Feature Representation for Effective Action-Item Detection

Paul N. Bennett
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
pbennett+@cs.cmu.edu

Jaime Carbonell
Language Technologies Institute.
Carnegie Mellon University
Pittsburgh, PA 15213
jgc+@cs.cmu.edu

ABSTRACT

E-mail users face an ever-growing challenge in managing their inboxes due to the growing centrality of email in the workplace for task assignment, action requests, and other roles beyond information dissemination. Whereas Information Retrieval and Machine Learning techniques are gaining initial acceptance in spam filtering and automated folder assignment, this paper reports on a new task: automated *action-item detection*, in order to flag emails that require responses, and to highlight the specific passage(s) indicating the request(s) for action. Unlike standard topic-driven text classification, action-item detection requires inferring the sender's intent, and as such responds less well to pure bag-of-words classification. However, using enriched feature sets, such as n-grams (up to n=4) with chi-squared feature selection, and contextual cues for action-item location improve performance by up to 10% over unigrams, using in both cases state of the art classifiers such as SVMs with automated model selection via embedded cross-validation.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.6 [**Artificial Intelligence**]: Learning; I.5.4 [**Pattern Recognition**]: Applications

General Terms

Experimentation

Keywords

Text classification, speech acts, feature selection, e-mail, n-grams, SVMs

1. INTRODUCTION

E-mail users are facing an increasingly difficult task of managing their inboxes in the face of mounting challenges that result from rising e-mail usage. This includes prioritizing e-mails over a range of sources from business partners to family members, filtering and reducing junk e-mail, and quickly managing requests that demand the receiver's attention or action. Automated *action-item detection* targets the third of these problems by attempting to detect which e-mails *require* an action or response with information, and within those e-mails, attempting to highlight the sentence (or other passage length) that directly indicates the action request.

From: Henry Hutchins <hhutchins@innovative.company.com>
To: Sara Smith; Joe Johnson; William Woolings
Subject: meeting with prospective customers
Sent: Fri 12/10/2005 8:08 AM

Hi All,
I'd like to remind all of you that the group from GRTY will be visiting us next Friday at 4:30 p.m. The current schedule looks like this:
+ 9:30 a.m. Informal Breakfast and Discussion in Cafeteria
+ 10:30 a.m. Company Overview
+ 11:00 a.m. Individual Meetings (Continue Over Lunch)
+ 2:00 p.m. Tour of Facilities
+ 3:00 p.m. Sales Pitch
In order to have this go off smoothly, I would like to practice the presentation well in advance. *As a result, I will need each of your parts by Wednesday.*
Keep up the good work!
–Henry

Figure 1: An E-mail with emphasized Action-Item, an *explicit* request that requires the recipient's attention or action.

Such a detection system can be used as one part of an e-mail agent which would assist a user in processing important e-mails quicker than would have been possible without the agent. We view action-item detection as one necessary component of a successful e-mail agent which would perform spam detection, action-item detection, topic classification and priority ranking, among other functions. The utility of such a detector can manifest as a method of prioritizing e-mails according to task-oriented criteria other than the standard ones of topic and sender or as a means of ensuring that the email user hasn't dropped the proverbial ball by forgetting to address an action request.

Action-item detection differs from standard text classification in two important ways. First, the user is interested both in detecting whether an email contains action items and in locating exactly where these action item requests are contained within the email body. In contrast, standard text categorization merely assigns a topic label to each text, whether that label corresponds to an e-mail folder or a controlled indexing vocabulary [12, 15, 22]. Second, action-item detection attempts to recover the email sender's intent — whether she means to elicit response or action on the part of the receiver; note that for this task, classifiers using only unigrams as

SIGIR'05, August 15–August 19, 2005, Salvador, Brazil.

features do not perform optimally, as evidenced in our results below. Instead we find that we need more information-laden features such as higher-order n-grams. Text categorization by topic, on the other hand, works very well using just individual words as features [2, 9, 13, 17]. In fact, genre-classification, which one would think may require more than a bag-of-words approach, also works quite well using just unigram features [14]. Topic detection and tracking (TDT), also works well with unigram feature sets [1, 20]. We believe that action-item detection is one of the first clear instances of an IR-related task where we must move beyond bag-of-words to achieve high performance, albeit not too far, as bag-of-n-grams seem to suffice.

We first review related work for similar text classification problems such as e-mail priority ranking and speech act identification. Then we more formally define the action-item detection problem, discuss the aspects that distinguish it from more common problems like topic classification, and highlight the challenges in constructing systems that can perform well at the sentence and document level. From there, we move to a discussion of feature representation and selection techniques appropriate for this problem and how standard text classification approaches can be adapted to smoothly move from the sentence-level detection problem to the document-level classification problem. We then conduct an empirical analysis that helps us determine the effectiveness of our feature extraction procedures as well as establish baselines for a number of classification algorithms on this task. Finally, we summarize this paper's contributions and consider interesting directions for future work.

2. RELATED WORK

Several other researchers have considered very similar text classification tasks. Cohen et al. [5] describe an ontology of "speech acts", such as "Propose a Meeting", and attempt to predict when an e-mail contains one of these speech acts. We consider action-items to be an important specific type of speech act that falls within their more general classification. While they provide results for several classification methods, their methods only make use of human judgments at the document-level. In contrast, we consider whether accuracy can be increased by using finer-grained human judgments that mark the specific sentences and phrases of interest.

Corston-Oliver et al. [6] consider detecting items in e-mail to "Put on a To-Do List". This classification task is very similar to ours except they do not consider "simple factual questions" to belong to this category. We include questions, but note that not all questions are action-items — some are rhetorical or simply social convention, "How are you?". From a learning perspective, while they make use of judgments at the sentence-level, they do not explicitly compare what if any benefits finer-grained judgments offer. Additionally, they do not study alternative choices or approaches to the classification task. Instead, they simply apply a standard SVM at the sentence-level and focus primarily on a linguistic analysis of how the sentence can be logically reformulated before adding it to the task list. In this study, we examine several alternative classification methods, compare document-level and sentence-level approaches and analyze the machine learning issues implicit in these problems.

Interest in a variety of learning tasks related to e-mail has been rapidly growing in the recent literature. For example, in a forum dedicated to e-mail learning tasks, Culotta et al. [7] presented methods for learning social networks from e-mail. In this work, we do not focus on peer relationships; however, such methods could complement those here since peer relationships often influence word choice when requesting an action.

3. PROBLEM DEFINITION & APPROACH

In contrast to previous work, we explicitly focus on the benefits that finer-grained, more costly, sentence-level human judgments offer over coarse-grained document-level judgments. Additionally, we consider multiple standard text classification approaches and analyze both the quantitative and qualitative differences that arise from taking a document-level vs. a sentence-level approach to classification. Finally, we focus on the representation necessary to achieve the most competitive performance.

3.1 Problem Definition

In order to provide the most benefit to the user, a system would not only detect the document, but it would also indicate the specific sentences in the e-mail which contain the action-items. Therefore, there are three basic problems:

1. *Document detection*: Classify a document as to whether or not it contains an action-item.
2. *Document ranking*: Rank the documents such that all documents containing action-items occur as high as possible in the ranking.
3. *Sentence detection*: Classify each sentence in a document as to whether or not it is an action-item.

As in most Information Retrieval tasks, the weight the evaluation metric should give to precision and recall depends on the nature of the application. In situations where a user will eventually read all received messages, ranking (*e.g.,* via precision at recall of 1) may be most important since this will help encourage shorter delays in communications between users. In contrast, high-precision detection at low recall will be of increasing importance when the user is under severe time-pressure and therefore will likely not read all mail. This can be the case for crisis managers during disaster management. Finally, sentence detection plays a role in both time-pressure situations and simply to alleviate the user's required time to gist the message.

3.2 Approach

As mentioned above, the labeled data can come in one of two forms: a *document-labeling* provides a yes/no label for each document as to whether it contains an action-item; a *phrase-labeling* provides only a yes label for the specific items of interest. We term the human judgments a *phrase-labeling* since the user's view of the action-item may not correspond with actual sentence boundaries or predicted sentence boundaries. Obviously, it is straightforward to generate a document-labeling consistent with a phrase-labeling by labeling a document "yes" if and only if it contains at least one phrase labeled "yes".

To train classifiers for this task, we can take several viewpoints related to both the basic problems we have enumerated and the form of the labeled data. The *document-level* view treats each e-mail as a learning instance with an associated class-label. Then, the document can be converted to a feature-value vector and learning progresses as usual. Applying a document-level classifier to document detection and ranking is straightforward. In order to apply it to sentence detection, one must make additional steps. For example, if the classifier predicts a document contains an action-item, then areas of the document that contain a high-concentration of words which the model weights heavily in favor of action-items can be indicated. The obvious benefit of the document-level approach is that training set collection costs are lower since the user only has to specify whether or not an e-mail contains an action-item and not the specific sentences.

In the *sentence-level* view, each e-mail is automatically segmented into sentences, and each sentence is treated as a learning instance with an associated class-label. Since the phrase-labeling provided by the user may not coincide with the automatic segmentation, we must determine what label to assign a partially overlapping sentence when converting it to a learning instance. Once trained, applying the resulting classifiers to sentence detection is now straightforward, but in order to apply the classifiers to document detection and document ranking, the individual predictions over each sentence must be aggregated in order to make a document-level prediction. This approach has the potential to benefit from more-specific labels that enable the learner to focus attention on the key sentences instead of having to learn based on data that the majority of the words in the e-mail provide no or little information about class membership.

3.2.1 Features

Consider some of the phrases that might constitute part of an action item: "would like to know", "let me know", "as soon as possible", "have you". Each of these phrases consists of common words that occur in many e-mails. However, when they occur in the same sentence, they are far more indicative of an action-item. Additionally, order can be important: consider "have you" versus "you have". Because of this, we posit that n-grams play a larger role in this problem than is typical of problems like topic classification. Therefore, we consider all n-grams up to size 4.

When using n-grams, if we find an n-gram of size 4 in a segment of text, we can represent the text as just one occurrence of the n-gram or as one occurrence of the n-gram and an occurrence of each smaller n-gram contained by it. We choose the second of these alternatives since this will allow the algorithm itself to smoothly back-off in terms of recall. Methods such as naïve Bayes may be hurt by such a representation because of double-counting.

Since sentence-ending punctuation can provide information, we retain the terminating punctuation token when it is identifiable. Additionally, we add a *beginning-of-sentence* and *end-of-sentence* token in order to capture patterns that are often indicators at the beginning or end of a sentence. Assuming proper punctuation, these extra tokens are unnecessary, but often e-mail lacks proper punctuation. In addition, for the sentence-level classifiers that use n-grams, we additionally code for each sentence a binary encoding of the *position* of the sentence relative to the document. This encoding has eight associated features that represent which octile (the first eighth, second eighth, *etc.*) contains the sentence.

3.2.2 Implementation Details

In order to compare the document-level to the sentence-level approach, we compare predictions at the document-level. We do not address how to use a document-level classifier to make predictions at the sentence-level.

In order to automatically segment the text of the e-mail, we use the RASP statistical parser [4]. Since the automatically segmented sentences may not correspond directly with the phrase-level boundaries, we treat any sentence that contains at least 30% of a marked action-item segment as an action-item. When evaluating sentence-detection for the sentence-level system, we use these class labels as ground truth. Since we are not evaluating multiple segmentation approaches, this does not bias any of the methods. If multiple segmentation systems were under evaluation, one would need to use a metric that matched predicted positive sentences to phrases labeled positive. The metric would need to punish overly long true predictions as well as too short predictions. Our criteria for converting to labeled instances implicitly includes both criteria. Since the segmentation is fixed, an overly long prediction would be predicting "yes" for many "no" instances since presumably the extra length corresponds to additional segmented sentences all of which do not contain 30% of action-item. Likewise, a too short prediction must correspond to a small sentence included in the action-item but not constituting all of the action-item. Therefore, in order to consider the prediction to be too short, there will be an additional preceding/following sentence that is an action-item where we incorrectly predicted "no".

Once a sentence-level classifier has made a prediction for each sentence, we must combine these predictions to make both a document-level prediction and a document-level score. We use the simple policy of predicting positive when any of the sentences is predicted positive. In order to produce a document score for ranking, the confidence that the document contains an action-item is:

$$\psi(d) = \begin{cases} \frac{1}{n(d)} \sum_{s \in d | \pi(s)=1} \psi(s) & \text{if for any } s \in d, \pi(s) = 1 \\ \frac{1}{n(d)} \max_{s \in d} \psi(s) & o.w. \end{cases}$$

where s is a sentence in document d, π is the classifier's 1/0 prediction, ψ is the score the classifier assigns as its confidence that $\pi(s) = 1$, and $n(d)$ is the greater of 1 and the number of (unigram) tokens in the document. In other words, when any sentence is predicted positive, the document score is the length normalized sum of the sentence scores above threshold. When no sentence is predicted positive, the document score is the maximum sentence score normalized by length. As in other text problems, we are more likely to emit false positives for documents with more words or sentences. Thus we include a length normalization factor.

4. EXPERIMENTAL ANALYSIS

4.1 The Data

Our corpus consists of e-mails obtained from volunteers at an educational institution and cover subjects such as: organizing a research workshop, arranging for job-candidate interviews, publishing proceedings, and talk announcements. The messages were anonymized by replacing the names of each individual and institution with a pseudonym.[1] After attempting to identify and eliminate duplicate e-mails, the corpus contains 744 e-mail messages.

After identity anonymization, the corpora has three basic versions. *Quoted material* refers to the text of a previous e-mail that an author often leaves in an e-mail message when responding to the e-mail. Quoted material can act as noise when learning since it may include action-items from previous messages that are no longer relevant. To isolate the effects of quoted material, we have three versions of the corpora. The *raw* form contains the basic messages. The *auto-stripped* version contains the messages after quoted material has been automatically removed. The *hand-stripped* version contains the messages after quoted material has been removed by a human. Additionally, the hand-stripped version has had any xml content and e-mail signatures removed — leaving only the essential content of the message. The studies reported here are performed with the hand-stripped version. This allows us to balance the cognitive load in terms of number of tokens that must be read in the user-studies we report — including quoted material would complicate the user studies since some users might skip the material while others read it. Additionally, ensuring all quoted material is removed

[1]We have an even more highly anonymized version of the corpus that can be made available for some outside experimentation. Please contact the authors for more information on obtaining this data.

prevents tainting the cross-validation since otherwise a test item could occur as quoted material in a training document.

4.1.1 Data Labeling

Two human annotators labeled each message as to whether or not it contained an action-item. In addition, they identified each segment of the e-mail which contained an action-item. A segment is a contiguous section of text selected by the human annotators and may span several sentences or a complete phrase contained in a sentence. They were instructed that an action item is "an explicit request for information that requires the recipient's attention or a required action" and told to "highlight the phrases or sentences that make up the request".

		Annotator 1	
		No	Yes
Annotator 2	No	391	26
	Yes	29	298

Table 1: Agreement of Human Annotators at Document Level

Annotator One labeled 324 messages as containing action items. Annotator Two labeled 327 messages as containing action items. The agreement of the human annotators is shown in Tables 1 and 2. The annotators are said to *agree at the document-level* when both marked the same document as containing no action-items or both marked at least one action-item regardless of whether the text segments were the same. At the document-level, the annotators agreed 93% of the time. The kappa statistic [3, 5] is often used to evaluate inter-annotator agreement:

$$\kappa = \frac{A - R}{1 - R}$$

A is the empirical estimate of the probability of **a**greement. R is the empirical estimate of the probability of **r**andom agreement given the empirical class priors. A value close to -1 implies the annotators agree far less often than would be expected randomly, while a value close to 1 means they agree more often than randomly expected.

At the document-level, the kappa statistic for inter-annotator agreement is 0.85. This value is both strong enough to expect the problem to be learnable and is comparable with results for similar tasks [5, 6].

In order to determine the sentence-level agreement, we use each judgment to create a sentence-corpus with labels as described in Section 3.2.2, then consider the agreement over these sentences. This allows us to compare agreement over "no judgments". We perform this comparison over the hand-stripped corpus since that eliminates spurious "no" judgments that would come from including quoted material, etc. Both annotators were free to label the subject as an action-item, but since neither did, we omit the subject line of the message as well. This only reduces the number of "no" agreements. This leaves 6301 automatically segmented sentences. At the sentence-level, the annotators agreed 98% of the time, and the kappa statistic for inter-annotator agreement is 0.82.

In order to produce one single set of judgments, the human annotators went through each annotation where there was disagreement and came to a consensus opinion. The annotators did not collect statistics during this process but anecdotally reported that the majority of disagreements were either cases of clear annotator oversight or different interpretations of conditional statements. For example, *"If you would like to keep your job, come to tomorrow's meeting"* implies a required action where *"If you would like to join the football betting pool, come to tomorrow's meeting"* does not. The first would be an action-item in most contexts while the second would not. Of course, many conditional statements are not so clearly interpretable. After reconciling the judgments there are 416 e-mails with no action-items and 328 e-mails containing action-items. Of the 328 e-mails containing action-items, 259 messages have one action-item segment; 55 messages have two action-item segments; 11 messages have three action-item segments. Two messages have four action-item segments, and one message has six action-item segments. Computing the sentence-level agreement using the reconciled "gold standard" judgments with each of the annotators' individual judgments gives a kappa of 0.89 for Annotator One and a kappa of 0.92 for Annotator Two.

		Annotator 1	
		No	Yes
Annotator 2	No	5810	65
	Yes	74	352

Table 2: Agreement of Human Annotators at Sentence Level

In terms of message characteristics, there were on average 132 content tokens in the body after stripping. For action-item messages, there were 115. However, by examining Figure 2 we see the length distributions are nearly identical. As would be expected for e-mail, it is a long-tailed distribution with about half the messages having more than 60 tokens in the body (this paragraph has 65 tokens).

4.2 Classifiers

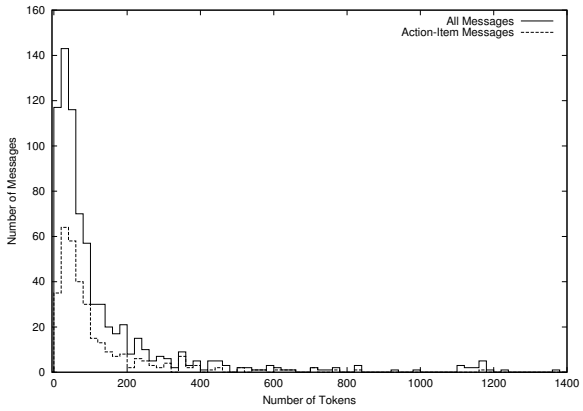
For this experiment, we have selected a variety of standard text classification algorithms. In selecting algorithms, we have chosen algorithms that are not only known to work well but which differ along such lines as discriminative vs. generative and lazy vs. eager. We have done this in order to provide both a competitive and thorough sampling of learning methods for the task at hand. This is important since it is easy to improve a strawman classifier by introducing a new representation. By thoroughly sampling alternative classifier choices we demonstrate that representation improvements over bag-of-words are not due to using the information in the bag-of-words poorly.

4.2.1 kNN

We employ a standard variant of the k-nearest neighbor algorithm used in text classification, kNN with s-cut score thresholding [19]. We use a tfidf-weighting of the terms with a distance-weighted vote of the neighbors to compute the score before thresholding it. In order to choose the value of s for thresholding, we perform leave-one-out cross-validation over the training set. The value of k is set to be $2(\lceil \log_2 N \rceil + 1)$ where N is the number of training points. This rule for choosing k is theoretically motivated by results which show such a rule converges to the optimal classifier as the number of training points increases [8]. In practice, we have also found it to be a computational convenience that frequently leads to comparable results with numerically optimizing k via a cross-validation procedure.

4.2.2 Naïve Bayes

We use a standard multinomial naïve Bayes classifier [16]. In using this classifier, we smoothed word and class probabilities using a Bayesian estimate (with the word prior) and a Laplace m-estimate, respectively.



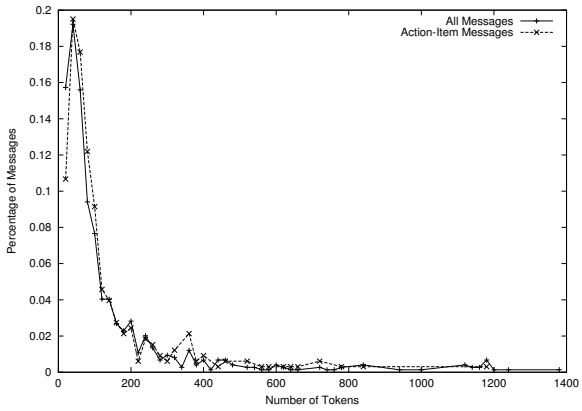


Figure 2: The Histogram (left) and Distribution (right) of Message Length. A bin size of 20 words was used. Only tokens in the body after hand-stripping were counted. After stripping, the majority of words left are usually actual message content.

	Classifiers	Document Unigram	Document Ngram	Sentence Unigram	Sentence Ngram
F1	kNN	0.6670 ± *0.0288*	0.7108 ± *0.0699*	0.7615 ± *0.0504*	**0.7790** ± *0.0460*
	naïve Bayes	0.6572 ± *0.0749*	0.6484 ± *0.0513*	0.7715 ± *0.0597*	**0.7777** ± *0.0426*
	SVM	0.6904 ± *0.0347*	0.7428 ± *0.0422*	0.7282 ± *0.0698*	**0.7682** ± *0.0451*
	Voted Perceptron	0.6288 ± *0.0395*	0.6774 ± *0.0422*	0.6511 ± *0.0506*	**0.6798** ± *0.0913*
Accuracy	kNN	0.7029 ± *0.0659*	0.7486 ± *0.0505*	0.7972 ± *0.0435*	**0.8092** ± *0.0352*
	naïve Bayes	0.6074 ± *0.0651*	0.5816 ± *0.1075*	0.7863 ± *0.0553*	**0.8145** ± *0.0268*
	SVM	0.7595 ± *0.0309*	0.7904 ± *0.0349*	0.7958 ± *0.0551*	**0.8173** ± *0.0258*
	Voted Perceptron	0.6531 ± *0.0390*	**0.7164** ± *0.0376*	0.6413 ± *0.0833*	0.7082 ± *0.1032*

Table 3: Average Document-Detection Performance during Cross-Validation for Each Method and the Sample Standard Deviation (S_{n-1}) in italics. The best performance for each classifier is shown in bold.

4.2.3 SVM

We have used a linear SVM with a tfidf feature representation and L2-norm as implemented in the SVM^{light} package v6.01 [11]. All default settings were used.

4.2.4 Voted Perceptron

Like the SVM, the Voted Perceptron is a kernel-based learning method. We use the same feature representation and kernel as we have for the SVM, a linear kernel with tfidf-weighting and an L2-norm. The voted perceptron is an online-learning method that keeps a history of past perceptrons used, as well as a weight signifying how often that perceptron was correct. With each new training example, a correct classification increases the weight on the current perceptron and an incorrect classification updates the perceptron. The output of the classifier uses the weights on the perceptra to make a final "voted" classification. When used in an offline-manner, multiple passes can be made through the training data. Both the voted perceptron and the SVM give a solution from the same hypothesis space — in this case, a linear classifier. Furthermore, it is well-known that the Voted Perceptron increases the margin of the solution after each pass through the training data [10]. Since Cohen et al. [5] obtain worse results using an SVM than a Voted Perceptron with one training iteration, they conclude that the best solution for detecting speech acts may not lie in an area with a large margin. Because their tasks are highly similar to ours, we employ both classifiers to ensure we are not overlooking a competitive alternative classifier to the SVM for the basic bag-of-words representation.

4.3 Performance Measures

To compare the performance of the classification methods, we look at two standard performance measures, F1 and accuracy. The F1 measure [18, 21] is the harmonic mean of precision and recall where $Precision = \frac{Correct\ Positives}{Predicted\ Positives}$ and $Recall = \frac{Correct\ Positives}{Actual\ Positives}$.

4.4 Experimental Methodology

We perform standard 10-fold cross-validation on the set of documents. For the sentence-level approach, all sentences in a document are either entirely in the training set or entirely in the test set for each fold. For significance tests, we use a two-tailed t-test [21] to compare the values obtained during each cross-validation fold with a p-value of 0.05.

Feature selection was performed using the chi-squared statistic. Different levels of feature selection were considered for each classifier. Each of the following number of features was tried: $10, 25, 50, 100, 250, 750, 1000, 2000, 4000$. There are approximately 4700 unigram tokens without feature selection. In order to choose the number of features to use for each classifier, we perform nested cross-validation and choose the settings that yield the optimal document-level F1 for that classifier. For this study, only the body of each e-mail message was used. Feature selection is always applied to all candidate features. That is, for the n-gram representation, the n-grams and position features are also subject to removal by the feature selection method.

4.5 Results

The results for document-level classification are given in Table 3. The primary hypothesis we are concerned with is that n-grams are critical for this task; if this is true, we expect to see a significant gap in performance between the document-level classifiers that use n-grams (denoted *Document Ngram*) and those using only unigram features (denoted *Document Unigram*). Examining Table 3, we observe that this is indeed the case for every classifier except naïve Bayes. This difference in performance produced by the n-gram representation is statistically significant for each classifier except for naïve Bayes and the accuracy metric for kNN (see Table 4). Naïve Bayes poor performance with the n-gram representation is not surprising since the bag-of-n-grams causes excessive double-counting as mentioned in Section 3.2.1; however, naïve bayes is not hurt at the sentence-level because the sparse examples provide few chances for agglomerative effects of double counting. In either case, when a language-modeling approach is desired, modeling the n-grams directly would be preferable to naïve Bayes. More importantly for the n-gram hypothesis, the n-grams lead to the best document-level classifier performance as well.

As would be expected, the difference between the *sentence-level* n-gram representation and unigram representation is small. This is because the window of text is so small that the unigram representation, when done at the sentence-level, implicitly picks up on the power of the n-grams. Further improvement would signify that the order of the words matter even when only considering a small sentence-size window. Therefore, the finer-grained sentence-level judgments allows a unigram representation to succeed but only when performed in a small window — behaving as an n-gram representation for all practical purposes.

	Document Winner	Sentence Winner
kNN	**Ngram**	Ngram
naïve Bayes	Unigram	Ngram
SVM	**Ngram**†	Ngram
Voted Perceptron	**Ngram**†	Ngram

Table 4: Significance results for n-grams versus unigrams for document detection using document-level and sentence-level classifiers. When the F1 result is statistically significant, it is shown in bold. When the accuracy result is significant, it is shown with a †.

	F1 Winner	Accuracy Winner
kNN	**Sentence**	**Sentence**
naïve Bayes	**Sentence**	**Sentence**
SVM	Sentence	**Sentence**
Voted Perceptron	Sentence	Document

Table 5: Significance results for sentence-level classifiers vs. document-level classifiers for the document detection problem. When the result is statistically significant, it is shown in bold.

Further highlighting the improvement from finer-grained judgments and n-grams, Figure 3 graphically depicts the edge the SVM sentence-level classifier has over the standard bag-of-words approach with a precision-recall curve. In the high precision area of the graph, the consistent edge of the sentence-level classifier is rather impressive — continuing at precision 1 out to 0.1 recall. This would mean that a tenth of the user's action-items would be placed at the top of their action-item sorted inbox. Additionally, the large separation at the top right of the curves corresponds to the area where the optimal F1 occurs for each classifier, agreeing with the large improvement from 0.6904 to 0.7682 in F1 score. Considering the relative unexplored nature of classification at the sentence-level, this gives great hope for further increases in performance.

	Accuracy		F1	
	Unigram	Ngram	Unigram	Ngram
kNN	0.9519	0.9536	0.6540	0.6686
naïve Bayes	0.9419	0.9550	0.6176	0.6676
SVM	0.9559	0.9579	0.6271	0.6672
Voted Perceptron	0.8895	0.9247	0.3744	0.5164

Table 6: Performance of the Sentence-Level Classifiers at Sentence Detection

Although Cohen et al. [5] observed that the Voted Perceptron with a single training iteration outperformed SVM in a set of similar tasks, we see no such behavior here. This further strengthens the evidence that an alternate classifier with the bag-of-words representation could not reach the same level of performance. The Voted Perceptron classifier does improve when the number of training iterations are increased, but it is still lower than the SVM classifier.

Sentence detection results are presented in Table 6. With regard to the sentence detection problem, we note that the $F1$ measure gives a better feel for the remaining room for improvement in this difficult problem. That is, unlike document detection where action-item documents are fairly common, action-item sentences are very rare. Thus, as in other text problems, the accuracy numbers are deceptively high sheerly because of the default accuracy attainable by always predicting "no". Although, the results here are significantly above-random, it is unclear what level of performance is necessary for sentence detection to be useful in and of itself and not simply as a means to document ranking and classification.

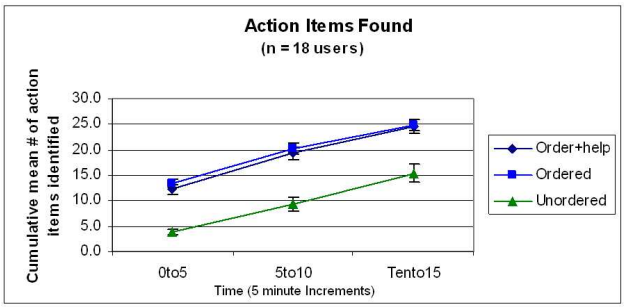


Figure 4: Users find action-items quicker when assisted by a classification system.

Finally, when considering a new type of classification task, one of the most basic questions is whether an accurate classifier built for the task can have an impact on the end-user. In order to demonstrate the impact this task can have on e-mail users, we conducted a user study using an earlier less-accurate version of the sentence classifier — where instead of using just a single sentence, a three-sentence *windowed-approach* was used. There were three distinct sets of e-mail in which users had to find action-items. These sets were either presented in a random order (*Unordered*), ordered by the classifier (*Ordered*), or ordered by the classifier and with the

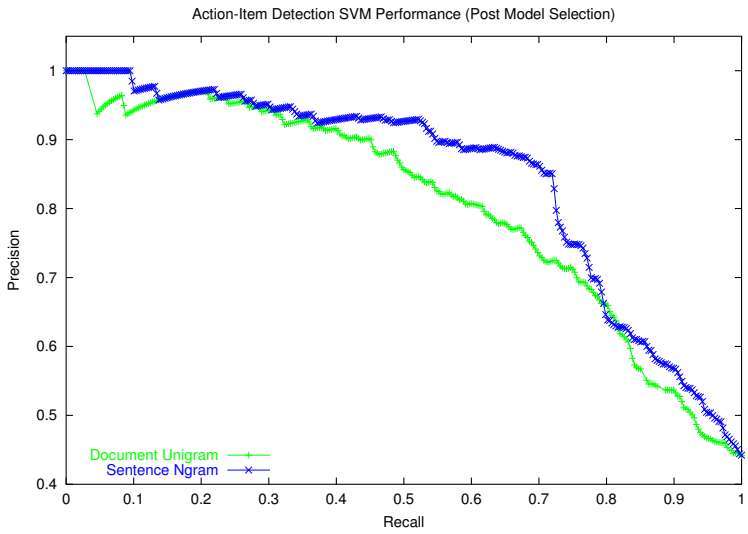


Figure 3: Both n-grams and a small prediction window lead to consistent improvements over the standard approach.

center sentence in the highest confidence window highlighted (*Order+help*). In order to perform fair comparisons between conditions, the overall number of tokens in each message set should be approximately equal; that is, the cognitive reading load should be approximately the same before the classifier's reordering. Additionally, users typically show "practice effects" by improving at the overall task and thus performing better at later message sets. This is typically handled by varying the ordering of the sets across users so that the means are comparable. While omitting further detail, we note the sets were balanced for the total number of tokens and a latin square design was used to balance practice effects.

Figure 4 shows that at intervals of 5, 10, and 15 minutes, users consistently found significantly more action-items when assisted by the classifier, but were most critically aided in the first five minutes. Although, the classifier consistently aids the users, we did not gain an additional end-user impact by highlighting. As mentioned above, this might be a result of the large room for improvement that still exists for sentence detection, but anecdotal evidence suggests this might also be a result of how the information is presented to the user rather than the accuracy of sentence detection. For example, highlighting the wrong sentence near an actual action-item hurts the user's trust, but if a vague indicator (*e.g.,* an arrow) points to the approximate area the user is not aware of the near-miss. Since the user studies used a three sentence window, we believe this played a role as well as sentence detection accuracy.

4.6 Discussion

In contrast to problems where n-grams have yielded little difference, we believe their power here stems from the fact that many of the meaningful n-grams for action-items consist of common words, *e.g.*, "let me know". Therefore, the document-level unigram approach cannot gain much leverage, even when modeling their joint probability correctly, since these words will often co-occur in the document but not necessarily in a phrase. Additionally, action-item detection is distinct from many text classification tasks in that a single sentence can change the class label of the document. As a result, good classifiers cannot rely on aggregating evidence from a large number of weak indicators across the entire document.

Even though we discarded the header information, examining the top-ranked features at the document-level reveals that many of the features are names or parts of e-mail addresses that occurred in the body and are highly associated with e-mails that tend to contain many or no action-items. A few examples are terms such as "org", "bob", and "gov". We note that these features will be sensitive to the particular distribution (senders/receivers) and thus the document-level approach may produce classifiers that transfer less readily to alternate contexts and users at different institutions. This points out that part of the problem of going beyond bag-of-words may be the methodology, and investigating such properties as learning curves and how well a model transfers may highlight differences in models which appear to have similar performance when tested on the distributions they were trained on. We are currently investigating whether the sentence-level classifiers do perform better over different test corpora without retraining.

5. FUTURE WORK

While applying text classifiers at the document-level is fairly well-understood, there exists the potential for significantly increasing the performance of the sentence-level classifiers. Such methods include alternate ways of combining the predictions over each sentence, weightings other than tfidf, which may not be appropriate since sentences are small, better sentence segmentation, and other types of phrasal analysis. Additionally, named entity tagging, time expressions, *etc.*, seem likely candidates for features that can further improve this task. We are currently pursuing some of these avenues to see what additional gains these offer.

Finally, it would be interesting to investigate the best methods for combining the document-level and sentence-level classifiers. Since the simple bag-of-words representation at the document-level leads to a learned model that behaves somewhat like a context-specific prior dependent on the sender/receiver and general topic, a first choice would be to treat it as such when combining probability estimates with the sentence-level classifier. Such a model might serve as a general example for other problems where bag-of-words can establish a baseline model but richer approaches are needed to achieve performance beyond that baseline.

6. SUMMARY AND CONCLUSIONS

The effectiveness of sentence-level detection argues that labeling at the sentence-level provides significant value. Further experiments are needed to see how this interacts with the amount of training data available. Sentence detection that is then agglomerated to document-level detection works surprisingly better given low recall than would be expected with sentence-level items. This, in turn, indicates that improved sentence segmentation methods could yield further improvements in classification.

In this work, we examined how action-items can be effectively detected in e-mails. Our empirical analysis has demonstrated that n-grams are of key importance to making the most of document-level judgments. When finer-grained judgments are available, then a standard bag-of-words approach using a small (sentence) window size and automatic segmentation techniques can produce results almost as good as the n-gram based approaches.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA), or the Department of Interior-National Business Center (DOI-NBC).

We would like to extend our sincerest thanks to Jill Lehman whose efforts in data collection were essential in constructing the corpus, and both Jill and Aaron Steinfeld for their direction of the HCI experiments. We would also like to thank Django Wexler for constructing and supporting the corpus labeling tools and Curtis Huttenhower's support of the text preprocessing package. Finally, we gratefully acknowledge Scott Fahlman for his encouragement and useful discussions on this topic.

7. REFERENCES

[1] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, Washington, D.C., 1998.

[2] C. Apte, F. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, July 1994.

[3] J. Carletta. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

[4] J. Carroll. High precision extraction of grammatical relations. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 134–140, 2002.

[5] W. W. Cohen, V. R. Carvalho, and T. M. Mitchell. Learning to classify email into "speech acts". In *EMNLP-2004 (Conference on Empirical Methods in Natural Language Processing)*, pages 309–316, 2004.

[6] S. Corston-Oliver, E. Ringger, M. Gamon, and R. Campbell. Task-focused summarization of email. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 43–50, 2004.

[7] A. Culotta, R. Bekkerman, and A. McCallum. Extracting social networks and contact information from email and the web. In *CEAS-2004 (Conference on Email and Anti-Spam)*, Mountain View, CA, July 2004.

[8] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, NY, 1996.

[9] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98, Proceedings of the 7th ACM Conference on Information and Knowledge Management*, pages 148–155, 1998.

[10] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[11] T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. J. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 41–56. MIT Press, 1999.

[12] L. S. Larkey. A patent search and classification system. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, pages 179 – 187, 1999.

[13] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *SIGIR '92, Proceedings of the 15th Annual International ACM Conference on Research and Development in Information Retrieval*, pages 37–50, 1992.

[14] Y. Liu, J. Carbonell, and R. Jin. A pairwise ensemble approach for accurate genre classification. In *Proceedings of the European Conference on Machine Learning (ECML)*, 2003.

[15] Y. Liu, R. Yan, R. Jin, and J. Carbonell. A comparison study of kernels for multi-label text classification using category association. In *The Twenty-first International Conference on Machine Learning (ICML)*, 2004.

[16] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *Working Notes of AAAI '98 (The 15th National Conference on Artificial Intelligence), Workshop on Learning for Text Categorization*, pages 41–48, 1998. TR WS-98-05.

[17] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March 2002.

[18] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[19] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):67–88, 1999.

[20] Y. Yang, J. Carbonell, R. Brown, T. Pierce, B. T. Archibald, and X. Liu. Learning approaches to topic detection and tracking. *IEEE EXPERT, Special Issue on Applications of Intelligent Information Retrieval*, 1999.

[21] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR '99, Proceedings of the 22nd Annual International ACM Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.

[22] Y. Yang, J. Zhang, J. Carbonell, and C. Jin. Topic-conditioned novelty detection. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2002.