Modeling Annotators: A Generative Approach to Learning from Annotator Rationales*

Omar F. Zaidan and **Jason Eisner**
Dept. of Computer Science, Johns Hopkins University
Baltimore, MD 21218, USA
{ozaidan,jason}@cs.jhu.edu

Abstract

A human annotator can provide hints to a machine learner by highlighting contextual "rationales" for each of his or her annotations (Zaidan et al., 2007). How can one exploit this side information to better learn the desired parameters θ? We present a generative model of how a given annotator, knowing the true θ, stochastically chooses rationales. Thus, observing the rationales helps us infer the true θ. We collect substring rationales for a sentiment classification task (Pang and Lee, 2004) and use them to obtain significant accuracy improvements for each annotator. Our new generative approach exploits the rationales more effectively than our previous "masking SVM" approach. It is also more principled, and could be adapted to help learn other kinds of probabilistic classifiers for quite different tasks.

1 Background

Many recent papers aim to reduce the amount of annotated data needed to train the parameters of a statistical model. Well-known paradigms include active learning, semi-supervised learning, and either domain adaptation or cross-lingual transfer from existing annotated data.

A rather different paradigm is to change the actual *task* that is given to annotators, giving them a greater hand in shaping the learned classifier. After all, human annotators themselves are more than just black-box classifiers to be run on training data. They possess some introspective knowledge about their own classification procedure. The hope is to mine this knowledge rapidly via appropriate questions and use it to help train a machine classifier. *How* to do this, however, is still being explored.

1.1 Hand-crafted rules

An obvious option is to have the annotators directly express their knowledge by hand-crafting rules. This approach remains "data-driven" if the annotators repeatedly refine their system against a corpus of labeled or unlabeled examples. This achieves high performance in some domains, such as NP chunking (Brill and Ngai, 1999), but requires more analytical skill from the annotators. One empirical study (Ngai and Yarowsky, 2000) found that it also required more annotation time than active learning.

1.2 Feature selection by humans

More recent work has focused on statistical classifiers. Training such classifiers faces the "credit assignment problem." Given a training example x with many features, which features are responsible for its annotated class y? It may take many training examples to distinguish useful vs. irrelevant features.[1]

To reduce the number of training examples needed, one can ask annotators to examine or propose some candidate features. This is possible even for the very large feature sets that are typically used in NLP. In document classification, Raghavan et al. (2006) show that feature selection by an oracle could be helpful, and that humans are both rapid and reasonably good at distinguishing highly useful n-gram features from randomly chosen ones, even when viewing these n-grams out of context.

Druck et al. (2008) show annotators some features f from a fixed feature set, and ask them to choose a class label y such that $p(y \mid f)$ is as high as possible. Haghighi and Klein (2006) do the reverse: for each class label y, they ask the annotators to propose a few "prototypical" features f such that $p(y \mid f)$ is as high as possible.

1.3 Feature selection in context

The above methods consider features out of context. An annotator might have an easier time examining

*This work was supported by National Science Foundation grant No. 0347822 and the JHU WSE/APL Partnership Fund. Special thanks to Christine Piatko for many useful discussions.

[1] Most NLP systems use thousands or millions of features, because it is helpful to include lexical features over a large vocabulary, often conjoined with lexical or non-lexical context.

Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 31–40,
Honolulu, October 2008.

features *in context* to recognize whether they appear relevant. This is particularly true for features that are only modestly or only sometimes helpful, which may be abundant in NLP tasks.

Thus, Raghavan et al. (2006) propose an active learning method in which, while classifying a training document, the annotator also identifies some features of *that* document as particularly relevant. E.g., the annotator might highlight particular unigrams as he or she reads the document. In their proposal, a feature that is highlighted in any document is assumed to be globally more relevant. Its dimension in feature space is scaled by a factor of 10 so that this feature has more influence on distances or inner products, and hence on the learned classifier.

1.4 Concerns about marking features

Despite the success of the above work, we have several concerns about asking annotators to identify globally relevant features.

First, a feature in isolation really does not have a well-defined worth. A feature may be useful only in conjunction with other features,[2] or be useful only to the extent that other correlated features are *not* selected to do the same work.

Second, it is not clear how an annotator would easily view and highlight features in context, except for the simplest feature sets. In the phrase Apple shares up 3%, there may be several features that fire on the substring Apple—responding to the string Apple, its case-invariant form apple, its lemma apple- (which would also respond to apples), its context-dependent sense Apple_2, its part of speech noun, etc. How does the annotator indicate which of these features are relevant?

Third, annotating features is only appropriate when the feature set can be easily understood by a human. This is not always the case. It would be hard for annotators to read, write, or evaluate a description of a complex syntactic configuration in NLP or a convolution filter in machine vision.

Fourth, traditional annotation efforts usually try to remain agnostic about the machine learning methods and features to be used. The project's cost is justified by saying that the annotations will be reused by many researchers (perhaps in a "shared task"), who are free to compete on how they tackle the learning problem. Unfortunately, feature annotation commits to a particular feature set at annotation time. Subsequent research cannot easily adjust the definition of the features, or obtain annotation of new features.

[2]For example, a linear classifier can learn that most training examples satisfy $A \rightarrow B$ by setting $\theta_A = -5$ and $\theta_{A \wedge B} = +5$, but this solution requires selecting both A and $A \wedge B$ as features. More simply, a polynomial kernel can consider the conjunction $A \wedge B$ only if both A and B are selected as features.

2 Annotating Rationales

To solve these problems, we propose that annotators should not *select features* but rather *mark relevant portions of the example*. In earlier work (Zaidan et al., 2007), we called these markings "rationales."

For example, when classifying a movie review as positive or negative, the annotator would also highlight phrases that supported that judgment. Figure 1 shows two such rationales.

A multi-annotator timing study (Zaidan et al., 2007) found that highlighting rationale phrases while reading movie reviews only doubled annotation time, although annotators marked 5–11 rationale substrings in addition to the simple binary class. The benefit justified the extra time. Furthermore, much of the benefit could have been obtained by giving rationales for only a fraction of the reviews.

In the visual domain, when classifying an image as containing a zoo, the annotator might circle some animals or cages and the sign reading "Zoo." The Peekaboom *game* (von Ahn et al., 2006) was in fact built to elicit such approximate yet relevant regions of images. Further scenarios were discussed in (Zaidan et al., 2007): rationale annotation for named entities, linguistic relations, or handwritten digits.

Annotating rationales does not require the annotator to think about the feature space, nor even to know anything about it. Arguably this makes annotation easier and more flexible. It also preserves the reusability of the annotated data. Anyone is free to reuse our collected rationales (section 4) to aid in learning a classifier with richer features, or a different kind of classifier altogether, using either our procedures or novel procedures.

3 Modeling Rationale Annotations

As rationales are more indirect than explicit features, they present a trickier machine learning problem.

We wish to learn the parameters θ of some classifier. How can the annotator's rationales help us to do this without many training examples? We will have to exploit a presumed relationship between the rationales and the optimal value of θ (i.e., the value that we would learn on an infinite training set).

This paper exploits an explicit, parametric model of that relationship. The model's parameters ϕ are intended to capture what that annotator is doing when he or she marks rationales. Most importantly, they capture how he or she is influenced by the true θ. Given this, our learning method will prefer values of θ that would adequately explain the rationales (as well as the training classifications).

3.1 A generative approach

For concreteness, we will assume that the task is document classification. Our training data consists of n triples $\{(x_1, y_1, r_1), ..., (x_n, y_n, r_n)\}$), where x_i is a document, y_i is its annotated class, and r_i is its rationale markup. At test time we will have to predict y_{n+1} from x_{n+1}, without any r_{n+1}.

We propose to jointly choose parameter vectors θ and ϕ to maximize the following regularized conditional likelihood:[3]

$$\prod_{i=1}^{n} p(y_i, r_i \mid x_i, \theta, \phi) \cdot p_{\text{prior}}(\theta, \phi) \quad (1)$$

$$\stackrel{\text{def}}{=} \prod_{i=1}^{n} p_\theta(y_i \mid x_i) \cdot p_\phi(r_i \mid x_i, y_i, \theta) \cdot p_{\text{prior}}(\theta, \phi)$$

Here we are trying to model all the annotations, both y_i and r_i. The first factor predicts y_i using an ordinary probabilistic classifier p_θ, while the novel second factor predicts r_i using a model p_ϕ of how annotators generate the rationale annotations.

The crucial point is that the second factor depends on θ (since r_i is supposed to reflect the relation between x_i and y_i that is modeled by θ). As a result, the learner has an incentive to modify θ in a way that increases the second factor, even if this somewhat decreases the first factor on training data.[4]

After training, one should simply use the first factor $p_\theta(y \mid x)$ to classify test documents x. The second factor is irrelevant for test documents, since they have not been annotated with rationales r.

The second factor may likewise be omitted for any training documents i that have not been annotated with rationales, as there is no r_i to predict in those cases. In the extreme case where no documents are annotated with rationales, equation (1) reduces to the standard training procedure.

3.2 Noisy channel design of rationale models

Like ordinary class annotations, rationale annotations present us with a "credit assignment problem," albeit a smaller one that is limited to features that fire "in the vicinity" of the rationale r. Some of these θ-features were likely responsible for the classification y and hence triggered the rationale. Other such θ-features were just innocent bystanders.

Thus, the interesting part of our model is $p_\phi(r \mid x, y, \theta)$, which models the rationale annotation process. *The rationales r reflect θ, but in noisy ways.*

Taking this noisy channel idea seriously, $p_\phi(r \mid x, y, \theta)$ should consider two questions when assessing whether r is a plausible set of rationales given θ. First, it needs a "language model" of rationales: does r consist of rationales that are well-formed *a priori*, i.e., before θ is considered? Second, it needs a "channel model": does r faithfully signal the features of θ that strongly support classifying x as y?

If a feature contributes heavily to the classification of document x as class y, then the **channel model** should tell us which *parts* of document x tend to be highlighted as a result.

The channel model must know about the particular kinds of features that are extracted by f and scored by θ. Suppose the feature not ... gripping,[5] with weight θ_h, is predictive of the annotated class y. This raises the probabilities of the annotator's highlighting each of various words, or combinations of words, in a phrase like not the most gripping banquet on film. The channel model parameters in ϕ

[3] It would be preferable to integrate out ϕ (and even θ), but more difficult.

[4] Interestingly, even examples where the annotation y_i is wrong or unhelpful can provide useful information about θ via the pair (y_i, r_i). Two annotators marking the same movie review might disagree on whether it is overall a positive or negative review—but the second factor still allows learning positive features from the first annotator's positive rationales, and negative features from the second annotator's negative rationales.

[5] Our current experiments use only unigram features, to match past work, but we use this example to outline how our approach generalizes to complex linguistic (or visual) features.

should specify how *much* each of these probabilities is raised, based on the magnitude of $\theta_h \in \mathbb{R}$, the class y, and the fact that the feature is an instance of the template <Neg> ... <Adjective>. (Thus, ϕ has no parameters specific to the word gripping; it is a low-dimensional vector that only describes the annotator's general style in translating θ into r.)

The **language model**, however, is independent of the feature set θ. It models what rationales tend to look like in the input domain—e.g., documents or images. In the document case, ϕ should describe: How frequent and how long are typical rationales? Do their edges tend to align with punctuation or major syntactic boundaries in x? Are they rarer in the middle of a document, or in certain documents?[6]

Thanks to the language model, we do not need to posit high θ features to explain every word in a rationale. The language model can "explain away" some words as having been highlighted only because this annotator prefers not to end a rationale in mid-phrase, or prefers to sweep up close-together features with a single long rationale rather than many short ones. Similarly, the language model can help explain why some words, though important, might *not* have been included in any rationale of r.

If there are multiple annotators, one can learn different ϕ parameters for each annotator, reflecting their different annotation styles.[7] We found this to be useful (section 8.2).

We remark that our generative modeling approach (equation (1)) would also apply if r were not rationale markup, but some other kind of so-called "side information," such as the *feature* annotations discussed in section 1. For example, Raghavan et al. (2006) assume that if feature h is relevant—a binary distinction—iff it was selected in at least one document. But it might be more informative to observe that h was selected in 3 of the 10 documents where it appeared, and to predict this via a model $p_\phi(3 \text{ of } 10 \mid \theta_h)$, where ϕ describes (e.g.) how to derive a binomial parameter nonlinearly from θ_h. This approach would not *how often* h was marked and infer *how relevant* is feature h (i.e., infer θ_h). In this case, p_ϕ is a simple channel that transforms relevant features into direct indicators of the feature. Our side information merely requires a more complex transformation—from relevant features into well-formed rationales, modulated by documents.

4 Experimental Data: Movie Reviews

In Zaidan et al. (2007), we introduced the "Movie Review Polarity Dataset Enriched with Annotator Rationales."[8] It is based on the dataset of Pang and Lee (2004),[9] which consists of 1000 positive and 1000 negative movie reviews, tokenized and divided into 10 folds (F_0–F_9). All our experiments use F_9 as their final blind test set.

The enriched dataset adds rationale annotations produced by an annotator A0, who annotated folds F_0–F_8 of the movie review set with rationales (in the form of textual substrings) that supported the gold-standard classifications. We will use A0's data to determine the improvement of our method over a (log-linear) baseline model without rationales. We also use A0 to compare against the "masking SVM" method and SVM baseline of Zaidan et al. (2007).

Since ϕ can be tuned to a particular annotator, we would also like to know how well this works with data from annotators other than A0. We randomly selected 100 reviews (50 positive and 50 negative) and collected both class and rationale annotation data from each of six new annotators A3–A8,[10] following the same procedures as (Zaidan et al., 2007). We report results using only data from A3–A5, since we used the data from A6–A8 as development data in the early stages of our work.

We use this new rationale-enriched dataset[8] to determine if our method works well across annotators. We will only be able to carry out that comparison

[6] Our current experiments do not model this last point. However, we imagine that if the document only has a few θ-features that support the classification, the annotator will probably mark most of them, whereas if such features are abundant, the annotator may lazily mark only a few of the strongest ones. A simple approach would equip ϕ with a different "bias" or "threshold" parameter ϕ_x for each rationale training document x, to modulate the *a priori* probability of marking a rationale in x. By fitting this bias parameter, we deduce how lazy the annotator was (for whatever reason) on document x. If desired, a prior on ϕ_x could consider whether x has many strong θ-features, whether the annotator has recently had a coffee break, etc.

[7] Given insufficient rationale data to recover some annotator's ϕ well, one could smooth using data from other annotators. But in our situation, ϕ had relatively few parameters to learn.

[8] Available at http://cs.jhu.edu/∼ozaidan/rationales.

[9] Polarity dataset version 2.0.

[10] We avoid annotator names A1–A2, which were already used in (Zaidan et al., 2007).

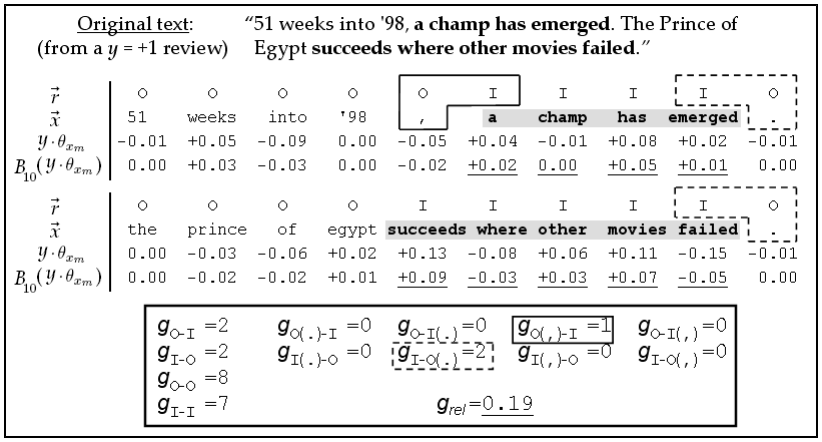

Figure 1: Rationales as sequence annotation: the annotator highlighted two textual segments as rationales for a positive class. Highlighted words in $\vec{x}$ are tagged I in $\vec{r}$, and other words are tagged O. The figure also shows some ϕ-features. For instance, $g_{O(,)\text{-}I}$ is a count of O−I transitions that occur with a comma as the left word. Notice also that g_{rel} is the sum of the underlined values.

at small training set sizes, due to limited data from A3–A8. The larger A0 dataset will still allow us to evaluate our method on a range of training set sizes.

5 Detailed Models

5.1 Modeling class annotations with p_θ

We define the basic classifier p_θ in equation (1) to be a standard conditional log-linear model:

$$p_\theta(y \mid x) \stackrel{\text{def}}{=} \frac{\exp(\vec{\theta} \cdot \vec{f}(x,y))}{Z_\theta(x)} \stackrel{\text{def}}{=} \frac{u(x,y)}{Z_\theta(x)} \quad (2)$$

where $\vec{f}(\cdot)$ extracts a feature vector from a classified document, $\vec{\theta}$ are the corresponding weights of those features, and $Z_\theta(x) \stackrel{\text{def}}{=} \sum_y u(x,y)$ is a normalizer.

We use the same set of binary features as in previous work on this dataset (Pang et al., 2002; Pang and Lee, 2004; Zaidan et al., 2007). Specifically, let $V = \{v_1, ..., v_{17744}\}$ be the set of word types with count ≥ 4 in the full 2000-document corpus. Define $f_h(x,y)$ to be y if v_h appears at least once in x, and 0 otherwise. Thus $\theta \in \mathbb{R}^{17744}$, and positive weights in θ favor class label $y = +1$ and equally discourage $y = -1$, while negative weights do the opposite.

This standard unigram feature set is linguistically impoverished, but serves as a good starting point for studying rationales. Future work should consider more complex features and how *they* are signaled by rationales, as discussed in section 3.2.

5.2 Modeling rationale annotations with p_ϕ

The rationales collected in this task are textual segments of a document to be classified. The document itself is a word token sequence $\vec{x} = x_1, ..., x_M$. We encode its rationales as a corresponding tag sequence $\vec{r} = r_1, ..., r_M$, as illustrated in Figure 1. Here $r_m \in \{\texttt{I}, \texttt{O}\}$ according to whether the token x_m is **in** a rationale (i.e., x_m was at least partly highlighted) or **outside** all rationales. x_1 and x_M are special boundary symbols, tagged with O.

We predict the full tag sequence $\vec{r}$ at once using a conditional random field (Lafferty et al., 2001). A CRF is just another conditional log-linear model:

$$p_\phi(r \,|\, x, y, \vec{\theta}) \stackrel{\text{def}}{=} \frac{\exp(\vec{\phi} \cdot \vec{g}(r,x,y,\vec{\theta}))}{Z_\phi(x,y,\vec{\theta})} \stackrel{\text{def}}{=} \frac{u(r,x,y,\vec{\theta})}{Z_\phi(x,y,\vec{\theta})}$$

where $\vec{g}(\cdot)$ extracts a feature vector, $\vec{\phi}$ are the corresponding weights of those features, and $Z_\phi(x,y,\vec{\theta}) \stackrel{\text{def}}{=} \sum_r u(r,x,y,\vec{\theta})$ is a normalizer.

As usual for linear-chain CRFs, $\vec{g}(\cdot)$ extracts two kinds of features: first-order "emission" features that relate r_m to (x_m, y, θ), and second-order "transition" features that relate r_m to r_{m-1} (although some of these also look at x).

These two kinds of features respectively capture the "channel model" and "language model" of section 3.2. The former says r_m is I because x_m is associated with a relevant θ-feature. The latter says r_m is I simply because it is next to another I.

5.3 Emission ϕ-features ("channel model")

Recall that our θ-features (at present) correspond to unigrams. Given $(\vec{x}, y, \vec{\theta})$, let us say that a unigram $w \in \vec{x}$ is **relevant**, **irrelevant**, or **anti-relevant** if $y \cdot \theta_w$ is respectively $\gg 0$, ≈ 0, or $\ll 0$. That is, w is relevant if its presence in x strongly supports the annotated class y, and anti-relevant if its presence strongly supports the opposite class $-y$.

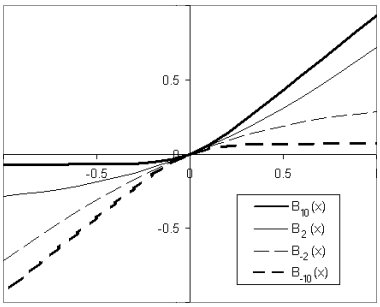

Figure 2: The function family B_s in equation (3), shown for $s \in \{10, 2, -2, -10\}$.

We would like to learn the extent ϕ_{rel} to which annotators try to *include* relevant unigrams in their rationales, and the (usually lesser) extent ϕ_{antirel} to which they try to *exclude* anti-relevant unigrams. This will help us infer $\vec{\theta}$ from the rationales.

The details are as follows. ϕ_{rel} and ϕ_{antirel} are the weights of two emission features extracted by $\vec{g}$:

$$g_{\text{rel}}(\vec{x}, y, \vec{r}, \vec{\theta}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} I(r_m = \texttt{I}) \cdot B_{10}(y \cdot \theta_{x_m})$$

$$g_{\text{antirel}}(\vec{x}, y, \vec{r}, \vec{\theta}) \stackrel{\text{def}}{=} \sum_{m=1}^{M} I(r_m = \texttt{I}) \cdot B_{-10}(y \cdot \theta_{x_m})$$

Here $I(\cdot)$ denotes the indicator function, returning 1 or 0 according to whether its argument is true or false. Relevance and negated anti-relevance are respectively measured by the differentiable nonlinear functions B_{10} and B_{-10}, which are defined by

$$B_s(a) = (\log(1 + \exp(a \cdot s)) - \log(2))/s \quad (3)$$

and graphed in Figure 2. Sample values of B_{10} and g_{rel} are shown in Figure 1.

How does this work? The g_{rel} feature is a sum over all unigrams in the document $\vec{x}$. It does not fire strongly on the irrelevant or anti-relevant unigrams, since B_{10} is close to zero there.[11] But it fires positively on relevant unigrams w if they are tagged with `I`, and the strength of such firing increases approximately linearly with θ_w. Since the weight $\phi_{\text{rel}} > 0$ in practice, this means that raising a relevant unigram's θ_w (if $y = +1$) will proportionately raise its log-odds of being tagged with `I`. Symmetrically, since $\phi_{\text{antirel}} > 0$ in practice, lowering an anti-relevant unigram's θ_w (if $y = +1$) will proportionately lower its log-odds of being tagged with `I`, though not necessarily at the same rate as for relevant unigrams.[12]

Should ϕ also include traditional CRF emission features, which would recognize that particular words like great tend to be tagged as `I`? No! Such features would undoubtedly do a better job predicting the rationales and hence increasing equation (1). However, crucially, our true goal is not to predict the rationales but to recover the classifier parameters θ. Thus, if great tends to be highlighted, then the model should not be permitted to explain this directly by increasing some feature ϕ_{great}, but only indirectly by increasing θ_{great}. We therefore permit our rationale prediction model to consider only the two emission features g_{rel} and g_{antirel}, which see the words in $\vec{x}$ only through their θ-values.

5.4 Transition ϕ-features ("language model")

Annotators highlight more than just the relevant unigrams. (After all, they aren't told that our current θ-features are unigrams.) They tend to mark full phrases, though perhaps taking care to exclude anti-relevant portions. ϕ models these phrases' shape, via weights for several "language model" features.

Most important are the 4 traditional CRF tag transition features $g_{\texttt{O-O}}, g_{\texttt{O-I}}, g_{\texttt{I-I}}, g_{\texttt{I-O}}$. For example, $g_{\texttt{O-I}}$ *counts* the number of `O`-to-`I` transitions in $\vec{r}$ (see Figure 1). Other things equal, an annotator with high $\phi_{\texttt{O-I}}$ is predicted to have many rationales per 1000 words. And if $\phi_{\texttt{I-I}}$ is high, rationales are predicted to be long phrases (including more irrelevant unigrams around or between the relevant ones).

We also learn more refined versions of these features, which consider how the transition probabilities are influenced by the punctuation and syntax of the document $\vec{x}$ (independent of $\vec{\theta}$). These refined features are more specific and hence more sparsely trained. Their weights reflect deviations from the simpler, "backed-off" transition features such as $g_{\texttt{O-I}}$. (Again, see Figure 1 for examples.)

Conditioning on left word. A feature of the form $g_{t_1(v)\text{-}t_2}$ is specified by a pair of tag types $t_1, t_2 \in \{\texttt{I}, \texttt{O}\}$ and a vocabulary word type v. It counts the

[11] B_{10} sets the threshold for relevance to be about 0. One could also include versions of the g_{rel} feature that set a higher threshold, using $B_{10}(y \cdot \theta_{x_m} - \text{threshold})$.

[12] If the two rates *are* equal ($\phi_{\text{rel}} = \phi_{\text{antirel}}$), we get a simpler model in which the log-odds change exactly linearly with θ_w for each w, regardless of w's relevance/irrelevance/anti-relevance. This follows from the fact that $B_s(a) + B_{-s}(a)$ simplifies to a.

number of times an t_1–t_2 transition occurs in $\vec{r}$ conditioned on v appearing as the first of the two word tokens where the transition occurs. Our experiments include $g_{t_1(v)\text{-}t_2}$ features that tie I−O and O−I transitions to the 4 most frequent punctuation marks v (comma, period, ?, !).

Conditioning on right word. A feature $g_{t_1\text{-}t_2(v)}$ is similar, but v must appear as the second of the two word tokens where the transition occurs. Again here, we use $g_{t_1\text{-}t_2(v)}$ features that tie I−O and O−I transitions to the four punctuation marks mentioned above. We also include five features that tie O−I transitions to the words *no*, *not*, *so*, *very*, and *quite*, since in our development data, those words were more likely than others to start rationales.[13]

Conditioning on syntactic boundary. We parsed each rationale-annotated training document (no parsing is needed at test time).[14] We then marked each word bigram x_1-x_2 with three nonterminals: N_{End} is the nonterminal of the largest constituent that contains x_1 and not x_2, N_{Start} is the nonterminal of the largest constituent that contains x_2 and not x_1, and N_{Cross} is the nonterminal of the *smallest* constituent that contains both x_1 and x_2.

For a nonterminal N and pair of tag types (t_1, t_2), we define three features, $g_{t_1\text{-}t_2/\text{E}=N}$, $g_{t_1\text{-}t_2/\text{S}=N}$, and $g_{t_1\text{-}t_2/\text{C}=N}$, which count the number of times a t_1-t_2 transition occurs in $\vec{r}$ with N matching the N_{End}, N_{Start}, or N_{Cross} nonterminal, respectively. Our experiments include these features for 11 common nonterminal types N (DOC, TOP, S, SBAR, FRAG, PRN, NP, VP, PP, ADJP, QP).

6 Training: Joint Optimization of θ and ϕ

To train our model, we use L-BFGS to locally maximize the log of the objective function (1):[15]

$$\sum_{i=1}^{n} \log p_\theta(y_i \mid x_i) - \frac{1}{2\sigma_\theta^2}\|\theta\|^2$$
$$+C\Big(\sum_{i=1}^{n} \log p_\phi(r_i \mid x_i, y_i, \theta)\Big) - \frac{1}{2\sigma_\phi^2}\|\phi\|^2 \qquad (4)$$

This defines p_{prior} from (1) to be a standard diagonal Gaussian prior, with variances σ_θ^2 and σ_ϕ^2 for the two sets of parameters. We optimize σ_θ^2 in our experiments. As for σ_ϕ^2, different values did not affect the results, since we have a large number of {I,O} rationale tags to train relatively few ϕ weights; so we simply use $\sigma_\phi^2 = 1$ in all of our experiments.

Note the new C factor in equation (4). Our initial experiments showed that optimizing equation (4) without C led to an increase in the likelihood of the rationale data at the expense of classification accuracy, which degraded noticeably. This is because the second sum in (4) has a much larger magnitude than the first: in a set of 100 documents, it predicts around 74,000 binary {I,O} tags, versus the one hundred binary class labels. While we are willing to reduce the log-likelihood of the training classifications (the first sum) to a certain extent, focusing too much on modeling rationales (the second sum) is clearly not our ultimate goal, and so we optimize C on development data to achieve some balance between the two terms of equation (4). Typical values of C range from $\frac{1}{300}$ to $\frac{1}{50}$.[16]

We perform alternating optimization on θ and ϕ:

1. Initialize θ to maximize equation (4) but with $C = 0$ (i.e. based only on class data).
2. Fix θ, and find ϕ that maximizes equation (4).
3. Fix ϕ, and find θ that maximizes equation (4).
4. Repeat 2 and 3 until convergence.

The L-BFGS method requires calculating the gradient of the objective function (4). The partial derivatives with respect to components of θ and ϕ involve calculating expectations of the feature functions, which can be computed in linear time (with respect to the size of the training set) using the forward-backward algorithm for CRFs. The partial derivatives also involve the derivative of (3), to determine how changing θ will affect the firing strength of the emission features g_{rel} and g_{antirel}.

[13] These are the function words with count ≥ 40 in a random sample of 100 documents, and which were associated with the O−I tag transition at more than twice the average rate. We do not use any other lexical ϕ-features that reference $\vec{x}$, for fear that they would enable the learner to explain the rationales without changing θ as desired (see the end of section 5.3).

[14] We parse each sentence with the Collins parser (Collins, 1999). Then the document has one big parse tree, whose root is DOC, with each sentence being a child of DOC.

[15] One might expect this function to be convex because p_θ and p_ϕ are both log-linear models with no hidden variables. However, $\log p_\phi(r_i \mid x_i, y_i, \theta)$ is not necessarily convex in θ.

[16] C also balances our confidence in the classifications y against our confidence in the rationales r; either may be noisy.

7 Experimental Procedures

We report on two sets of experiments. In the first set, we use the annotation data that A3–A5 provided for the small set of 100 documents (as well as the data from A0 on those same 100 documents). In the second set, we used A0's abundant annotation data to evaluate our method with training set sizes up to 1600 documents, and compare it with three other methods: log-linear baseline, SVM baseline, and the SVM masking method of (Zaidan et al., 2007).

7.1 Learning curves

The learning curves reported in section 8.1 are generated exactly as in (Zaidan et al., 2007). Each curve shows classification accuracy at training set sizes $T = 1, 2, ..., 9$ folds (i.e. $200, 400, ..., 1600$ training documents). For a given size T, the reported accuracy is an average of 9 experiments with different subsets of the entire training set, each of size T:

$$\frac{1}{9}\sum_{i=0}^{8} acc(F_9 \mid F_{i+1} \cup \ldots \cup F_{i+T}) \qquad (5)$$

where F_j denotes the fold numbered $j \bmod 9$, and $acc(F_9 \mid Y)$ means classification accuracy on the held-out test set F_9 after training on set Y.

We use an appropriate paired permutation test, detailed in (Zaidan et al., 2007), to test differences in (5). We call a difference significant at $p < 0.05$.

7.2 Comparison to "masking SVM" method

We compare our method to the "masking SVM" method of (Zaidan et al., 2007). Briefly, that method used rationales to construct several so-called *contrast examples* from every training example. A contrast example is obtained by "masking out" one of the rationales highlighted to support the training example's class. A *good* classifier should have more trouble on this modified example. Hence, Zaidan et al. (2007) required the learned SVM to classify each contrast example with a smaller margin than the corresponding original example (and did not require it to be classified correctly).

The masking SVM learner relies on a simple geometric principle; is trivial to implement on top of an existing SVM learner; and works well. However, we believe that the generative method we present here is more interesting and should apply more broadly.

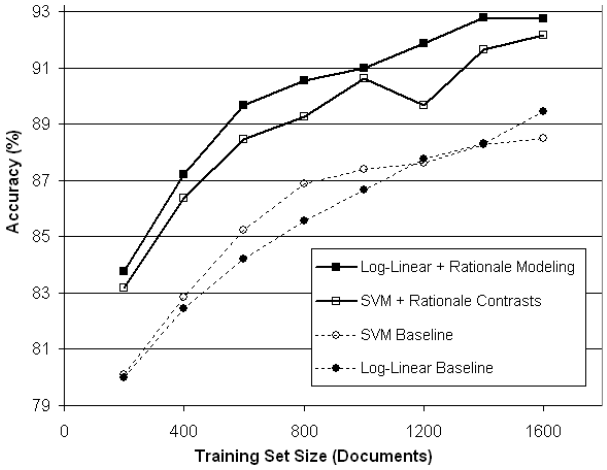

Figure 3: Classification accuracy curves for the 4 methods: the two baseline learners that only utilize class data, and the two learners that also utilize rationale annotations. The SVM curves are from (Zaidan et al., 2007).

First, the masking method is specific to improving an SVM learner, whereas our method can be used to improve any classifier by adding a rationale-based regularizer (the second half of equation (4)) to its objective function during training.

More important, there are tasks where it is unclear how to generate contrast examples. For the movie review task, it was natural to mask out a rationale by pretending its words never occurred in the document. After all, most word types do not appear in most documents, so it is natural to consider the non-presence of a word as a "default" state to which we can revert. But in an image classification task, how should one modify the image's features to ignore some spatial region marked as a rationale? There is usually no natural "default" value to which we could set the pixels. Our method, on the other hand, eliminates contrast examples altogether.

8 Experimental Results and Analysis

8.1 The added benefit of rationales

Fig. 3 shows learning curves for four methods. A log-linear model shows large and significant improvements, at all training sizes, when we incorporate rationales into its training via equation (4). Moreover, the resulting classifier consistently outperforms[17] prior work, the masking SVM, which starts with a slightly better baseline classifier (an SVM) but incorporates the rationales more crudely.

[17] Differences are not significant at sizes 200, 1000, and 1600.

	size	A0	A3	A4	A5
SVM baseline	100	72.0	72.0	72.0	70.0
SVM+contrasts	100	*75.0*	*73.0*	*74.0*	*72.0*
Log-linear baseline	100	71.0	*73.0*	71.0	70.0
Log-linear+rats	100	**76.0**	**76.0**	**77.0**	**74.0**
SVM baseline	20	63.4	*62.2*	60.4	62.6
SVM+contrasts	20	*65.4*	*63.4*	*62.4*	**64.8**
Log-linear baseline	20	63.0	*62.2*	60.2	62.4
Log-linear+rats	20	**65.8**	**63.6**	**63.4**	**64.8**

Table 1: Accuracy rates using each annotator's data. In a given column, a value in *italics* is not significantly different from the highest value in that column, which is **boldfaced**. The size=20 results average over 5 experiments.

To confirm that we could successfully model annotators other than A0, we performed the same comparison for annotators A3–A5; each had provided class and rationale annotations on a small 100-document training set. We trained a separate ϕ for each annotator. Table 1 shows improvements over baseline, usually significant, at 2 training set sizes.

8.2 Analysis

Examining the learned weights $\vec{\phi}$ gives insight into annotator behavior. High weights include `I-O` and `O-I` transitions conditioned on punctuation, e.g., $\phi_{\texttt{I(.)-O}} = 3.55$,[18] as well as rationales ending at the end of a major phrase, e.g., $\phi_{\texttt{I-O/E=VP}} = 1.88$.

The large emission feature weights, e.g., $\phi_{\text{rel}} = 14.68$ and $\phi_{\text{antirel}} = 15.30$, tie rationales closely to θ values, as hoped. For example, in Figure 1, the word $w =$ succeeds, with $\theta_w = 0.13$, drives up $p(\texttt{I})/p(\texttt{O})$ by a factor of 7 (in a positive document) relative to a word with $\theta_w = 0$.

In fact, feature ablation experiments showed that almost all the classification benefit from rationales can be obtained by using only these 2 emission ϕ-features and the 4 unconditioned transition ϕ-features. Our full ϕ (115 features) merely improves our ability to predict the rationales (whose likelihood does increase significantly with more features).

We also checked that annotators' styles differ enough that it helps to tune ϕ to the "target" annotator A who gave the rationales. Table 3 shows that a ϕ model trained on A's *own* rationales does best at predicting new rationales from A. Table 2 shows that as a result, classification performance on the test set is usually best if it was A's *own* ϕ that was used to help learn θ from A's rationales. In both cases, however, a different annotator's ϕ is better than nothing.

[18] When trained on folds F_4–F_8 with A0's rationales.

	ϕ_{A0}	ϕ_{A3}	ϕ_{A4}	ϕ_{A5}	Baseline
θ_{A0}	**76.0**	*73.0*	*74.0*	*73.0*	71.0
θ_{A3}	*73.0*	**76.0**	*74.0*	*73.0*	*73.0*
θ_{A4}	*75.0*	73.0	**77.0**	*74.0*	71.0
θ_{A5}	**74.0**	*71.0*	*72.0*	**74.0**	70.0

Table 2: Accuracy rate for an annotator's θ (rows) obtained when using some other annotator's ϕ (columns). Notice that the diagonal entries and the baseline column are taken from rows of Table 1 (size=100).

	ϕ_{A0}	ϕ_{A3}	ϕ_{A4}	ϕ_{A5}	Trivial model
$-L(r_{A0})$	**0.073**	0.086	*0.077*	0.088	0.135
$-L(r_{A3})$	0.084	**0.068**	*0.071*	**0.068**	0.130
$-L(r_{A4})$	0.088	0.084	**0.075**	0.085	0.153
$-L(r_{A5})$	0.058	**0.044**	*0.047*	**0.044**	0.111

Table 3: Cross-entropy per tag of rationale annotations $\vec{r}$ for each annotator (rows), when predicted from that annotator's $\vec{x}$ and $\vec{\theta}$ via a possibly different annotator's ϕ (columns). For comparison, the trivial model is a bigram model of $\vec{r}$, which is trained on the target annotator but ignores $\vec{x}$ and $\vec{\theta}$. 5-fold cross-validation on the 100-document set was used to prevent testing on training data.

9 Conclusions

We have demonstrated a effective method for eliciting extra knowledge from *naive* annotators, in the form of lightweight "rationales" for their annotations. By explicitly modeling the annotator's rationale-marking process, we are able to infer a better model of the original annotations.

We showed that our method performs significantly better than two strong baseline classifiers, and also outperforms our previous discriminative method for exploiting rationales (Zaidan et al., 2007). We also saw that it worked across four annotators who have different rationale-marking styles.

In future, we are interested in new domains that can adaptively solicit rationales for some or all training examples. Our new method, being essentially Bayesian inference, is potentially extensible to many other situations—other tasks, classifier architectures, and more complex features.

References

Eric Brill and Grace Ngai. 1999. Man [and woman] vs. machine: A case study in base noun phrase learning. In *Proceedings of the 37th ACL Conference*.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

G. Druck, G. Mann, and A. McCallum. 2008. Learning from labeled features using generalized expectation criteria. In *Proceedings of ACM Special Interest Group on Information Retrieval, (SIGIR)*.

A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA, June. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 117–125, Hong Kong.

B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proc. of ACL*, pages 271–278.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proc. of EMNLP*, pages 79–86.

Hema Raghavan and James Allan. 2007. An interactive algorithm for asking and incorporating feature feedback into support vector machines. In *Proceedings of SIGIR*.

Hema Raghavan, Omid Madani, and Rosie Jones. 2006. Active learning on both features and instances. *Journal of Machine Learning Research*, 7:1655–1686, Aug.

Luis von Ahn, Ruoran Liu, and Manuel Blum. 2006. Peekaboom: A game for locating objects. In *CHI '06: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 55–64.

Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using "annotator rationales" to improve machine learning for text categorization. In *NAACL HLT 2007; Proceedings of the Main Conference*, pages 260–267, April.