Sentiment Analysis Part 2

Jay Kaiser & Mike Czerniakowski

Dependency Tree-based Sentiment Classification using CRFs with Hidden Variables

Nakagawa et al. (2010)

Sentiment Classification

Similar to topic classification

Subjective sentences can be classified as *positive*, *negative*, or *neutral*.

Typically, this can be approached with supervised learning with a bag-of-words as a feature set.

In this manner, word order and head-modifier relations are ignored.

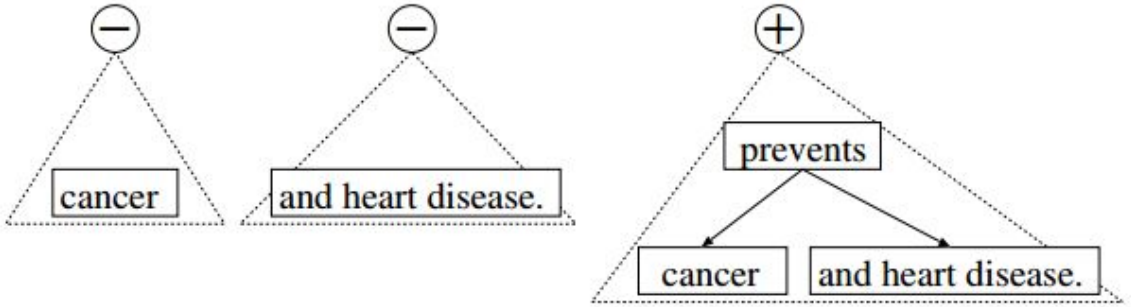
Sentiment Classification

However, unlike in topic classification, sentiment polarities can be reversed.

“It prevents cancer and heart disease.”

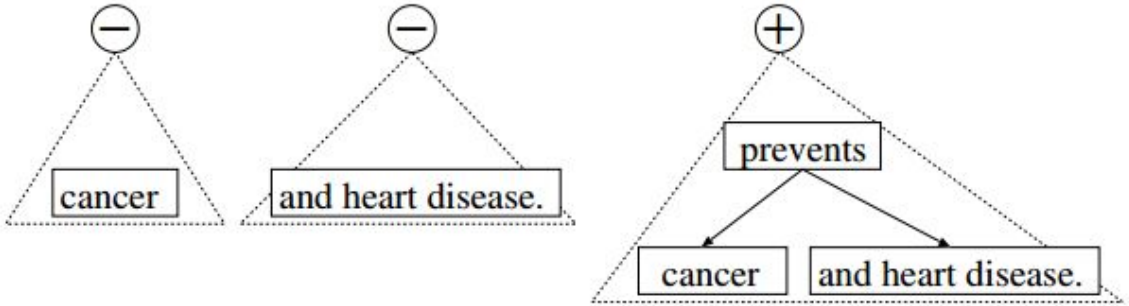
Cancer and *heart disease* have negative polarities.

Prevents reverses the polarity, so the full sentence is positive.



Sentiment Classification

Therefore, considering interactions between words instead of handling words independently is important.



Goal

Create a dependency tree-based method for Japanese and English sentiment classification using conditional random fields (CRFs) with hidden variables.

A Probabilistic Model based on Dependency Trees

Consider the following dependency tree.

Subtrees are the individual phrases of the tree whose root node is one of the phrases of the sentence.

Two phrases are dependent if they have a head-modifier relationship in the dependency tree.

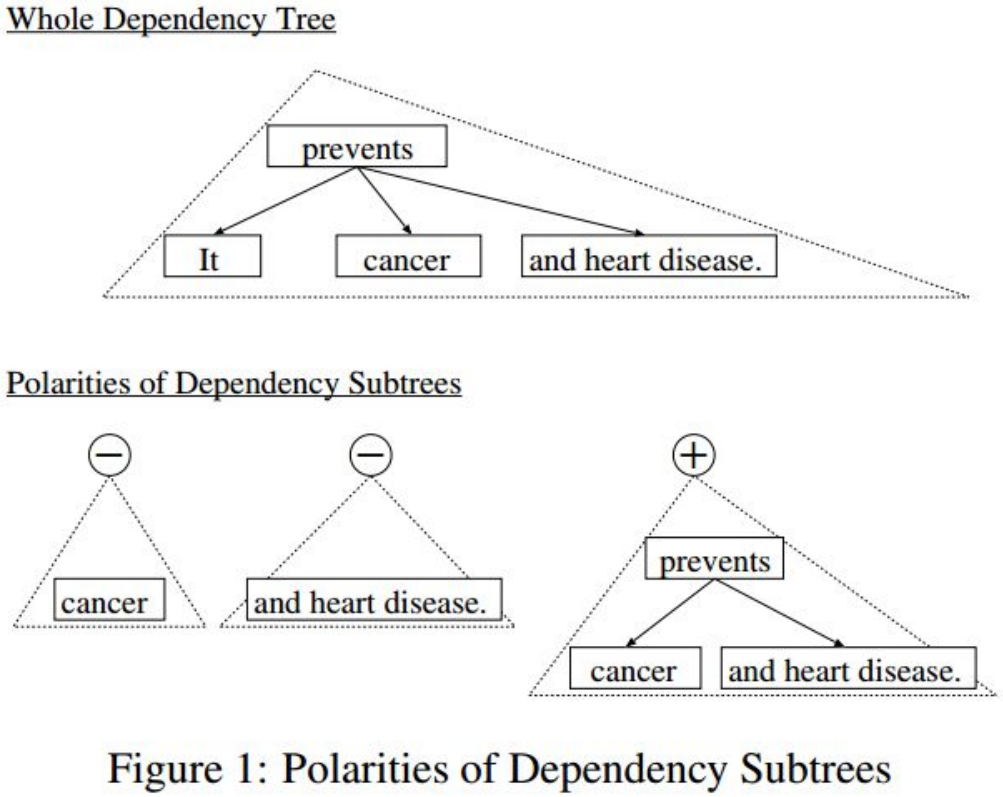


Figure 1: Polarities of Dependency Subtrees

A Probabilistic Model based on Dependency Trees

Each subtree's polarity is represented by a "hidden" variable, here represented with a circle.

The variables are "hidden" because they are not outputted in the final result; only the polarity of the entire sentence is.

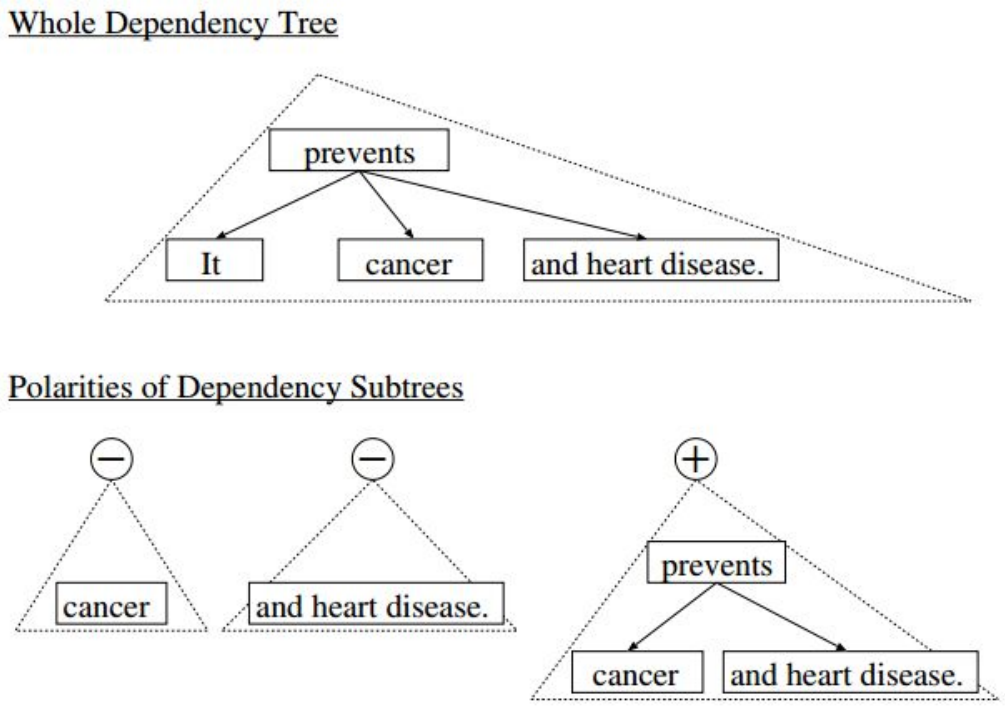


Figure 1: Polarities of Dependency Subtrees

Some Variables

n denotes the number of phrases in the sentence

w_i denotes the *i*-th phrase

h_i denotes the head-index of the *i*-th phrase

s_i denotes the polarity of the dependency subtree whose root is the *i*-th phrase

p denotes the polarity of the whole sentence (represented at w_0)

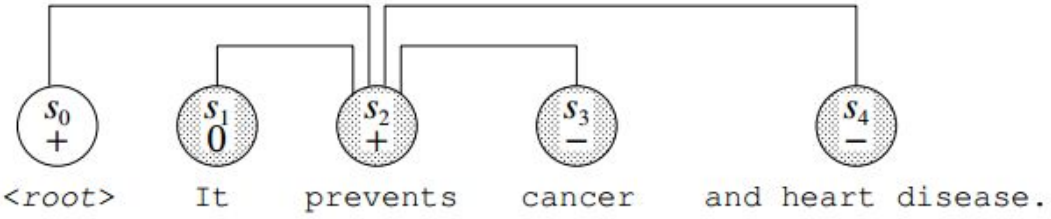


Figure 2: Probabilistic Model based on Dependency Tree

Joint Probability Distribution...

...of the sentiment polarities of dependency subtrees **s**,

given a subjective sentence **w** and its dependency tree **h**,

using log-linear models:

where **Λ** = {λ_1, ... , λ_K} is the set of parameters of the model.

$$P_\Lambda(\mathbf{s}|\mathbf{w},\mathbf{h}) = \frac{1}{Z_\Lambda(\mathbf{w},\mathbf{h})} \exp\left\{\sum_{k=1}^{K} \lambda_k F_k(\mathbf{w},\mathbf{h},\mathbf{s})\right\}, \quad (1)$$

$$Z_\Lambda(\mathbf{w},\mathbf{h}) = \sum_{\mathbf{s}} \exp\left\{\sum_{k=1}^{K} \lambda_k F_k(\mathbf{w},\mathbf{h},\mathbf{s})\right\}, \quad (2)$$

$$F_k(\mathbf{w},\mathbf{h},\mathbf{s}) = \sum_{i=1}^{n} f_k(i,\mathbf{w},\mathbf{h},\mathbf{s}), \quad (3)$$

Joint Probability Distribution...

$f_k(i, \mathbf{w}, \mathbf{h}, \mathbf{s})$, is the feature function of the ***i*-th** phrase, classified to

node feature which considers only the corresponding node, or

edge feature which considers both the corresponding node and its head

$$f_k(i, \mathbf{w}, \mathbf{h}, \mathbf{s}) = \begin{cases} f_k^{\mathrm{n}}(w_i, s_i) & (k \in \mathbf{K}^{\mathrm{n}}), \\ f_k^{\mathrm{e}}(w_i, s_i, w_{h_i}, s_{h_i}) & (k \in \mathbf{K}^{\mathrm{e}}), \end{cases} \quad (4)$$

where $\mathbf{K}^{\mathrm{n}}$ and $\mathbf{K}^{\mathrm{e}}$ respectively represent the sets of indices of node features and edge features.

Joint Probability Distribution...

The polarity *p* of the root node (s_0) is regarded as the polarity of the whole sentence.

a.k.a. the polarity of the sentence is obtained as the marginal probability of the root node polarity, by summing the probabilities for all the possible configurations of hidden variables.

$$p = \underset{p'}{\operatorname{argmax}}\, P_\Lambda(p'|\mathbf{w}, \mathbf{h}), \tag{5}$$

$$P_\Lambda(p|\mathbf{w}, \mathbf{h}) = \sum_{\mathbf{s}:s_0=p} P_\Lambda(\mathbf{s}|\mathbf{w}, \mathbf{h}). \tag{6}$$

Belief Propagation

Because enumerating all the possible configurations of hidden variables is computationally hard, *sum-product belief propagation* is used for the calculation.

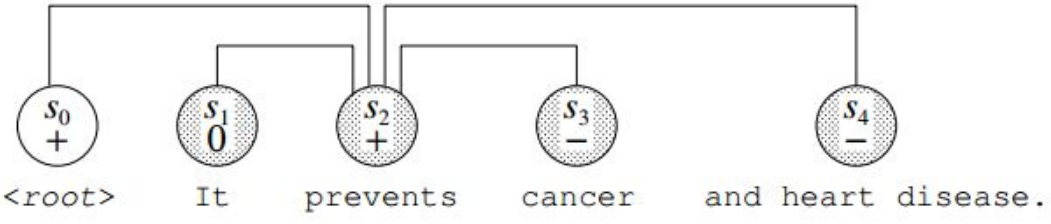


Figure 2: Probabilistic Model based on Dependency Tree

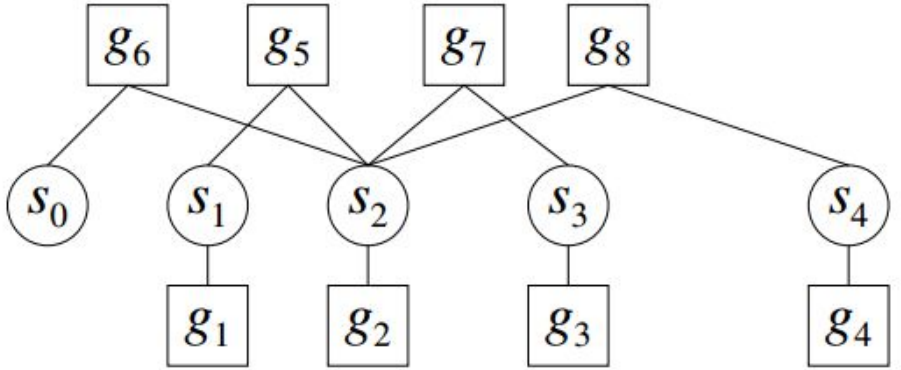


Figure 3: Factor Graph

Belief Propagation

Circles : variable nodes

Squares : factor (feature) nodes

g_1 - g_4 : node features

g_5 - g_8 : edge features

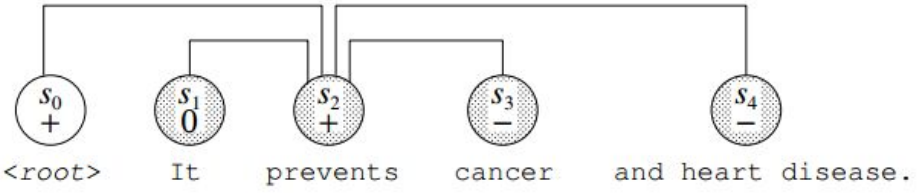


Figure 2: Probabilistic Model based on Dependency Tree

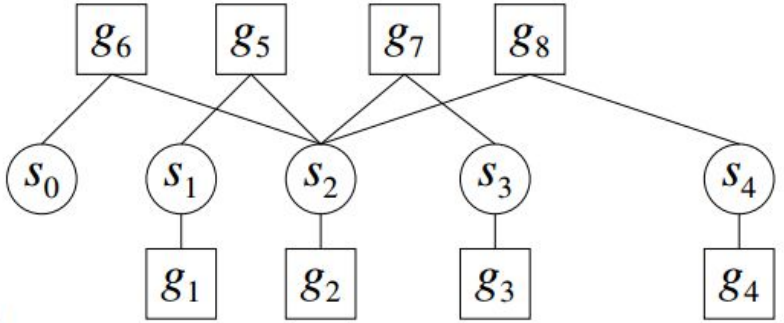


Figure 3: Factor Graph

$$f_k(i, \mathbf{w}, \mathbf{h}, \mathbf{s}) = \begin{cases} f_k^{\mathrm{n}}(w_i, s_i) & (k \in \mathbf{K}^{\mathrm{n}}), \\ f_k^{\mathrm{e}}(w_i, s_i, w_{h_i}, s_{h_i}) & (k \in \mathbf{K}^{\mathrm{e}}), \end{cases} \quad (4)$$

Belief Propagation

Marginal distribution is calculated by passing messages (beliefs) among the variables and factors connected by edges in the factor graph.

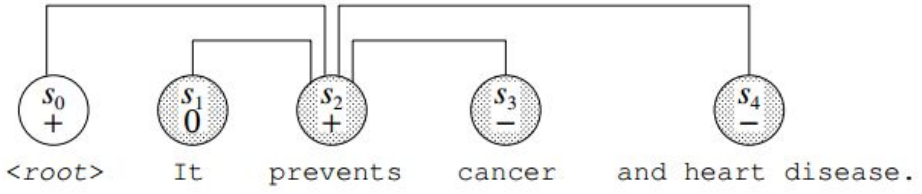


Figure 2: Probabilistic Model based on Dependency Tree

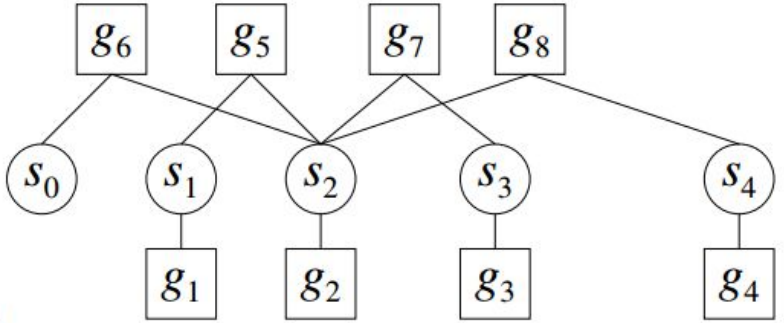


Figure 3: Factor Graph

$$f_k(i, \mathbf{w}, \mathbf{h}, \mathbf{s}) = \begin{cases} f_k^{\mathrm{n}}(w_i, s_i) & (k \in \mathbf{K}^{\mathrm{n}}), \\ f_k^{\mathrm{e}}(w_i, s_i, w_{h_i}, s_{h_i}) & (k \in \mathbf{K}^{\mathrm{e}}), \end{cases} \quad (4)$$

Parameter Estimation

To estimate model parameters **Λ**, given *L* training examples $D = \{\langle \mathbf{w}^l, \mathbf{h}^l, p^l \rangle\}_{l=1}^{L}$, *maximum a posteriori estimation with Gaussian priors* is used,

where **σ** is a parameter of Gaussian priors and is set to 1.0 in later experiments.

$$\mathcal{L}_\Lambda = \sum_{l=1}^{L} \log P_\Lambda(p^l | \mathbf{w}^l, \mathbf{h}^l) - \frac{1}{2\sigma^2} \sum_{k=1}^{K} \lambda_k^2, \quad (7)$$

$$\hat{\Lambda} = \underset{\Lambda}{\operatorname{argmax}}\, \mathcal{L}_\Lambda, \quad (8)$$

Parameter Estimation

The partial derivatives of $\mathcal{L}_\Lambda$ are as follows:

$$\frac{\partial \mathcal{L}_\Lambda}{\partial \lambda_k} = \sum_{l=1}^{L} \left[\sum_{\mathbf{s}} P_\Lambda(\mathbf{s}|\mathbf{w}^l, \mathbf{h}^l, p^l) F_k(\mathbf{w}^l, \mathbf{h}^l, \mathbf{s}) - \sum_{\mathbf{s}} P_\Lambda(\mathbf{s}|\mathbf{w}^l, \mathbf{h}^l) F_k(\mathbf{w}^l, \mathbf{h}^l, \mathbf{s}) \right] - \frac{1}{\sigma^2} \lambda_k. \tag{9}$$

Parameter Estimation

The model parameters are calculated with the *L-BFGS quasi-Newton method* using the objective function and its partial derivatives.

Features

s_i denotes the hidden variable which represents the polarity of the dependency subtree whose root node is the *i*-th phrase {+1, -1}

q_i denotes the *prior polarity* of the *i*-th phrase {+1, 0, -1}

r_i denotes the *polarity reversal* of the *i*-th phrase {0, 1}

m_i denotes the number of words in the *i*-th phrase

$u_{i,k'}$ $b_{i,k'}$ $c_{i,k'}$ $f_{i,k}$ respectively denote the surface form, base form, coarse-grained POS tag, and fine-grained POS tag of the *k*-th word in the *i*-th phrase

Node Features	
a	s_i
b	$s_i \& q_i$
c	$s_i \& q_i \& r_i$
d	$s_i \& u_{i,1}, \cdots, s_i \& u_{i,m_i}$
e	$s_i \& c_{i,1}, \cdots, s_i \& c_{i,m_i}$
f	$s_i \& f_{i,1}, \cdots, s_i \& f_{i,m_i}$
g	$s_i \& u_{i,1} \& u_{i,2}, \cdots, s_i \& u_{i,m_i-1} \& u_{i,m_i}$
h	$s_i \& b_{i,1} \& b_{i,2}, \cdots, s_i \& b_{i,m_i-1} \& b_{i,m_i}$
Edge Features	
A	$s_i \& s_j$
B	$s_i \& s_j \& r_j$
C	$s_i \& s_j \& r_j \& q_j$
D	$s_i \& s_j \& b_{i,1}, \cdots, s_i \& s_j \& b_{i,m_i}$
E	$s_i \& s_j \& b_{j,1}, \cdots, s_i \& s_j \& b_{j,m_j}$

Table 1: Features Used in This Study

Processing the Data

For the Japanese data, the JU-MAN morphological analysis system and the KNP[2] dependency parser were used to construct dependency trees.

For the English data, the MX-POST POS tagger and the MaltParser[3] were used to construct word-based dependency trees that were then converted to phrase-based ones (see Appendix A).

Acquiring the Polarities

Sentiment polarity dictionaries made by Kobayashi et al. (2007) and Higashiyama et al. (2008) were used for Japanese experiments (resulting in dictionaries of 6,974 positive expressions and 8,428 negative expressions).

A sentiment polarity dictionary made by Wilson et al. (2005) was used for English experiments (resulting in 2,289 positive expressions and 4,143 negative expressions).

Acquiring the Polarities

When a phrase contains the words registered in the dictionaries, its *prior polarity* is set to the registered polarity, otherwise the prior polarity is set to 0.

When a phrase contains multiple words in the dictionaries, the registered polarity of the last word is used.

Polarity Reversal - Content Words

Polarity reversal word dictionaries were prepared, and the *polarity reversal* $\boldsymbol{r}_i$ of a phrase was set to **1** if the phrase contained a word in the dictionaries, and set to **0** otherwise.

A Japanese polarity reverse sentiment dictionary was constructed from an automatically constructed corpus (see Appendix B), resulting in 219 polarity reversing words.

An English polarity reverse sentiment dictionary was constructed from the General Inquirer Dictionary (see Choi and Cardie (2008)), resulting in 121 polarity reversing words.

Polarity Reversal - Function Words

The scope of a function-word negator is generally limited to the phrase containing it in Japanese.

Therefore, the prior polarity q_i and the polarity reversal r_i are changed to the following if the phrase contains a function-word negator (in Japanese), or if the phrase is modified by a function-word negator (in English).

$$q_i' = -q_i, \quad (10)$$

$$r_i' = 1 - r_i. \quad (11)$$

Acquiring the Data

Four corpora were used for the Japanese sentiment classification:

- the Automatically Constructed Polarity-tagged corpus (ACP)
- the Kyoto University and NTT Blog corpus (KNB)
- the NTCIR Japanese opinion corpus (NTC-J)
- the 50 Topics Evaluative Information Corpus (50 Topics)

Four more were used for the English sentiment classification:

- the Customer Review data (CR)
- the MPQA Opinion corpus (MPQA)
- the Movie Review Data (MR)
- the NTCIR English opinion corpus (NTC-E)

Acquiring the Data

Each corpus was randomly split into 10 portions, and a 10-fold cross validation was conducted on each.

Language	Corpus	Number of Instances	(Positive / Negative)
Japanese	ACP	6,510	(2,738 / 3,772)
	KNB	2,288	(1,423 / 865)
	NTC-J	3,485	(1,083 / 2,402)
	50 Topics	5,366	(3,175 / 2,191)
English	CR	3,772	(2,406 / 1,366)
	MPQA	10,624	(3,316 / 7,308)
	MR	10,662	(5,331 / 5,331)
	NTC-E	3,812	(1,226 / 2,586)

Table 2: Statistical Information of Corpora

Baselines

Their method was compared to six baseline models.

p_0 denotes the major polarity in training data

$\mathbf{H}_i$ denotes the set consisting of all the ancestor nodes of the *i*-th phrase in the dependency tree

$$\operatorname{sgn}(x)=\begin{cases}+1 & (x>0),\\ 0 & (x=0),\\ -1 & (x<0).\end{cases}$$

Baseline 1: Voting without Polarity Reversal

The polarity of a subjective sentence is decided by voting of each phrase's prior polarity.

$$p = \mathrm{sgn}\left(\sum_{i=1}^{n} q_i + 0.5 p_0\right). \qquad (12)$$

Baseline 2: Voting with Polarity Reversal

Same as *Voting without Polarity Reversal*, but polarities of phrases which have **odd** numbers of reversal phrases in their ancestors are reversed before voting.

$$p=\mathrm{sgn}\left(\sum_{i=1}^{n} q_i \prod_{j\in\mathbf{H}_i}(-1)^{r_j} + 0.5p_0\right). \quad (13)$$

Baseline 3: Rule

Polarity is based on rules by considering the sentiment polarities of dependency subtrees.

The polarity of the dependency subtree whose root is the *i*-th phrase is decided by voting the prior polarity of the *i*-th phrase and the polarities of the dependency subtrees whose root nodes are the modifiers of the *i*-th phrase.

The polarities are reversed if their head phrase has a reversal word.

$$s_i = \mathrm{sgn}\left(q_i + \sum_{j:h_j=i} s_j(-1)^{r_i}\right), \quad (14)$$

$$p = \mathrm{sgn}(s_0 + 0.5p_0). \quad (15)$$

Baseline 4: Bag-of-Features with No Dictionaries

Polarity is classified using Support Vector Machines.

Surface forms, base forms, coarse-grained POS tags and fine-grained POS tags of word unigrams and bigrams in the sentence used as features.

No polarity dictionary is used.

Baseline 5: Bag-of-Features without Polarity Reversal

Same as *Bag-of-Features with No Dictionaries*, but the voting result of prior polarities is also used as a feature.

Baseline 6: Bag-of-Features with Polarity Reversal

Same as *Bag-of-Features without Polarity Reversal*, except the polarities of phrases which have odd numbers of reversal phrases in their ancestors are reversed before voting.

Results

Method	Japanese				English			
	ACP	KNB	NTC-J	50 Topics	CR	MPQA	MR	NTC-E
Voting-w/o Rev.	0.686	0.764	0.665	0.727	0.714	0.804	0.629	0.730
Voting-w/ Rev.	0.732	0.792	0.714	0.765	0.742	0.817	0.631	0.740
Rule	0.734	0.792	0.742	0.764	0.743	0.818	0.629	0.750
BoF-no Dic.	0.798	0.758	0.754	0.761	0.793	0.818	0.757	0.768
BoF-w/o Rev.	0.812	0.823	0.794	0.805	0.802	0.840	0.761	0.793
BoF-w/ Rev.	0.822	0.830	0.804	0.819	**0.814**	0.841	0.764	0.797
Tree-CRF	**0.846***	**0.847***	**0.826***	**0.841***	**0.814**	**0.861***	**0.773***	**0.804**

(* indicates statistical significance at $p < 0.05$)

Table 3: Accuracy of Sentiment Classification

Conclusion

Method	Japanese				English			
	ACP	KNB	NTC-J	50 Topics	CR	MPQA	MR	NTC-E
Voting-w/o Rev.	0.686	0.764	0.665	0.727	0.714	0.804	0.629	0.730
Voting-w/ Rev.	0.732	0.792	0.714	0.765	0.742	0.817	0.631	0.740
Rule	0.734	0.792	0.742	0.764	0.743	0.818	0.629	0.750
BoF-no Dic.	0.798	0.758	0.754	0.761	0.793	0.818	0.757	0.768
BoF-w/o Rev.	0.812	0.823	0.794	0.805	0.802	0.840	0.761	0.793
BoF-w/ Rev.	0.822	0.830	0.804	0.819	**0.814**	0.841	0.764	0.797
Tree-CRF	**0.846***	**0.847***	**0.826***	**0.841***	**0.814**	**0.861***	**0.773***	**0.804**

(* indicates statistical significance at $p < 0.05$)

Table 3: Accuracy of Sentiment Classification

The model performed better for the Japanese corpora than for the English corpora, and it outperformed the other tests by a statistically significant margin in 6 out 8 cases.

Methods with Polarity Reversal performed better than those without it.

Tree-CRF handled dictionary errors better than BoF-w/ Rev. and correctly modeled words that were not in the dictionary like *protect* and *withdraw*.

Learning Word Vectors for Sentiment Analysis

Maas et al. (2011)

Word Representations

Word representations are important in many natural language processing systems.

Although it is common to represent words as indices in a vocabulary, this fails to capture the rich relational structure of the lexicon.

Vector models encode continuous similarities between words as distance or angle between word vectors in a high-dimensional space, making them much better in this regard.

Therefore,

The paper presents a model to capture both semantic and sentiment similarities among words.

The semantic component of the model learns word vectors via an unsupervised probabilistic model of documents.

However,

This version of the model misses crucial sentiment information.

“For example, it learns that *wonderful* and *amazing* are semantically close, but it doesn’t capture that these are both very strong sentiment words, at the opposite end of the spectrum from *terrible* and *awful*.”

Therefore, the model is extended with a supervised sentiment component as well.

Thus, the vector representations of words are used to predict the sentiments of words in the contexts in which they appear, causing words with similar sentiments to have similar vectors.

Capturing Semantic Similarities

A probabilistic model of a document is built, using a continuous mixture distribution over words indexed by a multi-dimensional random variable θ.

Words in a document are assumed to be conditionally independent given the mixture variable θ.

Capturing Semantic Similarities

The probability of a document *d* is calculated using a joint distribution over the document and θ:

$$p(d) = \int p(d, \theta) d\theta = \int p(\theta) \prod_{i=1}^{N} p(w_i|\theta) d\theta. \quad (1)$$

where *N* is the number of words in *d*, and w_i is the i^{th} word in *d*.

Capturing Semantic Similarities

$p(w_i|\theta)$ is the conditional distribution using a log-linear model with parameters:

$R \in \mathbb{R}^{(\beta x |V|)}$ where each word w in the vocabulary V has a β-dimension vector representation $\phi_w = Rw$ corresponding to that word's column in R.

b_w is the bias for each word to capture differences in overall word frequencies.

$\theta \in \mathbb{R}^{\beta}$ is a β-dimensional vector random variable which weighs each of the β dimensions of words' representation vectors.

The energy function E assigns energy to a word w given these model parameters:

$$E(w; \theta, \phi_w, b_w) = -\theta^T \phi_w - b_w. \qquad (2)$$

Capturing Semantic Similarities

Altogether, the conditional distribution model is the following:

$$p(w|\theta; R, b) = \frac{\exp(-E(w; \theta, \phi_w, b_w))}{\sum_{w' \in V} \exp(-E(w'; \theta, \phi_{w'}, b_{w'}))} \tag{3}$$

$$= \frac{\exp(\theta^T \phi_w + b_w)}{\sum_{w' \in V} \exp(\theta^T \phi_{w'} + b_{w'})}. \tag{4}$$

Capturing Semantic Similarities

The maximum likelihood learning for this model when given a set of unlabeled documents *D* is the following:

$$\max_{R,b} p(D; R, b) = \prod_{d_k \in D} \int p(\theta) \prod_{i=1}^{N_k} p(w_i|\theta; R, b) d\theta. \tag{5}$$

Using *maximum a posteriori estimates* for θ, this can be approximated as:

$$\max_{R,b} \prod_{d_k \in D} p(\hat{\theta}_k) \prod_{i=1}^{N_k} p(w_i|\hat{\theta}_k; R, b), \tag{6}$$

where $\hat{\theta}_k$ denotes the *MAP* estimate of θ for d_k.

Capturing Semantic Similarities

Frobenius norm regularization for the word representation matrix *R*, maximized with respect to *R* and *b*:

$$\nu||R||_F^2 + \sum_{d_k \in D} \lambda||\hat{\theta}_k||_2^2 + \sum_{i=1}^{N_k} \log p(w_i|\hat{\theta}_k; R, b), \tag{7}$$

Capturing Word Sentiment

As of yet, nothing in the model captures sentiment information.

It will merely produce representations where words that occur together in documents have similar representations.

Therefore, labeled documents are utilized as well to improve the model's word representations.

Capturing Word Sentiment

The word vectors of the model should predict the sentiment label using an appropriate predictor:

$$\hat{s} = f(\phi_w). \qquad (8)$$

which maps a word vector ϕ_w to a predicted sentiment label $\hat{s}$.

Capturing Word Sentiment

For simplicity, they consider the case where the sentiment label ***s*** is a scalar continuous value representing the polarity of a document.

This reflects the case of online reviews where documents are given a label on a star rating scale.

Values are mapped to the interval $s \in [0, 1]$ and treated as a probability of positive sentiment polarity.

Capturing Word Sentiment

A logistic regression is employed as the predictor *f(x)*.

w's vector representation ϕ_w and regression weights Ψ are used to express this as the following:

$$p(s = 1|w; R, \psi) = \sigma(\psi^T \phi_w + b_c), \qquad (9)_1$$

where *σ(x)* is the logistic function, and $\Psi \in \mathbb{R}^\beta$ is the logistic regression weight vector.

Scalar bias b_c is also introduced for the classifier.

Capturing Word Sentiment

“The logistic regression weights Ψ and b_c define a linear hyperplane in the word vector space where a word vector’s positive sentiment probability depends on where it lies with respect to this hyperplane.

Learning over a collection of documents results in words residing different distances from this hyperplane based on the average polarity of documents in which the words occur.”

Capturing Word Sentiment

To maximize the probability of document labels given documents, where s_k is the sentiment label for document d_k, the following is obtained:

$$\max_{R,\psi,b_c} \sum_{k=1}^{|D|} \sum_{i=1}^{N_k} \log p(s_k|w_i; R, \psi, b_c). \qquad (10)$$

Learning

The final objective function:

$$\nu||R||_F^2 + \sum_{k=1}^{|D|} \lambda||\hat{\theta}_k||_2^2 + \sum_{i=1}^{N_k} \log p(w_i|\hat{\theta}_k; R, b)$$
$$+ \sum_{k=1}^{|D|} \frac{1}{|S_k|} \sum_{i=1}^{N_k} \log p(s_k|w_i; R, \psi, b_c). \qquad (11)$$

where $|S_k|$ denotes the number of documents in the dataset with the same rounded value of s_k.

Learning

The weighting $1/(|S_k|)$ is introduced to combat the imbalance in ratings present in review collections.

“This weighting prevents the overall distribution of document ratings from affective the estimate of document ratings in which a particular word occurs.”

Word Representation Learning

25,000 movie reviews from IMDB were induced, limited at most 30 reviews from any one movie.

A fixed dictionary of the 5,000 most frequent tokens was built, but the most frequent 50 were ignored.

Stop words were not removed, because certain stop words (like negating words) are indicative of sentiment.

Word Representation Learning

Stemming was not used, because the model already learns similar representations for words of the same stem when the data suggests it.

Certain non-word tokens that were indicative of sentiment, like *!* and *:-)*, were allowed in the vocabulary as well.

IMDB movie ratings from stars 1-10 were mapped linearly to [0,1] to use as document labels when training the models.

Word Representation Learning

The semantic component of the model does not require labelled documents.

Therefore, a variant of the model was trained which uses 50,000 unlabeled reviews in addition to the labeled set of 25,000 reviews.

The unlabeled set contains neutral reviews as well as the polarized ones found in the labeled set.

When training the model with additional unlabeled data, a common scenario is captured where the amount of labeled data is small relative to the amount of unlabeled data.

For all word vector models, 50-dimensional vectors are used.

Results

Accessing similarity of words with all other words *w'* using cosine similarity metrics, the words deemed most similar by the model were found.

Words indicative of sentiment tend to have high similarity with words of the same sentiment polarity, so even the purely unsupervised model's results prove promising.

	Our model Sentiment + Semantic	Our model Semantic only	LSA
melancholy	bittersweet	thoughtful	poetic
	heartbreaking	warmth	lyrical
	happiness	layer	poetry
	tenderness	gentle	profound
	compassionate	loneliness	vivid
ghastly	embarrassingly	predators	hideous
	trite	hideous	inept
	laughably	tube	severely
	atrocious	baffled	grotesque
	appalling	smack	unsuspecting
lackluster	lame	passable	uninspired
	laughable	unconvincing	flat
	unimaginative	amateurish	bland
	uninspired	clichéd	forgettable
	awful	insipid	mediocre
romantic	romance	romance	romance
	love	charming	screwball
	sweet	delightful	grant
	beautiful	sweet	comedies
	relationship	chemistry	comedy

Comparison 1: Latent Semantic Analysis (LSA)

A truncated SVD was applied to a tf.idf* weighted, cosine normalized count matrix (a standard weighting and smoothing scheme for VSM induction).

- tf.idf = term frequency-inverse document frequency

(reflects how important a word is to a document in a corpus)

Comparison 2: Latent Dirichlet Allocation (LDA)

A probabilistic document model that assumes each document is a mixture of latent topics was used for inducing word representations from the topic matrix, unoptimized.

See *Section 2: Related Work* for more information.

Comparison 3: Weighting Variants

Both binary term frequency weighting with smoothed delta idf and no idf were evaluated, using cosine normalization as well.

Document Polarity Classification

The classifier predicts whether a given review is positive or negative, given a document's bag of words vector *v*.

The features from the model were obtained using a matrix-vector product *Rv*, where *v* can have arbitrary tf.idf weighting.

Cosine normalization is applied to the final feature vector *Rv*.

This procedure was also used to obtain features from the LDA and LSA word vectors.

Pang and Lee Movie Review Database

Using Pang and Lee (2004)’s dataset of 2,000 movie reviews, where each review is associated with a binary sentiment polarity label,

10-fold cross validation was used.

A linear support vector machine classifier trained with LIBLINEAR was used, and the SVM regularization parameter was set to the same value used by Pang and Lee (2004).

Pang and Lee Movie Review Database Results

The model outperforms those of other VSMs, and it performs best when combined with the original bag of words representation.

The variant of the model trained with additional unlabeled data performed best, suggesting the model can effectively utilize large amounts of unlabeled data along with labeled examples.

Despite only using a vocabulary of only 5,000 words, the model still performs competitively.

Features	PL04	Our Dataset	Subjectivity
Bag of Words (bnc)	85.45	87.80	87.77
Bag of Words (bΔt'c)	85.80	88.23	85.65
LDA	66.70	67.42	66.65
LSA	84.55	83.96	82.82
Our Semantic Only	87.10	87.30	86.65
Our Full	84.65	87.44	86.19
Our Full, Additional Unlabeled	87.05	87.99	87.22
Our Semantic + Bag of Words (bnc)	88.30	88.28	88.58
Our Full + Bag of Words (bnc)	87.85	88.33	88.45
Our Full, Add'l Unlabeled + Bag of Words (bnc)	88.90	88.89	88.13
Bag of Words SVM (Pang and Lee, 2004)	87.15	N/A	90.00
Contextual Valence Shifters (Kennedy and Inkpen, 2006)	86.20	N/A	N/A
tf.Δidf Weighting (Martineau and Finin, 2009)	88.10	N/A	N/A
Appraisal Taxonomy (Whitelaw et al., 2005)	90.20	N/A	N/A

Pang and Lee Movie Review Database Results

To avoid the chance that the classifier learns set phrases regarding movie names or plot elements from reviews and uses those to classify two movies together under the same results, a substantially larger dataset using disjoint sets of movies for training and testing is introduced.

IMDB Review Dataset

50,000 reviews were taken from IMDB, allowing no more than 30 reviews per movie.

There were an even number of positive and negative reviews, so randomly guessing yields 50% accuracy.

Only highly polarized reviews were considered, where a score ≤ 4 are negative, and scores ≥ 7 are positive.

IMDB Review Dataset

The dataset was evenly divided into training and test sets.

The training set is the same 25,000 labeled reviews used to induce word vectors with the model.

Again, a linear SVM was used as a classifier after cross-validating classifier parameters on the training set.

IMDB Review Dataset Results

The model showed superior performance to the other approaches, and it performed best when concatenated with a bag-of-words representation.

The variant which utilized extra unlabeled data during training again performed best.

Changes in accuracy are small, but because the test set contains 25,000 examples, each accuracy increase of 0.1% corresponds to an additional 25 reviews classified correctly.

Features	PL04	Our Dataset	Subjectivity
Bag of Words (bnc)	85.45	87.80	87.77
Bag of Words (bΔt'c)	85.80	88.23	85.65
LDA	66.70	67.42	66.65
LSA	84.55	83.96	82.82
Our Semantic Only	87.10	87.30	86.65
Our Full	84.65	87.44	86.19
Our Full, Additional Unlabeled	87.05	87.99	87.22
Our Semantic + Bag of Words (bnc)	88.30	88.28	88.58
Our Full + Bag of Words (bnc)	87.85	88.33	88.45
Our Full, Add'l Unlabeled + Bag of Words (bnc)	88.90	88.89	88.13
Bag of Words SVM (Pang and Lee, 2004)	87.15	N/A	90.00
Contextual Valence Shifters (Kennedy and Inkpen, 2006)	86.20	N/A	N/A
tf.Δidf Weighting (Martineau and Finin, 2009)	88.10	N/A	N/A
Appraisal Taxonomy (Whitelaw et al., 2005)	90.20	N/A	N/A

Subjectivity Detection

A classifier is trained to decide whether a given sentence is either:

- subjective (expressing opinion)
- objective (expressing fact)

The dataset from Pang and Lee (2004) is again used, as it contains objective sentences from movie review summaries and objective sentences from movie plot summaries.

“This task is substantially different from the review classification task because it uses sentences as opposed to entire documents and the target concept is subjectivity instead of opinion polarity.”

Subjectivity Detection Results

The model again provided superior features when compared to other SVMs.

The 10,000 examples were randomly split into 10 folds and 10-fold cross validation accuracy using the SVM training protocol of Pang and Lee (2004) is reported.

Features	PL04	Our Dataset	Subjectivity
Bag of Words (bnc)	85.45	87.80	87.77
Bag of Words (bΔt'c)	85.80	88.23	85.65
LDA	66.70	67.42	66.65
LSA	84.55	83.96	82.82
Our Semantic Only	87.10	87.30	86.65
Our Full	84.65	87.44	86.19
Our Full, Additional Unlabeled	87.05	87.99	87.22
Our Semantic + Bag of Words (bnc)	88.30	88.28	88.58
Our Full + Bag of Words (bnc)	87.85	88.33	88.45
Our Full, Add'l Unlabeled + Bag of Words (bnc)	88.90	88.89	88.13
Bag of Words SVM (Pang and Lee, 2004)	87.15	N/A	90.00
Contextual Valence Shifters (Kennedy and Inkpen, 2006)	86.20	N/A	N/A
tf.Δidf Weighting (Martineau and Finin, 2009)	88.10	N/A	N/A
Appraisal Taxonomy (Whitelaw et al., 2005)	90.20	N/A	N/A

Discussion

A vector space model that learns word representations capturing semantic and sentiment information was presented.

The topical component of the model was parameterized to capture word representations instead of latent topics.

This method performed better than LDA, which models topics directly.

The unsupervised model was extended to incorporate sentiment information as well.

Thanks!