Mixture Model based Label Association Techniques for Web Accessibility

Muhammad Asiful Islam, Yevgen Borodin, I.V. Ramakrishnan
Department of Computer Science, Stony Brook University
Stony Brook, NY, USA
{maislam, borodin, ram}@cs.sunysb.edu

ABSTRACT
An important aspect of making the Web accessible to blind users is ensuring that all important web page elements such as links, clickable buttons, and form fields have explicitly assigned labels. Properly labeled content is then correctly read out by screen readers, a dominant assistive technology used by blind users. In particular, improperly labeled form fields can critically impede online transactions such as shopping, paying bills, etc. with screen readers. Very often labels are not associated with form fields or are missing altogether, making form filling a challenge for blind users. Algorithms for associating a form element with one of several candidate labels in its vicinity must cope with the variability of the element’s features including label’s location relative to the element, distance to the element, etc. Probabilistic models provide a natural machinery to reason with such uncertainties. In this paper we present a Finite Mixture Model (FMM) formulation of the label association problem. The variability of feature values are captured in the FMM by a mixture of random variables that are drawn from parameterized distributions. Then, the most likely label to be paired with a form element is computed by maximizing the log-likelihood of the feature data using the Expectation-Maximization algorithm. We also adapt the FMM approach for two related problems: assigning labels (from an external Knowledge Base) to form elements that have no candidate labels in their vicinity and for quickly identifying clickable elements such as add-to-cart, checkout, etc., used in online transactions even when these elements do not have textual captions (e.g., image buttons w/o alternative text). We provide a quantitative evaluation of our techniques, as well as a user study with two blind subjects who used an aural web browser implementing our approach.

Keywords: Mixture models, context, web forms, web accessibility, screen reader, aural web browser, blind user

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Algorithms, Design, Human Factors

1. INTRODUCTION
Evolution of the Web from single-author text-based web pages to interactive web applications with user-generated content is making it less accessible to people with vision impairments and blindness, a population that is large and growing ever larger.

The dominant assistive technology for non-visual web access used by people with vision impairments is typically represented by screen readers and audio browsers (e.g. JAWS: www.freedomscientific.com, Windows-Eyes: www.gwmicro.com, Apple’s VoiceOver: www.apple.com/accessibility/voiceover) that convert text to speech, reading aloud all the textual content on the web page. Screen-reader users are forced to process web content sequentially through a serial audio interface. Screen readers work well only on web pages that contain accessibility metadata complying with web accessibility guidelines (www.w3.org/TR/WCAG10/). However, accessibility metadata is often inadequate in both quality and quantity. As a result, blind users encounter numerous accessibility problems from having to listen to irrelevant content to not being able to complete their tasks at all because they cannot access information, fill forms, etc.

Most screen readers have shortcut-driven navigation, i.e. users can press arrow keys to navigate between web page elements or use element-specific navigation keys to navigate among elements of a particular type, e.g., [Shift+]E for editable fields, [Shift+]B for buttons, etc. The user can also press [Shift+]Tab to navigate among focusable elements, which include links and form elements. Pressing Enter on the form field typically switches screen-readers into the edit mode, pressing Escape switches back to browsing mode.

Being able to edit forms with the screen reader is an important aspect of web accessibility. To fill a form field, the screen-reader user has to hear its label, type, and value together, e.g., on the airline reservation website in Figure 1, the user may hear “From textbox NYC.” This requires that correct associations be made between form elements and their labels. For sighted users the association problem is a non-issue, as they can immediately see the labels near most form fields or guess what has to be filled into the unlabeled fields from context, other visual information, or experience. However, for blind users, this association has to be done by the screen-reader.
Screen readers typically use the HTML <label for…> tag to associate labels with the corresponding form elements. To illustrate, the textbox in the HTML snippet shown in Figure 3(a) has the label tag, while the checkbox in Figure 3(b) does not. In fact, in the dataset of 542 forms that we collected only 11% of the 7800 form fields had label tags. Some screen readers as well as VoiceOver exclusively depend on the label attribute to read out the captions associated with form elements. Others try to heuristically guess the label by taking the text that precedes a form element in the HTML DOM (www.w3.org/DOM/DOMTR). They often make mistakes because what is visually to the left of the form element is not always the label, or does not coincide with the preceding element in the DOM tree. Observe from Figure 2 that the order in which the labels in Figure 1 will be read aloud is: “Adults”, “Children”, “Seniors”, “Infants”, “Select”,… which is clearly confusing and disorienting to the user who is trying to guess the association between the spoken label and the corresponding Select list.

If the label tag is not there, the user has to explore the elements around the form field and attempt to infer its label. Since forms typically contain a number of fields separated by labels, screen-reader users often get confused trying to figure out whether a text label refers to the preceding or following field. It gets even harder if some form fields are missing labels altogether. Finally, looking around for a label is also inconvenient, especially when using the Tab key to navigate in the edit mode. Specifically, the user has to first come out of the edit mode, look for a label, and then go back into the edit mode to fill out the form field.

It is obvious that assigning the correct labels to form fields is critical for non-visual web browsing. This problem is not only important but also has subtleties that make it technically interesting. The question is: how do we select the correct label from among several others in the locality of the form element?

Observe in Figure 1 and Figure 4(a), 4(b) the differences in the structure and content of forms from different sites. Notice the different locations where “From” and “To” are placed; also notice how distances between the labels and the corresponding form elements differ in different sites. Such variations imply that algorithms for the label association problem must reason in the presence of uncertainty.

Parameterized probabilistic models provide an elegant framework for modeling and reasoning in the presence of uncertainty. A parameterized probability distribution function (e.g. Gaussian) models the uncertainty and its parameters are computed by maximizing the likelihood of observed data that is assumed to be generated by an underlying random process.

In our label association problem the locations of the labels relative to the form elements, their distances to the form elements, etc., constitute generic features of a form. Each such feature is represented by a random variable. Some are discrete (e.g. label placements – top, left, etc.) while others are continuous (e.g. distances).
Finite Mixture Models (FMMs) [1] have been developed to model random processes characterized by mixture of discrete and continuous random variables. In FMMs a collection of different parameterized distribution functions are composed to accurately reflect the behavior of the underlying random process. FMM extends Gaussian Mixture Model [12] based clustering beyond continuous random variables to include discrete random variables. In this paper we present a solution to the label association problem based on FMMs.

Summary of our Contributions

• We have developed a probabilistic formulation for the label association problem in terms of FMMs. In the formulation the form’s features are represented by random variables, feature values denote observable data and the number of labels corresponds to mixture components. An algorithm based on maximizing the likelihood of observed data using the Expectation Maximization (EM) algorithm [2] assigns the most likely label to a form element. (Sec. 3.1)

• To address the problem of form elements with missing labels we introduce the notion of context for a form element, which essentially consists of the words in the “locality” of the element. Using context information we show how to assign labels to such elements from an external Knowledge Base (KB) of labels. (Section 3.2)

• We adapt the idea of context to classify and label transaction elements, such as "add to cart" and "checkout", so that they can be quickly identified even when they do not have textual captions. (Section 4)

• We have incorporated these techniques in our HearSay aural web browser [11]. We have conducted quantitative evaluation of the techniques (Section 5) and have done a limited user study with two blind users (Section 6). The quantitative experiments show that our FMM-based approach correctly labels nearly 95% of form elements without the HTML <label for ..> tag. The user study suggests that it also makes form filling much more usable. See the supplementary video accompanying the paper showing a blind user demonstrating our aural browser vis-à-vis JAWS screen reader and VoiceOver.

The form labeling problem has many other applications beyond Web accessibility (e.g. web data integration). In the database community solutions to the form labeling problem ranging from manually constructed rule-based approaches to using classification technology have been reported [17,18,19,20]. While a detailed comparison of these works appears in the related work (Section 7) suffice it is to say here that there are quite a few differences between them and our work. Firstly, they focus exclusively on data integration whereas we approach it as a human computer interaction problem. Secondly, our approach captures the randomness that is intrinsic to the problem and provides experimental evidence of the merits of the approach in terms of performance and ability to control labeling errors - a problem not addressed in the aforementioned data integration works. Lastly, our approach deals with missing captions – an important issue not dealt with in these works.

2. PRELIMINARIES

We will be using the following notions: A web page is made up of several elements (links, forms, buttons, etc.). We assume each such web element has a unique (x,y) pixel coordinate, which we will refer to as its address. Addresses one can compute “closest” elements to a given web element e. We say two web elements are neighbors if they are assigned consecutive numbers in a preorder traversal. Text elements in a web page are leaf nodes in the page’s DOM tree whose attributes are text strings - e.g. the four text elements annotated “Adults”, “Children”, “Seniors”, and “Infants” respectively in Figure 2. We will use text elements and (their associated) text strings interchangeably.

A (X)HTML form consists of a set of form elements (e.g., 'textbox', 'select-list', 'radio button', 'checkbox', text elements, etc.). We use the terminology labels to refer to the text elements appearing in a form. By locality of a form element fe we mean the set of text element(s) delimited by the neighboring form elements of fe. In Figure 2, the leftmost select-list’s locality is {“Adults”, “Children”, “Seniors”, “Infants”}.

We define the context of a form element fe as a set consisting of its attribute values as well as the closest text element(s) in its locality. E.g. the context for the 2nd select list annotated “Economy” in Figure 5 is {“cabin”, “economy”, “business”, “first”}. Note there are no text elements in the locality of this select list.

![Figure 5: HTML specification for the 2nd select list.](image)

The normalized longest common subsequence between two strings σ₁, σ₂, denoted norm-LCS, is defined as (see [4]):

\[
\text{norm-LCS}(\sigma_1, \sigma_2) = \frac{\text{LCS}(\sigma_1, \sigma_2)}{\text{min}(\text{length}(\sigma_1), \text{length}(\sigma_2))}
\]

Its range is [0,1].

We associate a feature vector with every form element. Towards defining it, let F = {f₁, ..., fₙ} be the set of form elements of a web form f and C = {c₁, ..., cₘ} be the set of labels (i.e. text elements) appearing in f; fᵢ denotes the iᵗʰ form element and cᵦ denotes the kᵗʰ label (1 ≤ i ≤ n; 1 ≤ k ≤ m). With every (fᵢ, cᵦ) pair we associate a feature vector dᵢₗ:

\[
\text{sim}(i,k), \text{prox}(i,k), \text{place}(i,k), \text{align}(i,k), \text{type}(i), \text{font}(k)>
\]

where:

\[
\text{sim}(i,k) = \text{norm-LCS}(cᵦ, \text{the value of } fᵢ \text{'s name attribute})
\]

The name attribute takes a text string value, e.g., ‘cabin’ in Figure 5.

\[
\text{prox}(i,k) \text{ is the inverse of the distance between } fᵢ \text{ and } cᵦ;
\]
place(i,k) is the relative placement of \( c_k \) with respect to \( f_i \).
It can take one of \{TOP, LEFT, RIGHT, BOTTOM, TOPLEFT, BOTTOMLEFT, TOPRIGHT, BOTTOMRIGHT\}. This is readily computed from the addresses of \( c_k \) and \( f_i \), (e.g. TOP corresponds to \( y_i < y_i - \) the y coordinates of the elements);

\( \text{align}(i,k) \) takes the value HORIZONTAL (VERTICAL) to denote horizontal (vertical) geometric alignment between \( f_i \) and \( c_k \) within some small pixel margin \( \varepsilon \). The value NONE is assigned when neither holds.

\( \text{type}(i) \) is the type of \( f_i \) (e.g., TEXTBOX, SELECTION LIST, RADIO BUTTON, CHECKBOX);

\( \text{font}(k) \) is the font type of \( c_k \).

For illustration let’s consider the login form in Figure 6. For brevity we will drop the \( \text{align} \) and \( \text{font} \) components.

The labels are \( c_1 = "Username", c_2 = "Password" \) and the two textbox elements are \( f_1 \) and \( f_2 \) denoting top and bottom elements resp. \( f_2 \) ’s name attribute value is “Password”.

![Login Form and its HTML fragment](image)

**Figure 6:** Login Form and its HTML fragment

\( \text{sim}(2,2) \) is \( \text{norm}_LCS("Password","Password") = 6/6 = 1 \);
\( \text{prox}(2,2)=1; \text{place}(2,2)=\text{LEFT}, \text{type}(2)=\text{TEXTBOX} \). Hence,

\[
d_{22} = <1.0,1.0,\text{LEFT},\text{TEXTBOX} >. \text{Similarly},
\]

\[
d_{21} = <0.16,0.33,\text{TOP},\text{TEXTBOX} >. \text{Similarly},
\]

\[
d_{11} = <0.66,1.0,\text{LEFT},\text{TEXTBOX} >, \text{and}
\]

\[
d_{12} = <0.11,0.33,\text{BOTTOM},\text{TEXTBOX} >
\]

We use the term \( \text{pattern} \) to refer to the substring formed by the sequence of categorical feature values \(<\text{place}(i,k),\text{align}(i,k),\text{type}(i),\text{font}(k)\>). \text{E.g.},

\[
<\text{LEFT,HORIZONTAL,TEXTBOX,BOLD}> \text{is the pattern in } d_{11}
\]

For our probabilistic formulation we assume that the underlying parameterized probability distribution for the continuous random variables is the standard multivariate Gaussian:

\[
a(u|\mu, \Sigma) = \frac{1}{(2\pi)^{\text{dim}/2}|\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2}(u - \mu)^T \Sigma^{-1} (u - \mu) \right\}
\]

\( \mu, \Sigma \) are the two parameters denoting the mean and covariance respectively; \( \text{dim} \) is the dimension of vector \( u \); \( Z^T \) is the transpose of vector \( z \). Parameters of distributions are referred to as \( \text{model} \) parameters. A random variable is associated with each feature value; those corresponding to \( \text{sim} \) and \( \text{prox} \) are continuous random variables while \( \text{place}, \text{align}, \text{type} \) and \( \text{font} \) are discrete random variables. We will often refer to random variables as simply variables.

3. ALGORITHMS FOR LABEL ASSOCIATION

3.1 Finite Mixture Model Formulation

A finite mixture model (FMM) consists of: a collection \( \Omega = \{\omega_1, \omega_2, \ldots, \omega_t\} \) of parameterized probability distributions; a set \( \theta \) consisting of their model parameters; and a set \( C = \{c_1, \ldots, c_m\} \) of mixing components. Let \( D = \{d_1, d_2, \ldots, d_N\} \) be a set of independent and identically distributed data points. The likelihood of data \( d_i \in D \) is the sum of probabilities over all mixture components [1].

\[
p(d_i; \theta) = \sum_{k=1}^{M} p(c_k)p(d_i|c_k; \theta)
\]

(2)

\( p(c_k) \) is the prior probability of picking the \( k^{th} \) component, \( p(d_i|c_k; \theta) \) is the probability of \( d_i \) conditioned on the \( k^{th} \) component and \( \sum_{k=1}^{M} p(c_k) = 1 \). By Bayes theorem, the posterior probability or the probability that the \( k^{th} \) component is responsible for generating the data point \( d_i \) is:

\[
p(c_k|d_i; \theta) = \frac{p(c_k)p(d_i|c_k; \theta)}{p(d_i; \theta)}
\]

(3)

In our label association problem: \( c_k \) denotes the \( k^{th} \) label; \( i \) denotes the \( i^{th} \) form element; \( d_i = \{d_{i1}, d_{i2}, \ldots, d_{ik}, \ldots, d_{iM}\} \) where \( d_{ik} \) corresponds to the feature vector constructed from the \( i^{th} \) form element and the \( k^{th} \) label, and \( p(c_k|d_i; \theta) \) is the probability (or likelihood) of the \( k^{th} \) label being associated with the \( i^{th} \) form element on “observing” its features. We represent each feature vector \( d_{ik} = (u_{ik}, v_{ik}) \), where \( u_{ik} \) denotes the continuous feature values \( \text{sim} \) and \( \text{prox} \) and \( v_{ik} \) denotes the categorical feature values \( \text{place}, \text{align}, \text{type} \) and \( \text{font} \). We use the multivariate Gaussian distribution (equation (1) in Section 2) for the continuous valued features \( u_{ik} \). For brevity of notation we will sometimes omit \( c_k \) from \( p(d_{ik}|c_k; \theta) \) since the subscript \( k \) in \( d_{ik} \) implies the \( k^{th} \) label. Now

\[
p(d_{ik}|\theta) = p(u_{ik}, v_{ik}|\theta)
\]

(4)

To simplify the computation of probabilities involving mixture of continuous and discrete feature values one can assume that they are independent. But this is not the case in our label association problem. For instance, labels for radio buttons are usually placed to the right whereas they appear on the top for select-lists. We have also observed that proximity and placements are correlated. Such strong independence assumptions can be avoided by using the location model [3]. Let’s explain this model with the following example.

Suppose we have two categorical variables corresponding to \( \text{type} \) and \( \text{place} \) respectively and the two continuous variables corresponding to \( \text{sim} \) and \( \text{prox} \). The \( \text{type} \) variable can take four distinct values (TEXTBOX, SELECT LIST, RADIO BUTTON, CHECKBOX) and the \( \text{place} \) variable takes eight distinct values (TOP, LEFT, RIGHT, etc). In the location model, all the categorical variables are re-
placed by one new categorical variable. This new categorical variable can take thirty two distinct patterns such as \(<LEFT, TEXTBOX>, <TOP, SELECT>, etc. We will denote this new categorical random variable by \(X\) and any pattern value that it takes by \(x_j (1 \leq j \leq P)\).

Suppose the pattern value in \(d_{ik}\) is \(x_i\). So \(p(x_i)\) is \(p(X = x_i)\) and \(p(u_{ik}|v_{ik}; \theta)\) is \(p(u_{ik}|X = x_i; \theta)\). Since \(u_{ik}\) is a multivariate Gaussian \(p(u_{ik}|X = x_i; \theta) = \phi(u_{ik}|\mu_x, \Sigma; \theta)\). From equation (4) the posterior probability of equation (3) becomes:

\[
p(c_k|d_{ik}; \theta) = \frac{p(c_k)p(d_{ik}|c_k; \theta)}{\sum_{l=1}^{N} p(c_l)p(d_{il}|c_l; \theta)}
\]

\[
p(x_i) = \frac{\sum_{k=1}^{K} \delta_{ik} p(c_k|d_{ik}; \theta)}{\sum_{k=1}^{K} \sum_{l=1}^{L} \delta_{lk} p(c_k|d_{ik}; \theta)}
\]

Where, \(\delta_{ik} = \begin{cases} 
1, & \text{if } d_{ik} \text{ contains the } s^{th} \text{ pattern} \\
0, & \text{otherwise}
\end{cases}\)

\[
\mu_x = \frac{\sum_{k=1}^{K} \sum_{i=1}^{N} \delta_{ik} p(c_k|d_{ik}; \theta) u_{ik}}{\sum_{k=1}^{K} \sum_{i=1}^{N} \delta_{ik} p(c_k|d_{ik}; \theta)}
\]

\[
\Sigma = \frac{\sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{j=1}^{N} \delta_{ik} \delta_{ij} p(c_k|d_{ik}; \theta)(u_{ik} - \mu_x)(u_{ij} - \mu_x)^T}{\sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{j=1}^{N} \delta_{ik} \delta_{ij} p(c_k|d_{ik}; \theta)}
\]

Intuitively these equations say that mixing proportion \(p(c_k)\) is the average posterior probability \(p(c_k|d_{ik}; \theta)\), \(p(x_i)\) is the fraction of data points that match pattern \(x_i\) and the mean \(\mu_x\) (covariance \(\Sigma\)) is a weighted mean (covariance) of all the data points matching \(x_i\).

Notice that to compute \(p(c_k|d_{ik}; \theta)\) using equation (5) we need to estimate the model parameters using equation (6), (7), (8), and (9). However, they in turn depend on the posterior probability \(p(c_k|d_{ik}; \theta)\). The method used to compute the model parameters using equations (5) to (9) is the classic iterative Expectation-Maximization (EM) algorithm [2]. AssociateLabel is the high-level pseudo-code for associating labels with form elements based on EM. The E-step and the M-step in the algorithm are the Expectation and Maximization steps respectively.

Algorithm AssociateLabel

Input: An unlabeled web form

Output: The set of <form element, its label> pairs.

1. Construct data \(D\) from web form elements and their candidate labels. These labels are the text elements in the form.

2. Initialize the model parameters \(\theta\) arbitrarily.

3. Repeat Steps 3.1 and 3.2 while model parameters improve, as measured by the change in log-likelihood \(-\log L(D|\theta)\).

3.1 (E-Step) Use the current parameters \(\theta\) to estimate label association for each unlabeled form element, i.e. \(p(c_k|d_{ik}; \theta)\) using equation 5.

3.2 (M-Step) Given the estimated label probability for each form element, re-estimate the model parameters \(\theta\) using equations 6, 7, 8, 9.

4. Assign to each form element \(f_i\) the label \(c_k\) that has the highest \(p(c_k|d_{ik}; \theta)\) value.

Improving Accuracy of the Association Algorithm

AssociateLabel is an unsupervised (i.e. fully automatic) algorithm as no training data was used in making the label associations. In our experiments we observed that the algorithm can create ambiguities in the label assignment because of “almost identical” probabilities computed for competing candidate labels. One way to resolve many such ambiguities is to estimate the prior probabilities in Step 2 instead of initializing them with arbitrary values. These priors can be estimated from labeled training data using simple frequency counts. Each form in the labeled training set consists of form elements and their associated text elements. Suppose we had ten textboxes with seven having their associated text elements to their left. So the prior probability of \(<LEFT, TEXTBOX>\) is 0.7.

AssociateLabel in Action: An illustrative example

We will use the login form in Figure 6. For ease of exposition we will illustrate using the three patterns values \(\sigma_1=<LEFT, TEXTBOX>, \sigma_2=<TOP, TEXTBOX>\) and \(\sigma_3=<BOTTOM, TEXTBOX>\) with initial probabilities \(p(X=\sigma_1)=1/4\); \(p(X=\sigma_2)=1/4\); \(p(X=\sigma_3)=1/4\). Note that \(\mu_x\) is a pair of \(<\text{sim}, \text{prox}\>\) values. Let \(\mu_{x=\sigma_1} = [0.75, 0.80]\); \(\mu_{x=\sigma_2} = [0.50, 0.55]\); \(\mu_{x=\sigma_3} = [0.50, 0.55]\) and the covariance matrix \(\Sigma = [\Sigma_{11}; \Sigma_{21}; \Sigma_{22}] = [0.25, 0.0; 0.0, 0.30]\).

\(c_1\) is “Username” and \(c_2\) is “Password”. Initially \(p(c_1)=p(c_2)=1/2\). In the E-step using (3) we compute

\[
p(c_{1}|d_{11}; \theta) = \frac{p(c_{1})p(v_{11})p(u_{11}|v_{11}; \theta)}{\sum_{k=1}^{K} p(c_k)p(v_{11})p(u_{11}|v_{11}; \theta)}
\]

\[
= \frac{\frac{1}{2} \times \frac{1}{4} \times \phi(0.66, 1.0)|0.75, 0.80|, 0.25, 0.0, 0.0, 0.30)}{\frac{1}{2} \times \frac{1}{4} \times \phi(0.66, 1.0)|0.75, 0.80|, 0.25, 0.0, 0.0, 0.30)}
\]

\[
+ \frac{1}{2} \times \frac{1}{4} \times \phi(0.11, 0.33)|0.50, 0.55|, 0.25, 0.0, 0.0, 0.30)
\]

\[
= 0.6722
\]

Similarly \(p(c_{2}|d_{12}; \theta), p(c_{1}|d_{21}; \theta)\) and \(p(c_{2}|d_{22}; \theta)\) evaluate to \(0.3278, 0.3268\) and \(0.6732\) resp.

In the M-step we use equations (6), (7), (8) and (9) to re-estimate the model parameters.

\[
p(c_{1}) = \frac{\sum_{k=1}^{K} p(c_k|d_{11}; \theta)}{N} = 0.6722 + 0.3278 = \frac{1}{2}
\]
Similarly, \( p(c_2) \) evaluates to 1/2.

Using equation (7),

\[
p(X = \sigma_1) = \frac{0.6722 + 0.6732}{0.6722 + 0.3278 + 0.6722 + 0.3268} = 0.6727
\]

and \( p(X = \sigma_2) \), \( p(X = \sigma_3) \) evaluate to 0.134, 0.1639 resp.

Using equation (8),

\[
\mu_{X = \sigma_2} = \frac{0.6722 \times [0.66, 1.0] + 0.6732 \times [1.0, 1.0]}{0.6722 + 0.6732} = [0.83, 1.0]
\]

Similar evaluation results in \( \mu_{X = \sigma_3} = [0.16, 0.33] \) and

\[
\mu_{X = \sigma_4} = [0.11, 0.33].
\]

Using equation (9), \( \sum = [0.0289, 0.0; 0.0, 0.01] \).

The algorithm converges after three iterations and we get \( p(c_1|d_{11}; \theta) = 0.804, p(c_2|d_{12}; \theta) = 0.196, p(c_3|d_{21}; \theta) = 0.192, \) and \( p(c_4|d_{22}; \theta) = 0.808 \). Hence we associate ‘User-name’ with the 1st textbox and ‘Password’ with the 2nd.

3.2 The Missing Label Problem

Recall that there may be no labels to associate with a form element. E.g. the two select boxes “1 traveler” and “Economy” in Figure 5. AssociateLabel only considers the text elements present in a web form as the candidate labels for a form element. If all of these candidates get a very low likelihood score then no label is assigned to the form element.

Consequently, elements that have no text elements in the locality are not assigned any labels. We address the missing label problem as follows:

Recall the notion of context of a form element defined in Section 2. We create a Knowledge Base (KB) made up of a set \( L \) of labels and set of pairs \( \{< l_1, \text{ctxt}_{t_1} >, < l_2, \text{ctxt}_{t_2} >, \ldots, < l_0 \text{ ctxt}_{t_0} > \} \). Each \( l_i \in L \) is an externally created label and \( \text{ctxt}_{t_i} \) is a collection of the contexts of similar form elements. We can create such a KB from the labeled data that was used to compute the priors in Section 3.1. Recall also that the context of a form element is a collection of words. Accordingly, we can regard any \( \text{ctxt}_{t_i} \) as a text document class and \( l_i \) as its class label. Now given a form element with a missing label we can classify its context to one of the document classes and assign its label to the form element. We have thus transformed the missing label problem into a document classification problem.

Naive-Bayes classification has been shown to work remarkably well for text document classification [5]. Let \( \text{ctxt}_{t_i} \) denote the \( i \)th context in the KB. Naive-Bayes assumes that given \( l_k \) the words appearing in \( \text{ctxt}_{t_i} \) are independent of each other. Hence \( p(\text{ctxt}_{t_i}|l_k) = \prod_{j=1}^{\text{ctxt}_{t_i}} p(w_t|l_k) \), where \( p(w_t|l_k) \) is the probability of a context word \( w_t \) given label \( l_k \).

Let \( p(l_k|\text{ctxt}_{t_i}) \) denote the probability of associating label \( l_k \) with the form element after observing \( \text{ctxt}_{t_i} \). Note that for labeled data there is no uncertainty about its label i.e. \( p(l_i|\text{ctxt}_{t_i}) = 1 \) for any labeled form element in the KB.

Now \( p(w_t|l_k) \) is an estimate of the number of times word \( w_t \) occurs in the KB for label \( l_k \) divided by the total number of word occurrences in the KB for that label.

\[
p(w_t|l_k) = \frac{\sum_{t=1}^{\text{ctxt}_{t_i}} \text{Count}(w_t, \text{ctxt}_{t_i} p(l_k|\text{ctxt}_{t_i}))}{\sum_{t=1}^{\text{ctxt}_{t_i}} \text{Count}(w_t, \text{ctxt}_{t_i}) p(l_k|\text{ctxt}_{t_i})}
\]

\( \text{Count}(w_t|\text{ctxt}_{t_i}) \) is the number of times word \( w_t \) occurs in \( \text{ctxt}_{t_i} \) and \( V \) is the total number of words in the KB. The prior probabilities of labels \( p(l_k) \) are estimated in the same manner:

\[
p(l_k) = \frac{\sum_{i=1}^{\text{ctxt}_{t_i}} p(l_k|\text{ctxt}_{t_i})}{\text{ctxt}_{t_i}}
\]

For simplicity we have omitted Laplace smoothing in the above equations.

Finally, given \( \text{ctxt}_{u} \) of an unlabeled form element we can determine its label \( l_u \) using:

\[
p(l_k|\text{ctxt}_{u}) = \frac{p(l_k)p(\text{ctxt}_{u}|l_k)}{\sum_{i=1}^{\text{ctxt}_{t_i}} p(l_i)p(\text{ctxt}_{u}|l_i)}
\]

\[
= \frac{p(l_k)\prod_{i=1}^{\text{ctxt}_{t_i}} p(w_t|l_i)}{\sum_{i=1}^{\text{ctxt}_{t_i}} \prod_{i=1}^{\text{ctxt}_{t_i}} p(w_t|l_i)}
\]

and \( l_u = \arg \max_{l_k} p(l_k|\text{ctxt}_{u}) \)

### Making do with Limited Labeled Data

Solving the missing label problem with accuracy needs good quality labeled data in the KB. But creating labeled data is a labor intensive process. It is shown in [7] how to do text classification with little training data. In the terminology of our problem the idea is this: we start off with our KB consisting of the set \( L \) of externally created labels and a small set of labeled form elements along with their contexts. We also have at our disposal a large set \( U \) of unlabeled form elements with their contexts. The goal is to automatically transform \( U \) into a labeled set of form elements.

We bootstrap the transformation process by computing \( p(l_k|\text{ctxt}_{u}) \) for every \( l_k \) in \( L \) and for every \( \text{ctxt}_{u} \) in \( U \) using the probability estimates computed for the labeled data in the KB (equations 12). Note \( \sum_{k=1}^{\text{ctxt}_{t_i}} p(l_k|\text{ctxt}_{u}) = 1 \). What this means is that a “soft” assignment of labels is given to each \( \text{ctxt}_{u} \). The goal now is to finalize a unique label with \( \text{ctxt}_{u} \) and this is accomplished using the EM algorithm as follows: Following the “soft” assignment we merge all the contexts from KB and U. In the E-step we compute label probabilities (equation 12) of each \( \text{ctxt}_{u} \) in U. This feeds into the M-step where re-estimation of the word probabilities and the label probabilities are done using equations (10) and (11). At the end of the iterative process (i.e. when there is no significant improvement in the log-likelihood) a unique label gets assigned to every \( \text{ctxt}_{u} \) in U. We thus obtain a large training set with which we can now assign labels to form elements with missing labels (equation 12, 13).

This idea of coping with limited labeled data also applies to computing priors in AssociateLabel in Section 3.1.

### 4. LABELING TRANSACTION ELEMENTS

Online transactions broadly refer to doing tasks such as shopping, paying bills, making travel plans, etc using the Web. Typically they span several pages and can pose sig-
nificant challenges to blind users, the reason being that amidst a web page’s clutter they will have to: locate different actionable web elements that are relevant for doing the transaction such as ‘sign in’, ‘add to cart’, ‘check out’, etc.; listen to select fragments of the page’s content (e.g. product details); fill out forms, and so on. While extant screen readers provide shortcuts for navigating among links and buttons, doing transactions with non-visual interaction modalities is still a laborious time consuming process for blind users.

The two key problems to be addressed in making transactions accessible are: (i) correctly label all the form elements and (ii) facilitate quick identification of the aforementioned clickable transaction elements. The FMM-based association techniques in Section 3 solve the 1st problem. To address the 2nd problem let’s examine the underlying issues. First, a variety of terms are used to describe similar entities by different content providers. For example, a “search” button is captioned as: “search” in one site, “find” in a 2nd site, “go” in a 3rd site, and so on. Second, some of these clickable transaction elements are images with no alternative text, making them completely inaccessible.

We recently proposed a solution to the 2nd problem in [22]. The idea was to build a KB of transaction element classes such as an Add-To-Cart class, a Checkout class, and so on. Each of these classes is assigned an external class label. A Vector Space Model (VSM) [25] is built for each class consisting of the words appearing in the captions as well as in the contexts of the elements in the class. Context here means the words collected from the caption of the transaction element as well as other content words in its immediate “vicinity”. (Each solid rectangle in Figure 7 is the context of its enclosing element.) Class label for a clickable element is determined by comparing the cosine similarity [10] of its vector with that of every class vector in the KB. The label of the class with the highest similarity becomes the element’s label. Notice that even if the caption of the element is missing (as in Figure 7(d)) the similarity of the surrounding non-caption words in the context can help determine its class label.

But a major drawback of the VSM approach is that it can only accommodate words as features. In contrast FMMs can accommodate different kinds of features. In addition to words the placement of the pricing data in Figure 7 can be added as another feature. This flexibility in incorporating different features can improve the accuracy of labeling.

The simplest FMM formulation for this problem is given by equations (12) and (13) in Section 3.2. \( \text{cntxt}_u \) includes words in the caption and the non-caption words in the vicinity. The formulation can be refined by partitioning the context into a set consisting of two features – caption words and the non-caption words. To accommodate this new feature set we replace \( p(w) \) in equation (12) with

\[
p(w_c) = \pi_{\text{caption}} p(w_c, \text{caption}) + \pi_{\text{context}} p(w_c, \text{noncaption})
\]

\( \pi_{\text{caption}} \) and \( \pi_{\text{context}} \) are mixing proportions for caption and non-caption words and \( \pi_{\text{caption}} + \pi_{\text{context}} = 1 \). All these parameters are estimated using the EM algorithm. To enhance the accuracy even further we can add placements of specific non-caption words such as “price” in Figure 7 as yet another feature to the Add-To-Cart class.

5. EXPERIMENTAL EVALUATION

Effectiveness of Label Association to Form Fields

We evaluated our form labeling approach on two web form datasets: TEL-8 and our own FORM-1. The TEL-8 dataset from the UIUC Web Integration Repository (http://metaquerier.cs.uiuc.edu/repository) contained 447 web forms (5,811 form fields) belonging to ‘Airfare’, ‘Auto’, ‘Books’, ‘Movies’, ‘Rentals’, ‘Hotels’, ‘Jobs’, and ‘Music’ domains. FORM-1 contained 542 web forms (7,800 form fields) automatically collected with Heretrix crawler (http://crawler.archive.org/) from different websites. It is notable, that only 11% of form fields in FORM-1 dataset had explicit field-label associations (showing the current state of affairs in the wild) the rest of the associations we had to create manually. Even worse, up to 1% of form fields in the FORM-1 dataset did not have form labels at all.

To evaluate the accuracy of form label association, we conducted a standard 10-fold cross validation experiment over the combined TEL-8 and FORM-1 datasets. We used 90% of the web forms for training and used the remaining 10% for testing. Each form in the training dataset had every form field explicitly associated with its label. In contrast, there were no explicit field-label associations in the testing dataset. We conducted two separate experiments with and without training the FMM priors. Without the priors, the FMM achieved only 76% accuracy in field-label associations. With the trained priors, the FMM achieved 95% overall accuracy. The average recall (R), precision (P), and F-measure (F)\(^1\) computed over all 10 experiments are summarized in Figure 8. To evaluate the accuracy of label predictor, we removed all textual labels from the testing dataset, and evaluated the ability of label predictor to infer the form field labels by considering only the context of the

\(^{1}\) Precision is the proportion of the retrieved objects that are relevant to the user's information needs to all retrieved objects. Recall is the proportion of relevant objects that were retrieved to all relevant objects. F-Measure is the simple harmonic mean of Recall and Precision.
form. As can be seen in Figure 9, the label predictor achieved 81% accuracy.

To compare FMM approach against the state-of-the-art LABELEX system [20] that does label association in the context of web data integration, we conducted the same experiment using separate domains of the TEL-8 dataset. LABELEXi and FMMi were trained on the entire domain, making them domain independent, while both FMMd and LABELEXd were domain-dependent. Figure 10 demonstrates that FMM outperformed LABELEX in all experiments. As expected, domain-dependent classifiers performed better than the domain-independent ones.

![Comparative Summary](image)

**Comparative Summary.** Our FMM-based algorithm’s 5% error rate can be reduced even further by thresholding the probability scores returned by it. Labels scoring below the threshold are discarded. With a threshold of 0.8, our algorithm’s error rate reduced to 0.05% with 91% recall. Even without thresholds it reduced the number of errors by half compared to LABELEX [20]. This is significant for user experience. Since [20] uses a decision tree, its error rate can’t be reduced using thresholds.

In regards to our previous work labeling transaction objects [22], we point out that it only deals with such objects. The limitation precluding its extension to forms is that its Vector Space Model can only accommodate words as features. So it cannot utilize continuous-valued (CV) features like distance and similarity as is done by our labeling algorithm.

### 6. PRELIMINARY USER STUDY

To validate the results of our approach to form-field labeling with the target population, we used our HearSay non-visual web browser [11]. The browser provides a standard screen-reading interface similar to that of JAWS. For simplicity, we will refer to our basic non-visual browser as the Base System (BS). To verify the effectiveness of our label association method, we extended BS with a module that uses our FMM-based algorithm to create field-label associations. We will refer to the resulting aural browser as the Labeling System (LS).

To evaluate the new approach we recruited two blind subjects, who were expert BS users. The objective of the study was to see how labeling improved user experiences form in form filing.
The experiment was conducted on 6 web forms from expedia.com, kayak.com, jobs.com, and 3 other handcrafted web pages of structure similar to that of the first 3. The tasks included reserving a flight, renting a car and applying for a job. Each participant completed 3 tasks with BS and 3 tasks with LS for a total of 3 x 2 = 6 trials. The 3 tasks performed with a particular version of non-visual browser (either BS or LS) were conducted sequentially. The system order (BS/LS) was counter-balanced using a 4-cell design.

During the study, we measured the time it took the subject to complete each task. The results of the user study showed that the two subjects were able to fill out forms with LS at least twice as fast with BS. They spent on average 185.43 sec. (Std. Dev. 66.97) per task using BS, and 81.85 sec. (Std. Dev. 21.22) per task using LS (Figure 12).

Figure 12: Mean and St. Dev. of task completion time

We also counted the number of form fields per task which the two subjects had difficulty understanding. The forms on average contained 14 fields that had to be filled out. Using BS, both subjects had difficulty with an average of 3.25 form fields per task, while with LS they had trouble with an average of only 0.5 form fields per task. The most frequent cases of mislabeling by our algorithm occurred when candidate labels matched to meaningless form-field names, e.g., “box1”, sometimes causing text similarity dominate other feature values.

At the end of the study, the two subjects were asked Likert scale questions to get the subjective opinion about the BS and LS systems and their form filling experience in general (See Table 1). Both subjects admitted that they often experienced difficulty filling out web forms and strongly agreed that form filling would be easier if all form fields were properly labeled. The two unanimously agreed that LS made form filling easier for them.

<table>
<thead>
<tr>
<th>Likert Questions</th>
<th>U1</th>
<th>U2</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>I often encounter form fields which have not been labeled correctly.</td>
<td>3</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>I often experience difficulty in filling out web forms.</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Filling out labeled form fields is more efficient.</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Labeling System made form filling easier.</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 1: Likert Questions (1=Strong Disagree 5=Strongly Agree)

Our dataset shows that 89% of form fields don’t have associated label tags, requiring blind users to infer them, which can be frustrating. In our experience, user mistakes in label assignment can easily exceed the 5% error rate of our system. Our algorithm could also point out automatically assigned labels (e.g., with earcons or specific words), giving the user an indication that there is a possibility of a mistake. So overall, our approach reduces cognitive load on users and improves inherent serial interface inefficiencies.

7. RELATED WORK

The importance of web accessibility is well established. As far as assistive technologies are concerned, major operating systems (e.g. Windows, Linux) provide a choice of accessibility options such as magnifying glass, text narration, etc. There is also a variety of software tools used for screen-reading and (e.g. JAWS, VoiceOver, Windows-Eyes). Additionally, there are applications specifically targeting web browsing include [8, 9, 10, 11].

Inadequacies in both the quality and quantity of accessibility metadata have led to numerous attempts to automatically improve accessibility of web sites. One popular approach is transcoding – automatic modification of the original content before it reaches end-users. This technique has been extensively explored for web accessibility [13]. Many transcoding approaches are based on a proxy server performing the transcoding. Examples include [14] that automatically adds alternative text to images and [15] that uses an ontology in conjunction with CSS to make pages more accessible. In contrast our approach, which is also concerned with improving web accessibility, does not transcode web pages for consumption by other assistive technologies.

While the above works are broadly related to web accessibility, research specifically targeted towards labeling form and form elements appear in [17, 18, 19, 20]. The exclusive focus of all these works is on Information Integration from web data sources. [16] classifies forms into domain categories such as airlines, jobs, etc. [17] computes the visual distances between different elements in a form and associates the label which is closest to the form element. [18] and [19] cast the problem as a parsing problem using hand-crafted parsing rules. The notable differences between our work and [18,19] are that we use a probabilistic model to capture variability in the features of the form elements and more importantly, we use an automated approach based on machine learning. The LABELEX work in [20] describes experimenting with different classifiers for form element labeling and reports best results using a Naïve Bayes-Decision Tree tandem. Naïve Bayes assumes that all the features are independent which may not be realistic (see our arguments in Section 3.1). Our FMM formulation in Section 3.1 makes no such assumption. Moreover a Decision Tree classifier typically operates with discrete values and hence requires discretizing continuous valued features such as distance and similarity. But more importantly, it’s not clear how to set thresholds for controlling the labeling error rate with decision trees. In contrast, FMM provides a uniform...
framework to explicitly model the problem probabilistically without having to resort to unnatural discretization of continuous values. Being a Bayesian framework the setting of thresholds for controlling errors is naturally doable. [21] uses the heuristic of applying pixel distance and relative position in tandem for form labeling. Other features such as sim and type that improve accuracy are not used. Beyond all these differences none of these works address the issue of missing labels or labeling of transaction elements. For the latter problem a solution based on Vector Space Model appears in our earlier work [22]. But our FMM-based algorithm is more general and has no loss in performance. Another piece of related work concerns the use of context in web accessibility. The notion of context is quite broad with no unique definition. In the context-based summarization works [23], the context of a link is defined as an ad hoc collection of words around it. In our own context-directed browsing work [24], the context of a link is defined to be the content words in the link’s neighborhood related in the sense of “similar topic” to the caption on the link.

8. CONCLUSION
The research on the label association problem presented in this paper opens up interesting application avenues. On the engineering side it is conceivable to augment web content creation tools with the ideas presented here. Such augmented tools can create seamless accessible content even if the authors are unaware of accessibility guidelines. Another application is in information integration from web sources. Our techniques can lead to an integrated system that can supply answers to queries that require automatic filling out of forms. Lastly, adaptation of our techniques to work with any screen reader with minimal screen reader specific specialization is an intriguing and interesting problem.

9. ACKNOWLEDGEMENTS
The research reported here is supported in part by NSF grants IIS-0534419, CNS-0751083 and IIS-0808678.

10. REFERENCES