, we are not allowed to view the extractions. This is what makes it highly challenging to apply moment constraints directly, since this would require knowledge about the commercial content of the emails in question.

Note that our work also links to information extraction with weak supervision, which was originally proposed for biomedical entity recognition [11] and then applied in other scenarios such as entity recognition from multi-lingual corpora [25] and search queries [34], and entity relation extraction [32, 21]. The 'weak' supervision, in previous work, normally comes from direct text matches with some knowledge bases. However, as pointed out in [37, 21], when the target text is very diverse and not aligned with the knowledge bases, supervision helps little. In our scenario with emails as input, the text is quite different and most of the entities such as product names and addresses have multiple variants. Thus it provides little help from the direct match of the entity names. Our approach can be viewed as a form of weak supervision with a novel notion of weak signals (i.e. low-accuracy annotations) coupled with domain adaptation.

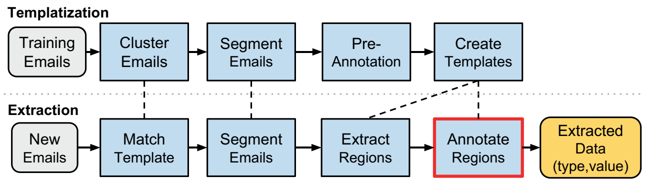


Figure 2: System components overview.

3. SYSTEM OVERVIEW

This section introduces the information extraction system, or wrapper, we developed for large scale email systems.

Some regions in a templatized email are static and do not change for different users or orders, which we call "fixed regions". Other regions vary in every document, such as `Order Number`, `Product` name. We call them "transient regions". Usually only transient regions contain user relevant data, and the template is made from fixed regions. We need to separate transient regions from fixed ones and annotate the semantic type of each transient region.

Figure 2 illustrates the components of our system. Basically, the system has the following major components.

Clustering. The very first step is to split emails by senders, then cluster similar emails which may be generated from a common template. Emails from the same sender are not necessarily from the same template, for instance some retailers may use one email address to send both order confirmation and shipment notification emails. It is also possible that multiple senders share the same template, which can be solved by merging clusters of different senders.

To preserve user privacy and avoid templatizing consumer-to-consumer (C2C) emails, we only induce templates from large clusters which contain many receivers and similar emails. Besides, it is easier to distinguish transient regions and fixed regions using lots of similar emails from the same cluster.

Templatization. We induce a template from emails of each cluster. First we segment each email into regions. The major difference between HTML and text emails is how we segment and locate the regions. Each HTML page has a DOM tree [18] and we can regard a text node as a region and use XPath to locate a node in the tree, while for plain text emails we need construct a finite-state machine (FSM) to represent the structure. Moreover, HTML pages may contain tables or hyperlinks, which can be used to get more features. The processes are similar for HTML and text emails. Without loss of generality, we only consider HTML emails below.

We declare regions with high document frequency as fixed. The rest is transient. Eventually a template consists of a dictionary of values in fixed regions (e.g. "Your order of", "Thank you!") and the location (XPath) of transient regions.

We also pre-annotate easy types, such as `Price`, `Address`, `Date` and store the mapping from XPath to its pre-annotated semantic type in the template, because usually the same node among templatized emails has a consistent type. For example, the cell of third row and fourth column of the second table is always `Price`. In case it is inconsistent, we store all the possible types with their probability.

Extraction. The previous steps are done offline, while extraction is performed online. That is, when a non-spam B2C email is received, we find its cluster and template, segment the email to regions, match the regions in the template, decide if it is a fixed region or transient, and finally get its type by the annotation module. The outcome is a list of (type, value) tuples, the structured data we need.

Annotation. If a transient region only has one pre-annotated type, we assign it this type. For regions with multiple candidate types, we either use the most likely type, or let the annotator to decide which one to use based on the text value, potentially using the prior distribution of types, or apply some heuristics according to the surrounding texts (context). Usually only a small portion of transient regions in an email have pre-annotated types. The others are `Unknown`.

For purchase emails, we currently only have one annotator to predict whether the value of an unknown regions is `Product`. However, it is a non-trivial task. Different products may have very similar names, and the same product could have quite different names as well, because the retailers can choose any form to present the products to users. Some product names are very short, such as "9" (a movie) or "hi" (a song). It is very common to include some product specifications in the name, or intentionally add some terms for "search engine optimization". In addition, many emails include recommendations or ads, which also contain production names. As the growing of products available online, it is almost impossible to maintain a database of product names.

The rest of the paper focuses on how to annotate product name in transient regions of unknown type.

4. STATISTICAL MODEL

4.1 Weak Annotations

The low accuracy estimators that we have at our disposal are all binary. That is, for a given position they estimate e.g. the probability of it being a number, or a product name, or a date. In other words, for each position i for document x_j we have the probability

$$\pi_{ij} := p_{\text{weak}}(y_i = y_i^* | i, x_j) \tag{1}$$

indicating the probability of the label y_i, as inferred from a weak, low-accuracy heuristic p_{weak}, being equal to the true label y_i^*. Note that there is no need that these probabilities are properly normalized, i.e. typically we will have $\sum_{y_i} \pi_{ij} \neq 1$, as another form of normalization will be given in (10). We simply require that the probabilities are bounded, i.e. $\pi_{ij} \in [0,1]$.

To convert this information into an actionable statistical model we follow the lines of [23]. That is, we treat the above problem as one of observing the correct label (denoted as t_i = TRUE) with probability π_{ij} for every position i, as specified by the low-accuracy estimator. In this case, the probability of inferring the correct labels for x_j is given by

$$p(\text{correct}|x_j, \theta) = \sum_{(y_1,\ldots,y_n)} \Big[\prod_{i=1}^{n} \pi_{ij}\Big] p(y_1, \ldots, y_n | x_j, \theta). \tag{2}$$

Here $p(y_1, \ldots y_n | x_j, \theta)$ denotes the CRF modeling a non-trivial joint probability over annotations. Moreover, we assume that the correct label distribution factorizes, hence the $\prod_i \pi_{ij}$ term to capture the joint probability of the inferred labels $(y_1, \ldots, y_n)$ being correct. The model of Figure 3 captures the basic idea.

In the special case of unstructured estimation this reduces to the model of [23]. Moreover, in the case where $\pi_{ij} \in \{0,1\}$ this reduces to the setting where we simply sum over a subset of possible label combinations. Unfortunately, in all the above settings $p(\text{correct}|x, \theta)$ is not log-concave in $p(y|x, \theta)$. This complicates matters in terms of inference and we will need to resort to DC programming [22], often also referred to loosely as Expectation Maximization in this context, for optimization. Before going into specifics let us briefly describe the statistical model.

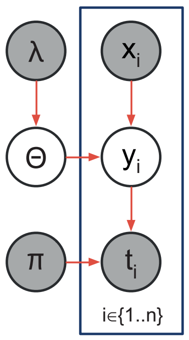


Figure 3: Partial observation model. Data x_i is generated iid from some underlying distribution $p(x_i)$, and labels are drawn from $p(y_i|x_i, \theta)$, as parameterized by θ in the form of a CRF. In turn, θ comes with its prior $p(\theta|\lambda)$ e.g. in the form of a Normal distribution. Finally, t_i, the binary variable showing whether the label y_i is the correct label y_i^*, is generated according to π_{ij}. We dropped dependency of y_i on x_{-i} as in standard CRF and the document index j for figure simplicity (see text for more details).

4.2 Conditional Random Fields

The primary benefit of using a CRF is that we may exploit structural correlation between adjacent labels [28]. This is achieved by expressing the entire chain of labels as a conditional undirected graphical model by multiplying adjacent clique potentials. In other words, we posit that the conditional label distribution is given by an exponential family model

$$p(y|x_j, \theta) = \exp\Big(\langle \phi(x_j, y), \theta\rangle - g(\theta|x_j)\Big), \tag{3}$$

where $\phi(x_j, y)$ is the feature function and the parameter θ acts as the coefficients of the features. Here $g(\theta|x_j)$ is the so-called conditional log-partition function ensuring that $p(y|x_j, \theta)$ is properly normalized as a distribution over y. It is well known that $g(\theta|x_j)$ is a convex function in θ. Computing $g(\theta|x_j)$ and its derivatives can be accomplished by dynamic programming. For this purpose we exploit that

$$g(\theta|x_j) = \log \sum_y \exp\left(\langle \phi(x_j, y), \theta\rangle\right), \tag{4}$$

$$\partial_\theta g(\theta|x_j) = \sum_y \phi(x_j, y) \exp\left(\langle \phi(x_j, y), \theta\rangle - g(\theta|x_j)\right) \tag{5}$$

$$= \mathbf{E}_{y \sim p(y|x_j, \theta)}\left[\phi(x_j, y)\right]. \tag{6}$$

Since the sufficient statistics $\phi(x_j, y)$ decomposes into terms on maximal cliques $(\ldots \phi(y_i, x_j), \phi(y_i, y_{i+1}, x_j), \ldots)$, it is sufficient to have access to $p(y_i|x_j, \theta)$ and $p(y_i, y_{i+1}|x_j, \theta)$ if we deal with the chain CRF [28]. Both terms can be efficiently computed using dynamic programming for $\mathbf{E}_{y_i|x_j,\theta}\left[\phi(y_i, x_{ij})\right]$ and $\mathbf{E}_{(y_i, y_{i+1})|x_j,\theta}\left[\phi(y_i, y_{i+1}, x_j)\right]$ respectively. The forward-backward algorithm suffices since we are dealing with a chain model. For more sophisticated structural evaluation models we would need to resort to a matching strategy satisfying the conditions of the Generalized Distributive Law [2] on the (log,+) semiring.

4.3 Expectation Maximization

Compared with standard CRF models, the key challenge in our scenario is that we do not have access to the correct labels y^* directly but rather only via π_{ij} as obtained by rather much weaker models. Hence, instead of dealing with the log-likelihood $\log p(y|x_j, \theta)$ from (3), we are dealing with $\log \sum_y [\prod_i \pi_{ij}] p(y|x_j, \theta)$ from (2), which is a nonconvex objective function. More specifically, in the case of a CRF the objective decomposes via

$$\log \sum_y \Big[\prod_i \pi_{ij}\Big] p(y|x_j, \theta) \tag{7}$$

$$= \log \sum_y \Big[\prod_i \pi_{ij}\Big] \exp\left(\langle \phi(x_j, y), \theta\rangle\right) - g(\theta|x_j). \tag{8}$$

As can be seen, the first term is convex and the second is concave. Hence, for the purpose of maximizing the log-likelihood

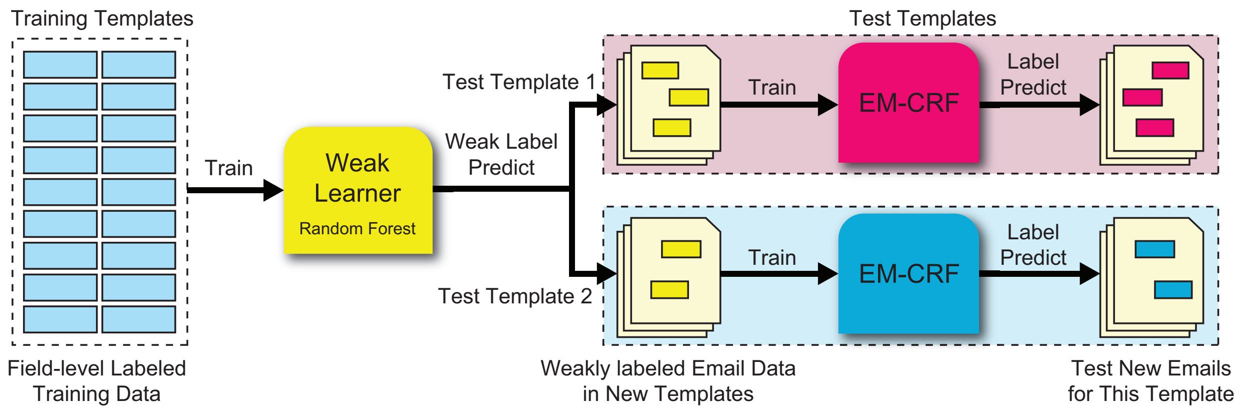


Figure 4: Data flow for training and evaluating the EM-CRF model. Auxiliary annotated regions are used to generate a random forest classifier as low-accuracy annotator of the template-specific emails. This data is then fed into the EM-CRF model to perform prediction on new documents.

Algorithm 1 CRF Inference with Side Labels (EM-CRF)

```
Require: Document set $\{x_j\}$
Require: Initial distributions $q_j(y)$, e.g. $q_j(y) = \text{const}$.
  Initialize CRFs parameters $\theta$, e.g. $\theta = 0$.
  while not converged do
    for each document $x_j$ do
      E-step: Update label distribution $q_j(y)$ via (10)
    end for
    M-Step: Train CRFs parameters $\theta$ maximizing (7)
  end while
  return $\theta$
```

we can lower-bound it by linearizing the first term via a Taylor approximation. This yields

$$\begin{aligned}
&\log \sum_y \Big[\prod_i \pi_{ij}\Big] p(y|x_j,\theta) \\
\geq & c + \Big\langle \theta, \partial_\theta \log \sum_y \Big[\prod_i \pi_{ij}\Big] \exp\left(\langle \phi(x_j,y),\theta\rangle\right)\Big\rangle - g(\theta|x_j) \\
= & c + \Big\langle \theta, \sum_y q_j(y)\phi(x_j,y)\Big\rangle - g(\theta|x_j),
\end{aligned} \tag{9}$$

where the distribution $q_j(y)$ is given by

$$q_j(y) = \frac{[\prod_i \pi_{ij}] \exp\left(\langle \phi(x_j,y),\theta\rangle\right)}{\sum_{y'} [\prod_i \pi_{ij}] \exp\left(\langle \phi(x_j,y'),\theta\rangle\right)} \propto \Big[\prod_i \pi_{ij}\Big] p(y|x_j,\theta). \tag{10}$$

Comparing with (3), the likelihood calculation in (9) takes the weighted average of $\phi(x_j, y)$ across the candidate labels y w.r.t. $q_j(y)$. In other words, we re-weight the conditional label estimates according to the outcomes of the low-accuracy annotators and the CRF model prediction using the current weight θ. Subsequently terms are renormalized.

Optimization proceeds by alternating between maximization of the lower bound of the log posterior over θ using the current estimate of label distribution $q_j(y)$ and by recomputing a new approximation $q_j(y)$ using the current estimate of CRF-weights θ. Note that by construction the lower bound is tight at the point of expansion. This follows directly from the fact that Taylor expansions are exact at the point of expansion. To train the CRF in the M-step, we generate a set of candidate labels for each sequence and weight each of them using $q_j(y)$, and then use a standard convex solver to maximize (7). We add an L_2 penalty on the weights θ of the CRF for regularization as is common in structured estimation. Algorithm 1 summarizes the inference procedure.

5. NEEDLES IN THE HAYSTACK

5.1 Setup

The key challenge in the experimental setup is to demonstrate the efficacy of the algorithm without the need for explicitly labeled training or test data and without the need to inspect the data, e.g. by means of editors. Yet, at the same time we want to have more directly quantifiable quantities than, say, predictive log-likelihoods. This makes assessment of the approach rather difficult.

The assumption is that while the templates might differ, we can at the very least assume that within a given template, the structure is sufficiently similar to apply a given CRF model. That is, we assume that each template comes with its own CRF model. In production, the data flow is as follows:

1. Human labeled region data from text unrelated to user's emails is used to train a random forest as the low-accuracy learner. This data contains only two possible labels — `Product` and `Other`. All the training data are performed over emails and templates that do not occur in test templates.
   Given the low quality estimator, we apply it to unlabeled emails to infer new templates. This allows us to mark up potential regions as `Product`.
2. In addition to the manually labeled auxiliary data we use a set of pre-annotators (independent of the email) to annotate easy types such as `Price`, `Email Address` and `Order Number` based on the textual description of the field. These annotators can be made from some heuristics or regular expressions.
3. For each new template (i.e. the templates from other senders or the new templates from the same senders) use the weak learner's prediction as weighted labels to train an EM-CRF model.
4. For each new email in the template we use the corresponding EM-CRF to predict the labels.

Figure 4 describes the data flow in such an environment. Note that training and test set are entirely disjoint. That is, emails required for testing the EM-CRF model are not used in the training of the EM-CRF.

In the above description, we assumed that the HTML template structure is fully given, that is, the template is entirely static with the only changes being relevant fields in question (e.g. purchase emails from a given sender). Section 6.2 provides results on the performance of the EM-CRF model in this context. Secondly, whenever the HTML structure is variable, with a similar method from [1], we can induce vari-

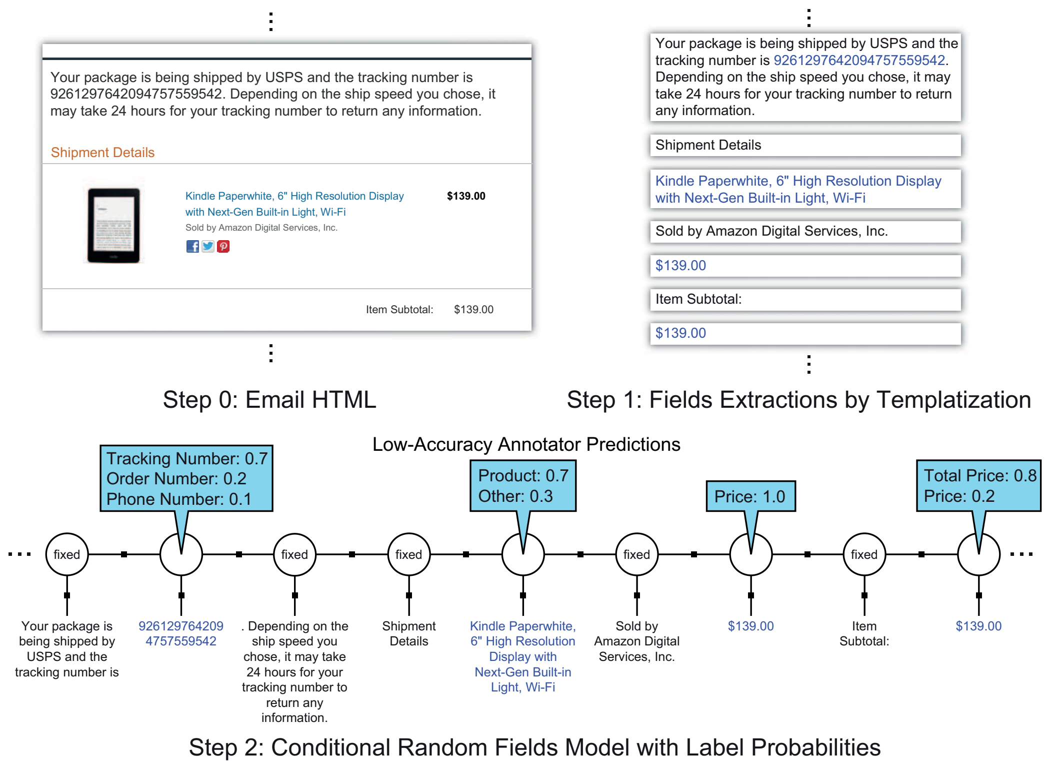

Your package is being shipped by USPS and the tracking number is 9261297642094757559542. Depending on the ship speed you chose, it may take 24 hours for your tracking number to return any information.
Shipment Details
Kindle Paperwhite, 6" High Resolution Display with Next-Gen Built-in Light, Wi-Fi
$139.00
Sold by Amazon Digital Services, Inc.
Item Subtotal: $139.00
Step 0: Email HTML
Your package is being shipped by USPS and the tracking number is 9261297642094757559542. Depending on the ship speed you chose, it may take 24 hours for your tracking number to return any information.
Shipment Details
Kindle Paperwhite, 6" High Resolution Display with Next-Gen Built-in Light, Wi-Fi
Sold by Amazon Digital Services, Inc.
$139.00
Item Subtotal:
$139.00
Step 1: Fields Extractions by Templatization
Low-Accuracy Annotator Predictions
Tracking Number: 0.7
Order Number: 0.2
Phone Number: 0.1
Product: 0.7
Other: 0.3
Price: 1.0
Total Price: 0.8
Price: 0.2
fixed
Your package is being shipped by USPS and the tracking number is
926129764209 4757559542
. Depending on the ship speed you chose, it may take 24 hours for your tracking number to return any information.
Shipment Details
Kindle Paperwhite, 6" High Resolution Display with Next-Gen Built-in Light, Wi-Fi
Sold by Amazon Digital Services, Inc.
$139.00
Item Subtotal:
$139.00
Step 2: Conditional Random Fields Model with Label Probabilities

Figure 5: Ingestion of an HTML email. Text is segmented into snippets according to the DOM tree. The labels for the variable regions are unknown. The low-accuracy predicted labels are used to initialize the CRF.

able and static contents using standard automatic methods based on text frequencies across emails in the same template.

5.2 From Email HTML to CRFs Data

Given an email in HTML, we need to transform it into a representation amenable to a CRF. This requires two stages — the actual segmentation and the annotation of content with low-accuracy classifiers, as illustrated in Figure 5.

Step 1. We begin by segment emails to regions using the system we described in section 3. The sequence of the text regions follows their order in the DOM tree of the HTML file. Some text regions are fixed, as part of the template, and some are transient, which may already have semantic types from pre-annotators. Moreover, some text regions may be highly relevant and specific to a receipt although they share the same DOM tree node with surrounding template text. For example, the order number 9261297642094757559542 as shown in step 1 of Figure 5. In such cases, the text regions will be further segmented (see step 2 of Figure 5).

Step 2. We align these text regions to build the linear CRFs data. The label of each fixed region of the template is set as `Fixed`, while the other regions will be given by the low-accuracy annotator (e.g. `Product`, `Other`) and pre-annotated types (e.g. `Price`, `Email Address`, etc.)

Compared with a more traditional word-level CRF annotator, we allow for the segments to contain several words. As such a possible problem is that the text features may not match across documents. In practice this turns out to be less of a problem than anticipated. This is due to two reasons: firstly, the template text regions are almost always repetitive; secondly, even for a new non-template node text, its label can be initially induced by the low-accuracy annotator or the manually crafted rules.

5.3 Training the Low-accuracy Annotator

The low-accuracy annotators are essentially binary classifiers which generate a conditional class probability. There are 32 high-level features of the following categories:

Content Based Number of words, number of digits, having a hyper-link, inside a table, etc.

Context Based Minimum and average distance to regions of price, order number, table header, etc.

Search Based If we send this text as a query to a search engine, how many results are from retailer domains.

Knowledge Based Find the knowledge graph entities in the text, and see how many of them are related to product or brand.

Annotation Based How much this text look like a price, date, address, time, etc.

We tried using N-Gram features, but it did not work well because first it largely increases the dimensionality, and second the feature distribution from one domain can be very different from another domain. Feature selection or reduction does not help either.

In total, we have 66,494 labeled regions generated from 2,310 *user donated* emails. Among these labeled regions, there are 2,740 `Product` regions and 45,637 non-product (`Other`) regions. As noted before, some of these regions are pre-annotated as `Price`, `Date`, `Time`, `Email Address`, `Address`, etc. These are highly accurate for easy fields and we regard them as ground truth throughout the experiments. We only focus on the harder task of detecting product name in the emails. We use this data to train the low-accuracy annotator. In total, we compare 7 different binary classifiers, as illustrated in Table 1.

LogRegL2 Logistic regression with ℓ_2 penalization [20]
LogRegL1 Logistic regression with ℓ_1 penalization [35]
CART Classification and Regression Trees [33]
MARS Multivariate Adaptive Regression Splines [16]
GBM Generalized Boosted Regression Models [36]
RandFor Random Forests [5]
SVM Support Vector Machines [10]

Model selection parameters inherent in the above algorithms are adjusted using cross validation in an automatic fashion.

Table 1: Low-accuracy annotator performance. The estimators are trained on dataset that is not used anywhere else in the rest of the experiments.

model	precision	recall	F1	AUC	runtime
LogRegL2	0.701	0.329	0.448	0.938	9.51
LogRegL1	0.692	0.399	0.507	0.943	11.26
CART	0.701	0.452	0.549	0.834	2.55
MARS	0.727	0.294	0.418	0.949	3.51
GBM	0.780	0.284	0.417	0.950	20.37
RandFor	**0.872**	**0.664**	**0.754**	**0.974**	17.02
SVM	0.846	0.445	0.583	0.935	957.66

`LogRegL2`, `LogRegL1` and `SVM` are linear models, the others are all nonlinear. We observe that `RandFor` provides the best prediction performance while having high computational efficiency. This is not too surprising given that it approximates a nonlinear Pitman estimator in the version space of trees. Consequently we use Random Forests as our low-accuracy region annotator in our work.

5.4 Training the EM-CRF

Based on the low-accuracy annotation we next infer the EM-CRF model as discussed in Section 4. More specifically, the low-accuracy annotator is used to obtain an initial estimate $q_j(y)$ of a region being a product. This allows us to tag the emails for new (or unseen) templates. These are then used to train the EM-CRF model for each template. We proceed by alternating between computing a convex upper bound on the negative log-posterior of the model, i.e. the negative log-likelihood and the Gaussian log-prior, and by invoking an off-the-shelf CRF inference algorithm using the probability estimates $q_j(y)$ for a weighted set of labels.

We perform 4-fold cross validation to test our model. For each test template we sample 75% of its emails and remove their labels to simulate regular unlabeled data. We apply a hand-tuned region annotator to detect specific regions such as `Price, Email Address`, and `Order Number`. For the regions labeled as `Product` or `Other`, we apply the weak learner to generate a probability of this region being labeled as `Product`. After that, we train an EM-CRF model for this template using the weighted (and noisy) labeled emails. Finally, for each template we test the EM-CRF model on the remaining 25% labeled emails.

6. EXPERIMENTAL RESULTS

6.1 Baseline

To illustrate the benefit of the EM-CRF approach over a plain CRF model we also trained a CRF model directly using the low-accuracy annotator's predictions. In this case we used the maximum likelihood annotation of the regions rather than the label probabilities. It is to be expected that this approach will still benefit from the regularity properties of the document. For instance, if in a template the `Product` region is always followed by a `Quantity` region and then a `Price` region, one would expect that the potential $\langle \phi(y_i, y_{i+1}), \theta_{yy} \rangle$ capturing adjacent labels would model this correlation. We call this model **CRF-Max** (since it used the maximum likelihood estimation of the low-accuracy annotator).

A second very natural baseline is to use the Weak Learning (**WL**) directly, i.e. to output just the prediction of the low-accuracy estimator. We expect this to perform the worst.

As has been noted previously, we focus the evaluation on the accuracy of tagging `Product` regions, which is the most difficult one. We use precision, recall and the F1 measure to illustrate overall accuracy.

6.2 Templates with Known Structure

We first focus on analyzing the performance of our approach when the template structure is well known. By template structure we mean which regions in the DOM tree are fixed for all emails in the template and which regions are specific to a given email instance from that template (e.g. **`recipient name`**, **`address`** and **`product`**(s) in the purchase). We use this setup to conduct a detailed analysis of our proposed approach. Large scale experiments with variable structure are reported in Section 6.5.

We used 4 frequent templates of user donated emails, and for any given template we manually engineer a template parser to annotate data with almost perfect precision as fixed or variable. This is a costly exercise. It is not scalable since an engineer needs to analyze HTML code to accomplish this task. Moreover, it requires us to continue investing resources whenever the template changes.

template	1	2	3	4
emails	36	105	50	60
products	78	191	59	231

As can be seen, even the frequent templates only consist of a rather small number of labeled emails.

6.3 Model Comparison

Figure 6 provides a comparison of F1, precision, and recall scores for all three algorithms (WL, CRF-Max and EM-CRF). To render the results comparable we tuned all models for maximum F1 score performance.

- EM-CRF outperforms all other algorithms on all test templates. It provides a 30% improvement relative to the WL annotator.
- The CRF-Max model also improves annotations generated by the WL model, albeit only by 26%. This confirms that training CRFs on noisy labels can lead to a *denoising* estimator.
- Compared to the plain CRF-Max model, EM-CRF is approximately 3.5% better. This shows that the EM iterations are helpful in reducing the noise further.

Particularly, for three out of these four templates, WL's achieved precision is lower than recall, which seems to be inconsistent with the `RandFor` figures in Table 1. This is just because the figures in Table 1 are calculated on the data across all the templates while the performance would be different for each individual template.

Besides these three algorithms compared in Figure 6, we also checked the performance of a CRF trained on the data across all other templates. The F1 performance on all four tested templates is almost 0, caused by the structure inconsistency brought from the huge difference across templates. This supports our motivation of training a CRF per template. Another observation is EM-CRF can still boost from WL although WL's precision is lower than 0.5 (F1 is still higher than 0.5). This is because T4 emails contains many (3.85) products on average, which help CRFs learn from abundant repeated structures.

Figure 7 illustrates the distribution of the F1 score across a range of hyperparameter values. Note that the variance of the EM-CRF model is lower than that of its alternatives. Not only is the model more accurate, it also provides a more reliable performance in large scale experiments where we do not have access to a development set for each template.

Figure 6 illustrates precision and recall when the hyperparameters have been tuned for the best F1 score for all models. Again, we may observe improved performance of CRF-Max and EM-CRF over WL. Compared to EM-CRF, the CRF-Max model occasionally exhibits higher precision but lower recall. This is because CRF-Max is trained directly on the labels predicted by the low-accuracy estimator. For

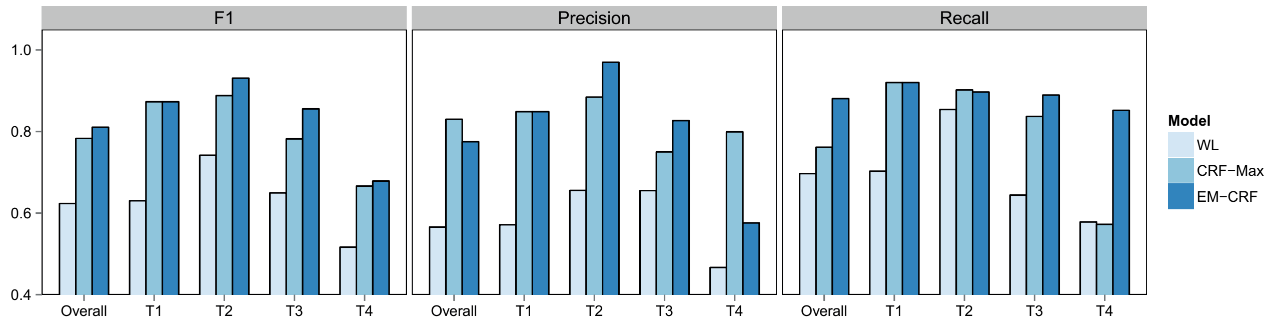


Figure 6: Comparison between the low-accuracy annotator (WL), a conventional conditional random field (CRF-Max), and the proposed algorithm (EM-CRF). We report F1 scores, precision and recall overall and on all four templates (T1 to T4).

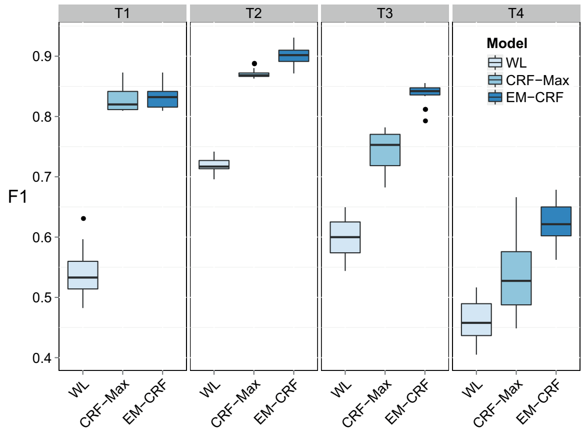


Figure 7: Distribution of F1 performance scores on four templates over a range of hyperparameters.

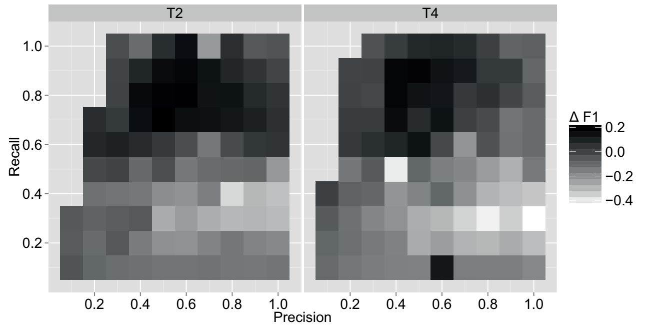


Figure 8: Improvement in F1 score of EM-CRF over WL as a function of precision and recall for templates 2 (left) and 4 (right). Some precision-recall regions are missing since the associated conditions cannot be satisfied.

the `Product` regions, which are correctly labeled by WL, the CRF-Max will train good parameters to predict them. On the other hand, the `Product` regions which are incorrectly labeled as `Other`, CRF-Max will be relatively easily induced to annotate these regions as `Other`. On the contrary, for EM-CRF, when WL provides the wrong prediction on `Product` regions, it just labels it as `Product` with a less-than-half probability. However, this probability will not be ignored during the EM-CRF training. EM-CRF will evaluate the posterior probability of when this region is labeled as `Product`, and usually increases the probability of this region being labeled as `Product` if it finds there is higher structural consistency.

Note that by construction the CRF-Max estimate is what the EM-CRF algorithm obtains in its first iteration with only max-likelihood label sequences. That is, EM-CRF creates a continuum between the decidedly suboptimal annotations of the low-accuracy annotators and the better CRF estimates to a fully self-consistent estimate.

6.4 Ablation Study

Precision and Recall of the low-accuracy estimator. Since EM-CRF is built on data labeled by the low-accuracy WL annotator, it is important to analyze under which conditions the EM-CRF results can improve. To conduct this study we simulated a weak learner with performance that spans the possible range of precision and recall. This can be achieved by oversampling the weight of positive product instances when training the low-accuracy annotators. We then compare the performance difference between EM-CRF and WL to the precision and recall of WL. The results on two large templates (2 and 4) are shown in Figure 8.

As can be seen, the results of both templates are qualitatively similar. That is, improvements are highest for WL whenever the recall is sufficiently high. This is quite possibly due to the fact that we use conditional independence assumptions on the weak learners, i.e. their estimates enter as products of probabilities $\prod_i \pi_{ij}$. Hence, poor recall immediately leads to a strong bias against the presence of a given attribute. A more refined model to capture correlation between the weak learners would probably ameliorate these issues. By and large, for over 50% recall improvements are pervasive.

Number of candidates in the E-Step. In (10) the posterior probability of each possible label combination y for the current document x_j needs be calculated. A potential problem here is that when there are many uncertainties over the labeling of regions in one email, as there could be exponentially large number of possible label combinations (2^N), which much reduces the efficiency of training. In our solution, we pick the top K candidate sequence labels with largest posterior probability in the step of recalculating distribution $q_j(y)$. This reduces the time of the costly M-step (7) significantly.

This raises the question as to how much the hyperparameter K influences the EM-CRF performance? We train models with K ranging from 1 to 500 and evaluate the EM-CRF performance on each template for each K, as shown in Figure 9. As can be seen, the F1 measure overall improves, albeit it does not change dramatically with varying K. This suggests that small values of K are perfectly acceptable. We finally set $K = 100$.

Regularization of CRF weights. We tune the L_2 regularization penalty during the training of the CRF given the current belief on each label combination $q_j(y)$. That is, we add $\lambda \|\theta\|^2$ to the negative log-likelihood in (7). Figure 10

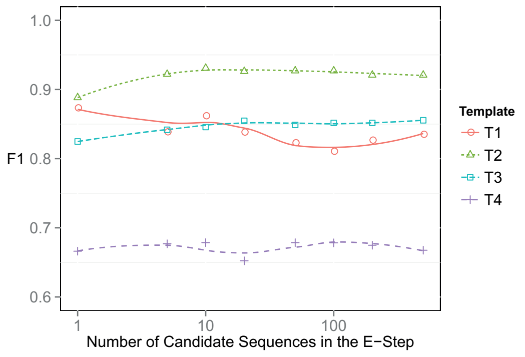


Figure 9: F1 as a function of the number of candidate sequences in the E-step. As can be seen, it is highly robust to sparse approximations.

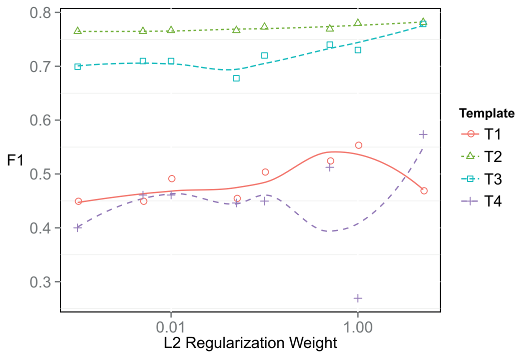


Figure 10: F1 as a function of the L2 regularization parameters. More regularization improves accuracy.

show the accuracy as a function of the regularization parameter for $\lambda \in [10^{-3}, 10]$. As expected, increasing regularization improves generalization up to some point, since it restricts the model class. Beyond that, accuracy decreases again due to underfitting. Empirically, we set L2 regularization as 10^{-2} for the large scale experiments below.

6.5 Templates with Inferred Structure

In this section we demonstrate the behavior of our method by a larger scale evaluation study on 200 templates. The number of emails in each template ranges from 10 to 2500 emails.

We used simple frequency counting with exact textual match as simple template extraction method. Those fixed regions are labeled as `Fixed` during our CRF analysis, however, their textual context is used to as feature for the CRF based models. We use the low-accuracy WL estimator from Section 5.3 which was trained over a separate dataset as our weak learner. The training method proceeds as discussed earlier over each template separately. Once we learn a structural model for each template, we apply this model to predict product regions in new and unseen emails under the same template. We compare the performance of the three methods: WL, CRF-Max and EM-CRF. Those unseen emails are donated with user consent and as such we hand-labeled them strictly for evaluation purposes. During training of the CRF-based models, no labels are needed and as such we do not perform any manual tuning of the hyperparameters and we rather fix them, across all templates, to default values acquired in the previous section. The training of EM-CRF is efficient because in practice we find the performance gets convergence within 10 EM iterations. The average real runtime on a single machine is 1.6s for the large templates studied in this section.

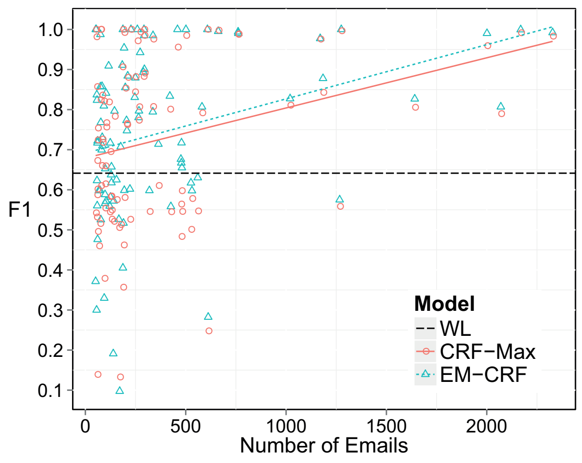


Figure 11: F1 performance as a function of the number of unlabeled emails in a given template.

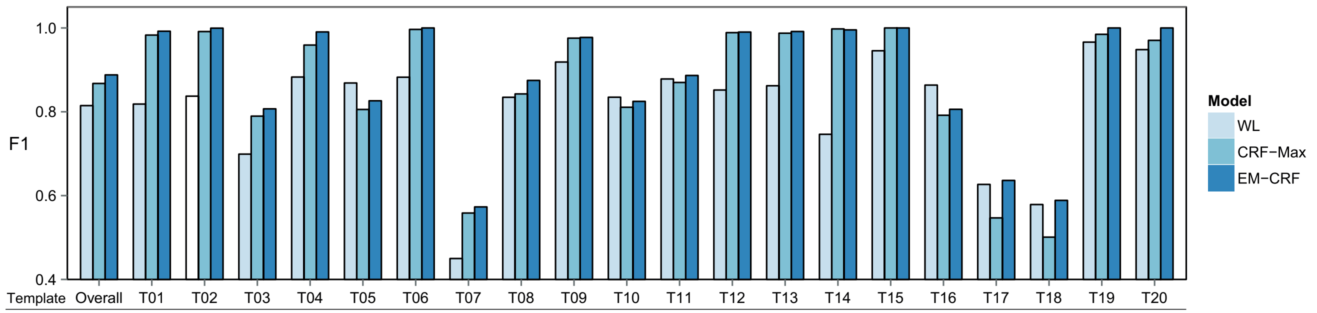
We first compare the performance of the three methods as a function of the number of emails in each template. As show in Figure 11, both EM-CRF and CRF-Max provide a higher overall performance than WL, which verifies the effectiveness of structural information extraction on large-scale data. Furthermore, as the template email number increases, the improvement over WL becomes more significant. This is reasonable since more emails help CRF models learn the template structure pattern better. Moreover, EM-CRF results in higher improvement for templates with large number of emails ($>$ 1000 emails) while its performance is similar to the CRF-Max model for templates with small number of emails. Thus one can use CRF-Max for cold start situations (new templates) and then switch to using EM-CRF for large templates (with over 1000 emails). We further focus on the performance of the largest 20 templates and give a detailed break-down of the performance in Figure 12. The performance break-down over templates with small number of emails is similar to the one shown in Figure 6 and is omitted due to space constraints.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we addressed the challenging problem of information extraction from commercial emails. While supervised information extraction itself is a relatively well-studied area, information extraction from emails presents distinct privacy challenges that prevent us from even looking at the emails to either annotate fields for training or tune the hyper-parameters of the learning method.

We tackled this problem using a fully automatic approach that requires almost no manual tuning and is unsupervised in nature. The approaches presented in this paper (EM-CRF and CRF-Max) make use of low-accuracy annotators trained using weak features on a separate dataset. Our methods leverage those weak signals to learn, in an unsupervised fashion, structural patterns specific to each template and then use those patterns to extract information from future emails in the same template. We provided a thorough evaluation of our approach over large scale emails and showed that our proposed approaches result in significant performance improvement. Moreover, we analyzed the model and data conditions which could lead large improvement.

In the future we plan to investigate joint approaches for template extraction and structural learning as well as exploring the efficacy of transfer learning approaches of structural patterns across different email templates.



Template	Overall	T01	T02	T03	T04	T05	T06	T07	T08	T09	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20
Emails	22,566	2,328	2,169	2,070	2,001	1,641	1,275	1,267	1,184	1,170	1,021	1,020	762	761	662	608	581	560	526	501	459
Products	39,035	2,336	2,171	2,102	2,027	4,694	1,275	2,574	3,376	1,224	4,777	3,781	765	769	874	608	2,627	565	530	501	459

Figure 12: F1 performance on the 20 large email templates and the email & product numbers of each template.

Acknowledgements.

We sincerely thank Andrei Broder for helpful advices and support for the project. Also, we thank the team and collaborators for the excellent work.

8. REFERENCES

[1] N. Ailon, Z. S. Karnin, E. Liberty, and Y. Maarek. Threading machine generated email. In *WSDM*. ACM, 2013.

[2] S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2), 2000.

[3] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In *Workshop on Mining software repositories*, pages 137–143. ACM, 2006.

[4] E. Blanzieri and A. Bryl. A survey of learning-based techniques of email spam filtering. *Artificial Intelligence Review*, 29(1):63–92, 2008.

[5] L. Breiman. Random forests. *Machine learning*, 45(1), 2001.

[6] J. D. Brutlag and C. Meek. Challenges of the email domain for text classification. In *ICML*, pages 103–110, 2000.

[7] D. Buttler, L. Liu, and C. Pu. A fully automated object extraction system for the world wide web. In *ICDCS*, 2001.

[8] C.-H. Chang and S.-C. Lui. Iepad: information extraction based on pattern discovery. In *WWW*, pages 681–688. ACM, 2001.

[9] G. V. Cormack. Email spam filtering: A systematic review. *Foundations and Trends in Information Retrieval*, 1(4), 2007.

[10] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[11] M. Craven, J. Kumlien, et al. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86, 1999.

[12] V. Crescenzi, G. Mecca, P. Merialdo, et al. Roadrunner: Towards automatic data extraction from large web sites. In *VLDB*, volume 1, pages 109–118, 2001.

[13] N. Dalvi, R. Kumar, and M. Soliman. Automatic wrappers for large scale web extraction. *Proceedings of the VLDB Endowment*, 4(4):219–230, 2011.

[14] J. Diesner, T. L. Frantz, and K. M. Carley. Communication networks from the enron email corpus "it's always about the people. enron is no different". *Computational & Mathematical Organization Theory*, 11(3):201–228, 2005.

[15] D. W. Embley, Y. Jiang, and Y.-K. Ng. Record-boundary discovery in web documents. *ACM SIGMOD Record*, 28(2):467–478, 1999.

[16] J. H. Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.

[17] K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar. Posterior regularization for structured latent variable models. *JMLR*, 99:2001–2049, 2010.

[18] S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm. Dom-based content extraction of html documents. In *WWW*, 2003.

[19] K. Hall, R. McDonald, J. Katz-Brown, and M. Ringgaard. Training dependency parsers by jointly optimizing multiple objectives. In *EMNLP*, pages 1489–1499, 2011.

[20] T. Hastie, R. Tibshirani, and J. J. H. Friedman. *The elements of statistical learning*, volume 1. Springer New York, 2001.

[21] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*, 2011.

[22] R. Horst and N. V. Thoai. Dc programming: overview. *Journal of Optimization Theory and Applications*, 103(1):1–43, 1999.

[23] R. Jin and Z. Ghahramani. Learning with multiple labels. In *NIPS*, pages 897–904, 2002.

[24] S. Kiritchenko and S. Matwin. Email classification with co-training. In *CASCON*, pages 301–312. IBM Corp., 2011.

[25] A. Klementiev and D. Roth. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *ACL*, pages 817–824, 2006.

[26] B. Klimt and Y. Yang. The enron corpus: A new dataset for email classification research.

[27] A. Kulkarni and T. Pedersen. Name discrimination and email clustering using unsupervised clustering and labeling of similar contexts. In *IICAI*, pages 703–722, 2005.

[28] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[29] H. Li, D. Shen, B. Zhang, Z. Chen, and Q. Yang. Adding semantics to email clustering. In *ICDM*, 2006.

[30] B. Liu, R. Grossman, and Y. Zhai. Mining data records in web pages. In *KDD*, pages 601–606. ACM, 2003.

[31] G. S. Mann and A. McCallum. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *JMLR*, 11:955–984, 2010.

[32] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL*, pages 1003–1011, 2009.

[33] L. B. J. F. R. Olshen and C. J. Stone. Classification and regression trees. *Wadsworth International Group*, 1984.

[34] M. Paşca. Weakly-supervised discovery of named entities using web search queries. In *CIKM*, pages 683–690. ACM, 2007.

[35] P. Ravikumar, M. J. Wainwright, and J. D. Lafferty. High-dimensional ising model selection using l1-regularized logistic regression. *The Annals of Statistics*, 38(3), 2010.

[36] G. Ridgeway. Generalized boosted regression models. *Documentation on the R Package 'gbm', version*, 1(5):7, 2006.

[37] S. Riedel, L. Yao, and A. McCallum. Modeling relations and their mentions without labeled text. In *ECML*, 2010.

[38] R. Rowe, G. Creamer, S. Hershkop, and S. J. Stolfo. Automated social hierarchy detection through email network analysis. In *WebKDD and SNA-KDD*, pages 109–117. ACM, 2007.

[39] A. M. Rush. A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing. 2012.

[40] A. M. Rush, D. Sontag, M. Collins, and T. Jaakkola. On dual decomposition and linear programming relaxations for natural language processing. In *EMNLP*, pages 1–11, 2010.

[41] C.-Y. Tseng, J.-W. Huang, and M.-S. Chen. Promail: using progressive email social network for spam detection. In *Advances in Knowledge Discovery and Data Mining*. 2007.

[42] S. Yoo, Y. Yang, F. Lin, and I.-C. Moon. Mining social networks for personalized email prioritization. In *KDD*, 2009.

[43] S. Youn and D. McLeod. A comparative study for email classification. In *Advances and Innovations in Systems, Computing Sciences and Software Engineering*. 2007.

[44] Y. Zhai and B. Liu. Web data extraction based on partial tree alignment. In *WWW*, pages 76–85. ACM, 2005.

[45] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. 2d conditional random fields for web information extraction. In *ICML*, pages 1044–1051. ACM, 2005.

[46] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. Simultaneous record detection and attribute labeling in web data extraction. In *KDD*, pages 494–503. ACM, 2006.

[47] J. Zhu, Z. Nie, B. Zhang, and J.-R. Wen. Dynamic hierarchical markov random fields for integrated web data extraction. *JMLR*, 9:1583–1614, 2008.

[48] X. Zhu. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison*, 2:3, 2006.

[49] X. Zhu and A. B. Goldberg. *Introduction to Semi-supervised Learning*. Number 6. Morgan & Claypool Publishers, 2009.