Using "Annotator Rationales" to Improve Machine Learning for Text Categorization*

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

Christine D. Piatko
JHU Applied Physics Laboratory
11100 Johns Hopkins Road
Laurel, MD 20723 USA
christine.piatko@jhuapl.edu

Abstract

We propose a new framework for supervised machine learning. Our goal is to learn from smaller amounts of supervised training data, by collecting a richer *kind* of training data: annotations with "rationales." When annotating an example, the human teacher will also highlight evidence supporting this annotation—thereby teaching the machine learner *why* the example belongs to the category. We provide some rationale-annotated data and present a learning method that exploits the rationales during training to boost performance significantly on a sample task, namely sentiment classification of movie reviews. We hypothesize that in some situations, providing rationales is a more fruitful use of an annotator's time than annotating more examples.

1 Introduction

Annotation cost is a bottleneck for many natural language processing applications. While supervised machine learning systems are effective, it is labor-intensive and expensive to construct the many training examples needed. Previous research has explored active or semi-supervised learning as possible ways to lessen this burden.

We propose a new way of breaking this annotation bottleneck. Annotators currently indicate *what* the correct answers are on training data. We propose that they should also indicate *why*, at least by coarse hints. We suggest new machine learning approaches that can benefit from this "why" information.

For example, an annotator who is categorizing phrases or documents might also be asked to highlight a few substrings that significantly influenced her judgment. We call such clues "rationales." They need not correspond to machine learning features.

In some circumstances, rationales should not be too expensive or time-consuming to collect. As long as the annotator is spending the time to study example x_i and classify it, it may not require much extra effort for her to mark reasons for her classification.

2 Using Rationales to Aid Learning

We will not rely exclusively on the rationales, but use them only as an added source of information. The idea is to help direct the learning algorithm's attention—helping it tease apart signal from noise.

Machine learning algorithms face a well-known "credit assignment" problem. Given a complex datum x_i and the desired response y_i, many features of x_i could be responsible for the choice of y_i. The learning algorithm must tease out which features were *actually* responsible. This requires a lot of training data, and often a lot of computation as well.

Our rationales offer a shortcut to solving this "credit assignment" problem, by providing the learning algorithm with hints as to which features of x_i were relevant. Rationales should help guide the learning algorithm toward the correct classification function, by pushing it toward a function that correctly pays attention to each example's relevant features. This should help the algorithm learn from less data and avoid getting trapped in local maxima.[1]

In this paper, we demonstrate the "annotator rationales" technique on a text categorization problem previously studied by others.

*This work was supported by the JHU WSE/APL Partnership Fund; National Science Foundation grant No. 0347822 to the second author; and an APL Hafstad Fellowship to the third.

[1] To understand the local maximum issue, consider the hard problem of training a standard 3-layer feed-forward neural network. If the activations of the "hidden" layer's features (nodes) were observed at training time, then the network would decompose into a pair of independent 2-layer perceptrons. This turns an NP-hard problem with local maxima (Blum and Rivest, 1992) to a polytime-solvable convex problem. Although rationales might only provide *indirect* evidence of the hidden layer, this would still modify the objective function (see section 8) in a way that tended to make the correct weights easier to discover.

Proceedings of NAACL HLT 2007, pages 260–267,
Rochester, NY, April 2007.

3 Discriminative Approach

One popular approach for text categorization is to use a discriminative model such as a Support Vector Machine (SVM) (e.g. (Joachims, 1998; Dumais, 1998)). We propose that SVM training can in general incorporate annotator rationales as follows.

From the rationale annotations on a positive example $\overrightarrow{x_i}$, we will construct one or more "not-quite-as-positive" *contrast examples* $\overrightarrow{v_{ij}}$. In our text categorization experiments below, each contrast document $\overrightarrow{v_{ij}}$ was obtained by starting with the original and "masking out" one or all of the several rationale substrings that the annotator had highlighted (r_{ij}). The intuition is that the *correct* model should be less sure of a positive classification on the contrast example $\overrightarrow{v_{ij}}$ than on the original example $\vec{x_i}$, because $\overrightarrow{v_{ij}}$ lacks evidence that the annotator found significant.

We can translate this intuition into additional constraints on the correct model, i.e., on the weight vector $\vec{w}$. In addition to the usual SVM constraint on positive examples that $\vec{w} \cdot \overrightarrow{x_i} \geq 1$, we also want (for each j) that $\vec{w} \cdot \vec{x}_i - \vec{w} \cdot \overrightarrow{v_{ij}} \geq \mu$, where $\mu \geq 0$ controls the size of the desired margin between original and contrast examples.

An ordinary soft-margin SVM chooses $\vec{w}$ and $\vec{\xi}$ to minimize

$$\frac{1}{2}\|\vec{w}\|^2 + C(\sum_i \xi_i) \quad (1)$$

subject to the constraints

$$(\forall i) \;\; \vec{w} \cdot \overrightarrow{x_i} \cdot y_i \;\; \geq \;\; 1 - \xi_i \quad (2)$$

$$(\forall i) \;\; \xi_i \;\; \geq \;\; 0 \quad (3)$$

where $\overrightarrow{x_i}$ is a training example, $y_i \in \{-1, +1\}$ is its desired classification, and ξ_i is a slack variable that allows training example $\overrightarrow{x_i}$ to miss satisfying the margin constraint if necessary. The parameter $C > 0$ controls the cost of taking such slack, and should generally be lower for noisier or less linearly separable datasets. We add the *contrast constraints*

$$(\forall i, j) \;\; \vec{w} \cdot (\overrightarrow{x_i} - \overrightarrow{v_{ij}}) \cdot y_i \geq \mu(1 - \xi_{ij}), \quad (4)$$

where $\overrightarrow{v_{ij}}$ is one of the contrast examples constructed from example $\overrightarrow{x_i}$, and $\xi_{ij} \geq 0$ is an associated slack variable. Just as these extra constraints have their own margin μ, their slack variables have their own cost, so the objective function (1) becomes

$$\frac{1}{2}\|\vec{w}\|^2 + C(\sum_i \xi_i) + C_{contrast}(\sum_{i,j} \xi_{ij}) \quad (5)$$

The parameter $C_{contrast} \geq 0$ determines the importance of satisfying the contrast constraints. It should generally be less than C if the contrasts are noisier than the training examples.[2]

In practice, it is possible to solve this optimization using a standard soft-margin SVM learner. Dividing equation (4) through by μ, it becomes

$$(\forall i, j) \;\; \vec{w} \cdot \overrightarrow{x_{ij}} \cdot y_i \geq 1 - \xi_{ij}, \quad (6)$$

where $\overrightarrow{x_{ij}} \stackrel{\text{def}}{=} \frac{\overrightarrow{x_i} - \overrightarrow{v_{ij}}}{\mu}$. Since equation (6) takes the same form as equation (2), we simply add the pairs $(\overrightarrow{x_{ij}}, y_i)$ to the training set as *pseudoexamples*, weighted by $C_{contrast}$ rather than C so that the learner will use the objective function (5).

There is one subtlety. To allow a biased hyperplane, we use the usual trick of prepending a 1 element to each training example. Thus we require $\vec{w} \cdot (1, \overrightarrow{x_i}) \geq 1 - \xi_i$ (which makes w_0 play the role of a bias term). This means, however, that we must prepend a 0 element to each pseudoexample: $\vec{w} \cdot \frac{(1,\vec{x}_i) - (1,\overrightarrow{v_{ij}})}{\mu} = \vec{w} \cdot (0, \overrightarrow{x_{ij}}) \geq 1 - \xi_{ij}$.

In our experiments, we optimize μ, C, and $C_{contrast}$ on held-out data (see section 5.2).

4 Rationale Annotation for Movie Reviews

In order to demonstrate that annotator rationales help machine learning, we needed annotated data that included rationales for the annotations.

We chose a dataset that would be enjoyable to reannotate: the movie review dataset of (Pang et al., 2002; Pang and Lee, 2004).[3] The dataset consists of 1000 positive and 1000 negative movie reviews obtained from the Internet Movie Database (IMDb) review archive, all written before 2002 by a total of 312 authors, with a cap of 20 reviews per author per

[2] Taking $C_{contrast}$ to be constant means that all rationales are equally valuable. One might instead choose, for example, to reduce $C_{contrast}$ for examples x_i that have *many* rationales, to prevent x_i's contrast examples v_{ij} from together dominating the optimization. However, in this paper we assume that an x_i with more rationales really does provide more evidence about the true classifier $\vec{w}$.

[3] Polarity dataset version 2.0.

category. Pang and Lee have divided the 2000 documents into 10 folds, each consisting of 100 positive reviews and 100 negative reviews.

The dataset is arguably artificial in that it keeps only reviews where the reviewer provided a rather high or rather low numerical rating, allowing Pang and Lee to designate the review as positive or negative. Nonetheless, most reviews contain a difficult mix of praise, criticism, and factual description. In fact, it is possible for a mostly critical review to give a positive overall recommendation, or vice versa.

4.1 Annotation procedure

Rationale annotators were given guidelines[4] that read, in part:

> Each review was intended to give either a positive or a negative overall recommendation. You will be asked to justify why a review is positive or negative. To justify why a review is positive, highlight the most important words and phrases that would tell someone to see the movie. To justify why a review is negative, highlight words and phrases that would tell someone not to see the movie. These words and phrases are called **rationales**.
>
> You can highlight the rationales as you notice them, which should result in several rationales per review. Do your best to mark enough rationales to provide convincing support for the class of interest.
>
> You do not need to go out of your way to mark everything. You are probably doing too much work if you find yourself going back to a paragraph to look for even more rationales in it. Furthermore, it is perfectly acceptable to skim through sections that you feel would not contain many rationales, such as a reviewer's plot summary, even if that might cause you to miss a rationale here and there.

The last two paragraphs were intended to provide some guidance on how *many* rationales to annotate. Even so, as section 4.2 shows, some annotators were considerably more thorough (and slower).

Annotators were also shown the following examples[5] of positive rationales:

- **you will enjoy the hell out of** American Pie.
- fortunately, they **managed to do it in an interesting and funny way**.
- he is **one of the most exciting martial artists on the big screen**, continuing to perform his own stunts and **dazzling audiences** with his flashy kicks and punches.
- the romance was **enchanting**.

and the following examples[5] of negative rationales:

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

[5] For our controlled study of annotation time (section 4.2), different examples were given with full document context.

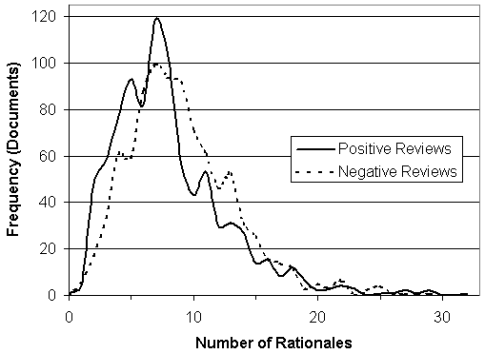

Figure 1: Histograms of rationale counts per document (A0's annotations). The overall mean of 8.55 is close to that of the four annotators in Table 1. The median and mode are 8 and 7.

- A woman in peril. A confrontation. An explosion. The end. **Yawn. Yawn. Yawn.**
- when a film makes watching Eddie Murphy **a tedious experience, you know something is terribly wrong**.
- the movie is **so badly put together** that even the most casual viewer may notice the **miserable pacing and stray plot threads**.
- **don't go see** this movie

The annotation involves **boldfacing** the rationale phrases using an HTML editor. Note that a fancier annotation tool would be necessary for a task like named entity tagging, where an annotator must mark many named entities in a single document. At any given moment, such a tool should allow the annotator to highlight, view, and edit only the several rationales for the "current" annotated entity (the one most recently annotated or re-selected).

One of the authors (A0) annotated folds 0–8 of the movie review set (1,800 documents) with rationales that supported the gold-standard classifications. This training/development set was used for all of the learning experiments in sections 5–6. A histogram of rationale counts is shown in Figure 1. As mentioned in section 3, the rationale annotations were just textual substrings. The annotator did not require knowledge of the classifier features. Thus, our rationale dataset is a new resource[4] that could also be used to study exploitation of rationales under feature sets or learning methods other than those considered here (see section 8).

4.2 Inter-annotator agreement

To study the annotation process, we randomly selected 150 documents from the dataset. The doc-

	Rationales per document	% rationales also annotated by A1	% rationales also annotated by A2	% rationales also annotated by AX	% rationales also annotated by **AY**	% rationales also ann. by **anyone else**
A1	5.02	(100)	69.6	63.0	80.1	91.4
A2	10.14	42.3	(100)	50.2	67.8	80.9
AX	6.52	49.0	68.0	(100)	79.9	90.9
AY	11.36	39.7	56.2	49.3	(100)	75.5

Table 1: Average number of rationales and inter-annotator agreement for Tasks 2 and 3. A rationale by Ai ("I think **this is a great** movie!") is considered to have been annotated also by Aj if at least one of Aj's rationales overlaps it ("I think this is a **great movie!**"). In computing pairwise agreement on rationales, we ignored documents where Ai and Aj disagreed on the class. Notice that the most thorough annotator **AY** caught most rationales marked by the others (exhibiting high "recall"), and that most rationales enjoyed some degree of consensus, especially those marked by the least thorough annotator **A1** (exhibiting high "precision").

uments were split into three groups, each consisting of 50 documents (25 positive and 25 negative). Each subset was used for one of three tasks:[6]

- **Task 1:** Given the document, annotate only the class (positive/negative).
- **Task 2:** Given the document and its class, annotate some rationales for that class.
- **Task 3:** Given the document, annotate both the class and some rationales for it.

We carried out a pilot study (annotators AX and AY: two of the authors) and a later, more controlled study (annotators A1 and A2: paid students). The latter was conducted in a more controlled environment where both annotators used the same annotation tool and annotation setup as each other. Their guidelines were also more detailed (see section 4.1). In addition, the documents for the different tasks were interleaved to avoid any practice effect.

The annotators' classification accuracies in Tasks 1 and 3 (against Pang & Lee's labels) ranged from 92%–97%, with 4-way agreement on the class for 89% of the documents, and pairwise agreement also ranging from 92%–97%. Table 1 shows how many rationales the annotators provided and how well their rationales agreed.

Interestingly, in Task 3, four of **AX's rationales for a positive class** were also partially highlighted by AY as support for AY's (incorrect) *negative* classifications, such as:

[6]Each task also had a "warmup" set of 10 documents to be annotated before that tasks's 50 documents. Documents for Tasks 2 and 3 would automatically open in an HTML editor while Task 1 documents opened in an HTML viewer with no editing option. The annotators recorded their classifications for Tasks 1 and 3 on a spreadsheet.

min./KB	A1 time	A2 time	AX time	AY time
Task 1	0.252	0.112	0.150	0.422
Task 2	0.396	0.537	0.242	0.626
Task 3	0.399	0.505	0.288	1.01
min./doc.	A1 time	A2 time	AX time	AY time
Task 1	1.04	0.460	0.612	1.73
min./rat.	A1 time	A2 time	AX time	AY time
Task 2	0.340	0.239	0.179	0.298
Task 3	0.333	0.198	0.166	0.302

Table 2: Average annotation rates on each task.

- **Even with its numerous flaws, the movie all comes together**, if only for those who ...
- "Beloved" acts like **an incredibly difficult chamber drama paired with a ghost story**.

4.3 Annotation time

Average annotation times are in Table 2. As hoped, rationales did not take too much extra time for most annotators to provide. For each annotator except A2, providing rationales only took roughly twice the time (Task 3 vs. Task 1), even though it meant marking an average of 5–11 rationales in addition to the class.

Why this low overhead? Because marking the class already required the Task 1 annotator to read the document and find some rationales, even if s/he did not mark them. The only extra work in Task 3 is in making them explicit. This synergy between class annotation and rationale annotation is demonstrated by the fact that doing both at once (Task 3) was faster than doing them separately (Tasks 1+2).

We remark that this task—binary classification on full documents—seems to be almost a worst-case scenario for the annotation of rationales. At a purely mechanical level, it was rather heroic of A0 to attach 8–9 new rationale phrases r_{ij} to every bit y_i of ordinary annotation. Imagine by contrast a more local task of identifying entities or relations. Each

lower-level annotation y_i will tend to have fewer rationales r_{ij}, while y_i itself will be more complex and hence more difficult to mark. Thus, we expect that the overhead of collecting rationales will be less in many scenarios than the factor of 2 we measured.

Annotation overhead could be further reduced. For a multi-class problem like relation detection, one could ask the annotator to provide rationales *only* for the rarer classes. This small amount of extra time where the data is sparsest would provide extra guidance where it was most needed. Another possibility is passive collection of rationales via eye tracking.

5 Experimental Procedures

5.1 Feature extraction

Although this dataset seems to demand discourse-level features that contextualize bits of praise and criticism, we exactly follow Pang et al. (2002) and Pang and Lee (2004) in merely using binary unigram features, corresponding to the 17,744 unstemmed word or punctuation types with count ≥ 4 in the full 2000-document corpus. Thus, each document is reduced to a 0-1 vector with 17,744 dimensions, which is then normalized to unit length.[7]

We used the method of section 3 to place additional constraints on a linear classifier. Given a training document, we create several contrast documents, each by deleting exactly one rationale substring from the training document. Converting documents to feature vectors, we obtained an original example $\overrightarrow{x_i}$ and several contrast examples $\overrightarrow{v_{i1}}, \overrightarrow{v_{i2}}, \ldots$.[8] Again, our training method required each original document to be classified more confidently (by a margin μ) than its contrast documents.

If we were using more than unigram features, then simply *deleting* a rationale substring would not always be the best way to create a contrast document, as the resulting ungrammatical sentences might cause deep feature extraction to behave strangely (e.g., parse errors during preprocessing). The goal in creating the contrast document is merely to suppress features (n-grams, parts of speech, syntactic dependencies ...) that depend in part on material in one or more rationales. This could be done directly by modifying the feature extractors, or if one prefers to use existing feature extractors, by "masking" rather than deleting the rationale substring—e.g., replacing each of its word tokens with a special MASK token that is treated as an out-of-vocabulary word.

5.2 Training and testing procedures

We transformed this problem to an SVM problem (see section 3) and applied SVM^{light} for training and testing, using the default linear kernel. We used only A0's rationales and the true classifications.

Fold 9 was reserved as a test set. All accuracy results reported in the paper are the result of testing on fold 9, after training on subsets of folds 0–8.

Our learning curves show accuracy after training on $T < 9$ folds (i.e., $200T$ documents), for various T. To reduce the noise in these results, the accuracy we report for training on T folds is actually the average of 9 different experiments with different (albeit overlapping) training sets that cover folds 0–8:

$$\frac{1}{9}\sum_{i=0}^{8} acc(F_9 \mid \theta^*, F_{i+1} \cup \ldots \cup F_{i+T}) \qquad (7)$$

where F_j denotes the fold numbered $j \bmod 9$, and $acc(Z \mid \theta, Y)$ means classification accuracy on the set Z after training on Y with hyperparameters θ.

To evaluate whether two different training methods A and B gave significantly different average-accuracy values, we used a paired permutation test (generalizing a sign test). The test assumes independence among the 200 test examples but not among the 9 overlapping training sets. For each of the 200 test examples in fold 9, we measured (a_i, b_i), where a_i (respectively b_i) is the number of the 9 training sets under which A (respectively B) classified the example correctly. The p value is the probability that the absolute difference between the average-accuracy values would reach or exceed the observed absolute difference, namely $|\frac{1}{200}\sum_{i=1}^{200}\frac{a_i - b_i}{9}|$, if each (a_i, b_i) had an independent 1/2 chance of being replaced with (b_i, a_i), as per the null hypothesis that A and B are indistinguishable.

For any given value of T and any given training method, we chose hyperparameters $\theta^* =$

[7] The vectors are normalized *before* prepending the 1 corresponding to the bias term feature (mentioned in section 3).

[8] The contrast examples were not normalized to precisely unit length, but instead were normalized by the same factor used to normalize $\overrightarrow{x_i}$. This conveniently ensured that the pseudoexamples $\overrightarrow{x_{ij}} \stackrel{\text{def}}{=} \frac{\vec{x}_i - \overrightarrow{v_{ij}}}{\mu}$ were sparse vectors, with 0 coordinates for all words not in the j^{th} rationale.

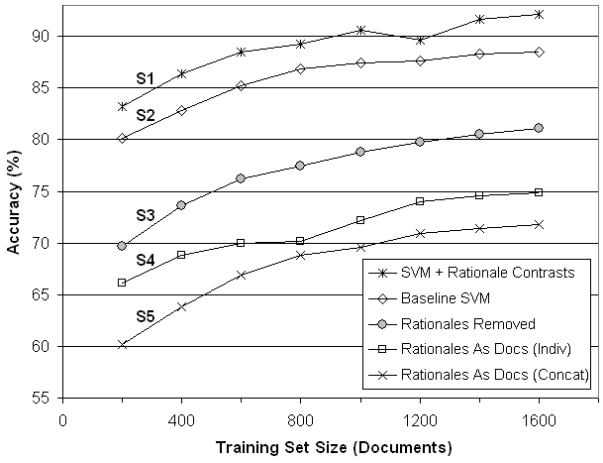

Figure 2: Classification accuracy under five different experimental setups (S1–S5). At each training size, the 5 accuracies are pairwise significantly different (paired permutation test, $p < 0.02$; see section 5.2), except for {S3,S4} or {S4,S5} at some sizes.

$(C, \mu, C_{contrast})$ to maximize the following cross-validation performance:[9]

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{i=0}^{8} acc(F_i \mid \theta, F_{i+1} \cup \ldots \cup F_{i+T}) \quad (8)$$

We used a simple alternating optimization procedure that begins at $\theta_0 = (1.0, 1.0, 1.0)$ and cycles repeatedly through the three dimensions, optimizing along each dimension by a local grid search with resolution 0.1.[10] Of course, when training without rationales, we did not have to optimize μ or $C_{contrast}$.

6 Experimental Results

6.1 The value of rationales

The top curve (S1) in Figure 2 shows that performance does increase when we introduce rationales for the training examples as contrast examples (section 3). S1 is significantly higher than the baseline curve (S2) immediately below it, which trains an ordinary SVM classifier without using rationales. At the largest training set size, rationales raise the accuracy from 88.5% to 92.2%, a 32% error reduction.

[9]One might obtain better performance (across *all* methods being compared) by choosing a separate θ^* for each of the 9 training sets. However, to simulate real limited-data training conditions, one should then find the θ^* for each $\{i, ..., j\}$ using a separate cross-validation within $\{i, ..., j\}$ only; this would slow down the experiments considerably.

[10]For optimizing along the C dimension, one could use the efficient method of Beineke et al. (2004), but not in SVM^{light}.

The lower three curves (S3–S5) show that learning is separately helped by the rationale and the non-rationale portions of the documents. S3–S5 are degraded versions of the baseline S2: they are ordinary SVM classifiers that perform significantly worse than S2 ($p < 0.001$).

Removing the rationale phrases from the training documents (S3) made the test documents much harder to discriminate (compared to S2). This suggests that annotator A0's rationales often covered *most* of the usable evidence for the true class.

However, the pieces to solving the classification puzzle cannot be found solely in the short rationale phrases. Removing all *non*-rationale text from the training documents (S5) was even worse than removing the rationales (S3). In other words, we cannot hope to do well simply by training on just the rationales (S5), although that approach is improved somewhat in S4 by treating each rationale (similarly to S1) as a *separate* SVM training example.

This presents some insight into why our method gives the best performance. The classifier in S1 is able to extract subtle patterns from the corpus, like S2, S3, or any other standard machine learning method, but it is *also* able to learn from a human annotator's decision-making strategy.

6.2 Using fewer rationales

In practice, one might annotate rationales for only *some* training documents—either when annotating a new corpus or when adding rationales *post hoc* to an existing corpus. Thus, a range of options can be found between curves S2 and S1 of Figure 2.

Figure 3 explores this space, showing how far the learning curve S2 moves upward if one has time to annotate rationales for a fixed number of documents R. The key useful discovery is that much of the benefit can actually be obtained with relatively few rationales. For example, with 800 training documents, annotating (0%, 50%, 100%) of them with rationales gives accuracies of (86.9%, 89.2%, 89.3%). With the maximum of 1600 training documents, annotating (0%, 50%, 100%) with rationales gives (88.5%, 91.7%, 92.2%).

To make this point more broadly, we find that the $R = 200$ curve is significantly above the $R = 0$ curve ($p < 0.05$) at all $T \leq 1200$. By contrast, the $R = 800, R = 1000, \ldots R = 1600$ points at each T

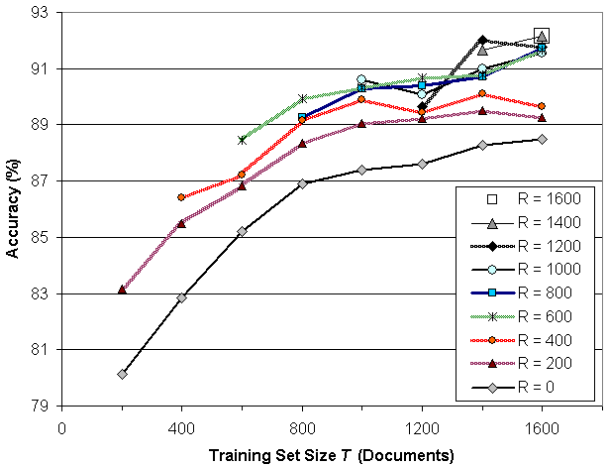

Figure 3: Classification accuracy for $T \in \{200, 400, ..., 1600\}$ training documents (x-axis) when only $R \in \{0, 200, ..., T\}$ of them are annotated with rationales (different curves). The $R = 0$ curve above corresponds to the baseline S2 from Figure 2. S1's points are found above as the leftmost points on the other curves, where $R = T$.

value are all-pairs statistically indistinguishable.

The figure also suggests that rationales and documents may be somewhat orthogonal in their benefit. When one has many documents and few rationales, there is no longer much benefit in adding more documents (the curve is flattening out), but adding more rationales seems to provide a fresh benefit: rationales have not yet reached *their* point of diminishing returns. (While this fresh benefit was often statistically significant, and greater than the benefit from more documents, our experiments did not establish that it was significantly greater.)

The above experiments keep *all* of A0's rationales on *a fraction of* training documents. We also experimented with keeping *a fraction of* A0's rationales (chosen randomly with randomized rounding) on *all* training documents. This yielded no noteworthy or statistically significant differences from Figure 3.

These latter experiments simulate a "lazy annotator" who is less assiduous than A0. Such annotators may be common in the real world. We also suspect that they will be more desirable. First, they should be able to add more rationales per hour than the A0-style annotator from Figure 3: some rationales are simply more noticeable than others, and a lazy annotator will quickly find the most noticeable ones without wasting time tracking down the rest. Second, the "most noticeable" rationales that they mark may be the most effective ones for learning, although our random simulation of laziness could not test that.

7 Related Work

Our rationales resemble "side information" in machine learning—supplementary information about the target function that is available at training time. Side information is sometimes encoded as "virtual examples" like our contrast examples or pseudoexamples. However, past work generates these by *automatically* transforming the training examples in ways that are expected to preserve or alter the classification (Abu-Mostafa, 1995). In another formulation, virtual examples are automatically generated but must be manually annotated (Kuusela and Ocone, 2004). Our approach differs because a human helps to generate the virtual examples. Enforcing a margin between ordinary examples and contrast examples also appears new.

Other researchers have considered how to reduce annotation effort. In active learning, the annotator classifies only documents where the system so far is less confident (Lewis and Gale, 1994), or in an information extraction setting, incrementally corrects details of the system's less confident entity segmentations and labelings (Culotta and McCallum, 2005).

Raghavan et al. (2005) asked annotators to identify globally "relevant" *features*. In contrast, our approach does not force the annotator to evaluate the importance of features individually, nor in a global context outside any specific document, nor even to know the learner's feature space. Annotators only mark text that supports their classification decision. Our methods then consider the combined effect of this text on the feature vector, which may include complex features not known to the annotator.

8 Future Work: Generative models

Our SVM contrast method (section 3) is not the only possible way to use rationales. We would like to explicitly *model* rationale annotation as a noisy process that reflects, imperfectly and incompletely, the annotator's internal decision procedure.

A natural approach would start with log-linear models in place of SVMs. We can define a probabilistic classifier

$$p_\theta(y \mid x) \stackrel{\text{def}}{=} \frac{1}{Z(x)} \exp \sum_{h=1}^{k} \theta_h f_h(x, y) \qquad (9)$$

where $\vec{f}(\cdot)$ extracts a feature vector from a classified document.

A standard training method would be to choose θ to maximize the conditional likelihood of the training classifications:

$$\underset{\vec{\theta}}{\operatorname{argmax}} \prod_{i=1}^{n} p_\theta(y_i \mid x_i) \quad (10)$$

When a rationale r_i is also available for each (x_i, y_i), we propose to maximize a likelihood that tries to predict these rationale data *as well*:

$$\underset{\vec{\theta}}{\operatorname{argmax}} \prod_{i=1}^{n} p_\theta(y_i \mid x_i) \cdot p_{\theta'}(r_i \mid x_i, y_i, \theta) \quad (11)$$

Notice that a given guess of θ might make equation (10) large, yet accord badly with the annotator's rationales. In that case, the second term of equation (11) will exert pressure on θ to change to something that conforms more closely to the rationales. If the annotator is correct, such a θ will generalize better beyond the training data.

In equation (11), $p_{\theta'}$ models the stochastic process of rationale annotation. What is an annotator actually doing when she annotates rationales? In particular, how do her rationales derive from the true value of θ and thereby tell us about θ? Building a good model $p_{\theta'}$ of rationale annotation will require some exploratory data analysis. Roughly, we expect that if $\theta_h f_h(x_i, y)$ is much higher for $y = y_i$ than for other values of y, then the annotator's r_i is correspondingly more likely to indicate in some way that feature f_h strongly influenced annotation y_i. However, we must also model the annotator's limited patience (she may not annotate all important features), sloppiness (she may indicate only indirectly that f_h is important), and bias (tendency to annotate some *kinds* of features at the expense of others).

One advantage of this generative approach is that it eliminates the need for contrast examples. Consider a non-textual example in which an annotator highlights the line crossing in a digital image of the digit "8" to mark the rationale that distinguishes it from "0." In this case it is not clear how to mask out that highlighted rationale to create a contrast example in which relevant features would not fire.[11]

[11] One cannot simply flip those highlighted pixels to white or black, since that would cause new features to fire. Possibly one could simply suppress any feature that depends in any way on the highlighted pixels, but this would take away too many important features, including global features.

9 Conclusions

We have proposed a quite simple approach to improving machine learning by exploiting the cleverness of annotators, asking them to provide enriched annotations for training. We developed and tested a particular discriminative method that can use "annotator rationales"—even on a fraction of the training set—to significantly improve sentiment classification of movie reviews.

We found fairly good annotator agreement on the rationales themselves. Most annotators provided several rationales per classification without taking too much extra time, even in our text classification scenario, where the rationales greatly outweigh the classifications in number and complexity. Greater speed might be possible through an improved user interface or passive feedback (e.g., eye tracking).

In principle, many machine learning methods might be modified to exploit rationale data. While our experiments in this paper used a discriminative SVM, we plan to explore generative approaches.

References

Y. S. Abu-Mostafa. 1995. Hints. *Neural Computation*, 7:639–671, July.

P. Beineke, T. Hastie, and S. Vaithyanathan. 2004. The sentimental factor: Improving review classification via human-provided information. In *Proc. of ACL*, pages 263–270.

A. L. Blum and R. L. Rivest. 1992. Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1):117–127.

A. Culotta and A. McCallum. 2005. Reducing labeling effort for structured prediction tasks. In *AAAI*, pages 746–751.

S. Dumais. 1998. Using SVMs for text categorization. *IEEE Intelligent Systems Magazine*, 13(4), July/August.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proc. of the European Conf. on Machine Learning*, pages 137–142.

P. Kuusela and D. Ocone. 2004. Learning with side information: PAC learning bounds. *J. of Computer and System Sciences*, 68(3):521–545, May.

D. D. Lewis and W. A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proc. of ACM-SIGIR*, pages 3–12.

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.

H. Raghavan, O. Madani, and R. Jones. 2005. Interactive feature selection. In *Proc. of IJCAI*, pages 41–46.