Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions

Richard Socher **Jeffrey Pennington*** **Eric H. Huang** **Andrew Y. Ng** **Christopher D. Manning**
Computer Science Department, Stanford University, Stanford, CA 94305, USA
*SLAC National Accelerator Laboratory, Stanford University, Stanford, CA 94309, USA
richard@socher.org {jpennin,ehhuang,ang,manning}@stanford.edu ang@cs.stanford.edu

Abstract

We introduce a novel machine learning framework based on recursive autoencoders for sentence-level prediction of sentiment label distributions. Our method learns vector space representations for multi-word phrases. In sentiment prediction tasks these representations outperform other state-of-the-art approaches on commonly used datasets, such as movie reviews, without using any pre-defined sentiment lexica or polarity shifting rules. We also evaluate the model's ability to predict sentiment distributions on a new dataset based on confessions from the experience project. The dataset consists of personal user stories annotated with multiple labels which, when aggregated, form a multinomial distribution that captures emotional reactions. Our algorithm can more accurately predict distributions over such labels compared to several competitive baselines.

1 Introduction

The ability to identify sentiments about personal experiences, products, movies etc. is crucial to understand user generated content in social networks, blogs or product reviews. Detecting sentiment in these data is a challenging task which has recently spawned a lot of interest (Pang and Lee, 2008).

Current baseline methods often use bag-of-words representations which cannot properly capture more complex linguistic phenomena in sentiment analysis (Pang et al., 2002). For instance, while the two phrases "white blood cells destroying an infection" and "an infection destroying white blood cells" have the same bag-of-words representation, the former is a positive reaction while the later is very negative. More advanced methods such as (Nakagawa et al., 2010) that can capture such phenomena use many manually constructed resources (sentiment lexica, parsers, polarity-shifting rules). This limits the applicability of these methods to a broader range of tasks and languages. Lastly, almost all previous work is based on single, positive/negative categories or scales such as star ratings. Examples are movie reviews (Pang and Lee, 2005), opinions (Wiebe et al., 2005), customer reviews (Ding et al., 2008) or multiple aspects of restaurants (Snyder and Barzilay, 2007). Such a one-dimensional scale does not accurately reflect the complexity of human emotions and sentiments.

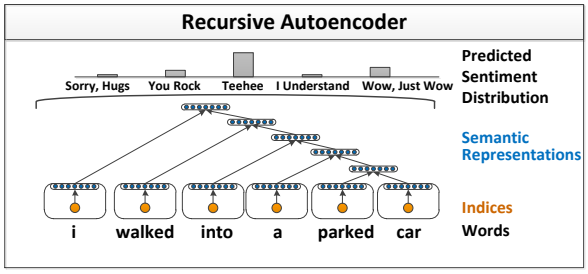


Figure 1: Illustration of our recursive autoencoder architecture which learns semantic vector representations of phrases. Word indices (orange) are first mapped into a semantic vector space (blue). Then they are recursively merged by the same autoencoder network into a fixed length sentence representation. The vectors at each node are used as features to predict a distribution over sentiment labels.

In this work, we seek to address three issues. (i) Instead of using a bag-of-words representation, our model exploits hierarchical structure and uses compositional semantics to understand sentiment. (ii) Our system can be trained both on unlabeled domain data and on supervised sentiment data and does not require any language-specific sentiment lexica,

parsers, etc. (iii) Rather than limiting sentiment to a positive/negative scale, we predict a multidimensional distribution over several complex, interconnected sentiments.

We introduce an approach based on semi-supervised, recursive autoencoders (RAE) which use as input continuous word vectors. Fig. 1 shows an illustration of the model which learns vector representations of phrases and full sentences as well as their hierarchical structure from unsupervised text. We extend our model to also learn a distribution over sentiment labels at each node of the hierarchy.

We evaluate our approach on several standard datasets where we achieve state-of-the art performance. Furthermore, we show results on the recently introduced experience project (EP) dataset (Potts, 2010) that captures a broader spectrum of human sentiments and emotions. The dataset consists of very personal confessions anonymously made by people on the experience project website *www.experienceproject.com*. Confessions are labeled with a set of five reactions by other users. Reaction labels are *you rock* (expressing approvement), *tehee* (amusement), *I understand*, *Sorry, hugs* and *Wow, just wow* (displaying shock). For evaluation on this dataset we predict both the label with the most votes as well as the full distribution over the sentiment categories. On both tasks our model outperforms competitive baselines. A set of over 31,000 confessions as well as the code of our model are available at *www.socher.org*.

After describing the model in detail, we evaluate it qualitatively by analyzing the learned n-gram vector representations and compare quantitatively against other methods on standard datasets and the EP dataset.

2 Semi-Supervised Recursive Autoencoders

Our model aims to find vector representations for variable-sized phrases in either unsupervised or semi-supervised training regimes. These representations can then be used for subsequent tasks. We first describe neural word representations and then proceed to review a related recursive model based on autoencoders, introduce our recursive autoencoder (RAE) and describe how it can be modified to jointly learn phrase representations, phrase structure and sentiment distributions.

2.1 Neural Word Representations

We represent words as continuous vectors of parameters. We explore two settings. In the first setting we simply initialize each word vector $x \in \mathbb{R}^n$ by sampling it from a zero mean Gaussian distribution: $x \sim \mathcal{N}(0, \sigma^2)$. These word vectors are then stacked into a word embedding matrix $L \in \mathbb{R}^{n \times |V|}$, where $|V|$ is the size of the vocabulary. This initialization works well in supervised settings where a network can subsequently modify these vectors to capture certain label distributions.

In the second setting, we pre-train the word vectors with an unsupervised neural language model (Bengio et al., 2003; Collobert and Weston, 2008). These models jointly learn an embedding of words into a vector space and use these vectors to predict how likely a word occurs given its context. After learning via gradient ascent the word vectors capture syntactic and semantic information from their co-occurrence statistics.

In both cases we can use the resulting matrix of word vectors L for subsequent tasks as follows. Assume we are given a sentence as an ordered list of m words. Each word has an associated vocabulary index k into the embedding matrix which we use to retrieve the word's vector representation. Mathematically, this look-up operation can be seen as a simple projection layer where we use a binary vector b which is zero in all positions except at the kth index,

$$x_i = Lb_k \in \mathbb{R}^n. \tag{1}$$

In the remainder of this paper, we represent a sentence (or any n-gram) as an ordered list of these vectors $(x_1, \ldots, x_m)$. This word representation is better suited to autoencoders than the binary number representations used in previous related autoencoder models such as the recursive autoassociative memory (RAAM) model (Pollack, 1990; Voegtlin and Dominey, 2005) or recurrent neural networks (Elman, 1991) since sigmoid units are inherently continuous. Pollack circumvented this problem by having vocabularies with only a handful of words and by manually defining a threshold to binarize the resulting vectors.

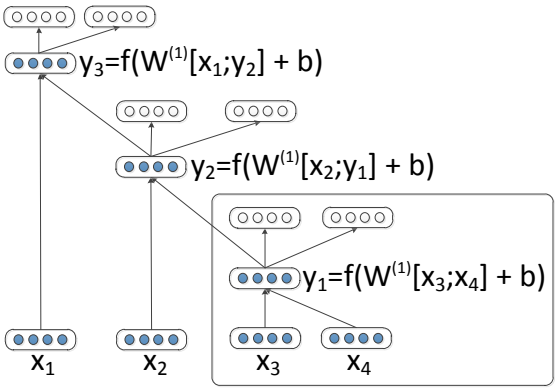


Figure 2: Illustration of an application of a recursive autoencoder to a binary tree. The nodes which are not filled are only used to compute reconstruction errors. A standard autoencoder (in box) is re-used at each node of the tree.

2.2 Traditional Recursive Autoencoders

The goal of autoencoders is to learn a representation of their inputs. In this section we describe how to obtain a reduced dimensional vector representation for sentences.

In the past autoencoders have only been used in setting where the tree structure was given a-priori. We review this setting before continuing with our model which does not require a given tree structure. Fig. 2 shows an instance of a recursive autoencoder (RAE) applied to a given tree. Assume we are given a list of word vectors $x = (x_1, \dots, x_m)$ as described in the previous section as well as a binary tree structure for this input in the form of branching triplets of parents with children: $(p \rightarrow c_1 c_2)$. Each child can be either an input word vector x_i or a nonterminal node in the tree. For the example in Fig. 2, we have the following triplets: $((y_1 \rightarrow x_3 x_4), (y_2 \rightarrow x_2 y_1), (y_1 \rightarrow x_1 y_2))$. In order to be able to apply the same neural network to each pair of children, the hidden representations y_i have to have the same dimensionality as the x_i's.

Given this tree structure, we can now compute the parent representations. The first parent vector y_1 is computed from the children $(c_1, c_2) = (x_3, x_4)$:

$$p \quad = \quad f(W^{(1)}[c_1; c_2] + b^{(1)}), \tag{2}$$

where we multiplied a matrix of parameters $W^{(1)} \in \mathbb{R}^{n \times 2n}$ by the concatenation of the two children. After adding a bias term we applied an element-wise activation function such as $\tanh$ to the resulting vector. One way of assessing how well this n-dimensional vector represents its children is to try to reconstruct the children in a reconstruction layer:

$$\left[c_1'; c_2'\right] \quad = \quad W^{(2)} p + b^{(2)}. \tag{3}$$

During training, the goal is to minimize the reconstruction errors of this input pair. For each pair, we compute the Euclidean distance between the original input and its reconstruction:

$$E_{rec}([c_1; c_2]) = \frac{1}{2} \left|\left| [c_1; c_2] - \left[c_1'; c_2'\right] \right|\right|^2. \tag{4}$$

This model of a standard autoencoder is boxed in Fig. 2. Now that we have defined how an autoencoder can be used to compute an n-dimensional vector representation (p) of two n-dimensional children (c_1, c_2), we can describe how such a network can be used for the rest of the tree.

Essentially, the same steps repeat. Now that y_1 is given, we can use Eq. 2 to compute y_2 by setting the children to be $(c_1, c_2) = (x_2, y_1)$. Again, after computing the intermediate parent vector y_2, we can assess how well this vector capture the content of the children by computing the reconstruction error as in Eq. 4. The process repeat until the full tree is constructed and we have a reconstruction error at each nonterminal node. This model is similar to the RAAM model (Pollack, 1990) which also requires a fixed tree structure.

2.3 Unsupervised Recursive Autoencoder for Structure Prediction

Now, assume there is no tree structure given for the input vectors in x. The goal of our structure-prediction RAE is to minimize the reconstruction error of all vector pairs of children in a tree. We define $A(x)$ as the set of all possible trees that can be built from an input sentence x. Further, let $T(y)$ be a function that returns the triplets of a tree indexed by s of all the non-terminal nodes in a tree. Using the reconstruction error of Eq. 4, we compute

$$RAE_\theta(x) = \underset{y \in A(x)}{\arg\min} \sum_{s \in T(y)} E_{rec}([c_1; c_2]_s) \tag{5}$$

We now describe a greedy approximation that constructs such a tree.

Greedy Unsupervised RAE. For a sentence with m words, we apply the autoencoder recursively. It takes the first pair of neighboring vectors, defines them as potential children of a phrase $(c_1; c_2) = (x_1; x_2)$, concatenates them and gives them as input to the autoencoder. For each word pair, we save the potential parent node p and the resulting reconstruction error.

After computing the score for the first pair, the network is shifted by one position and takes as input vectors $(c_1, c_2) = (x_2, x_3)$ and again computes a potential parent node and a score. This process repeats until it hits the last pair of words in the sentence: $(c_1, c_2) = (x_{m-1}, x_m)$. Next, it selects the pair which had the lowest reconstruction error (E_{rec}) and its parent representation p will represent this phrase and replace both children in the sentence word list. For instance, consider the sequence (x_1, x_2, x_3, x_4) and assume the lowest E_{rec} was obtained by the pair (x_3, x_4). After the first pass, the new sequence then consists of $(x_1, x_2, p_{(3,4)})$. The process repeats and treats the new vector $p_{(3,4)}$ like any other input vector. For instance, subsequent states could be either: $(x_1, p_{(2,(3,4))})$ or $(p_{(1,2)}, p_{(3,4)})$. Both states would then finish with a deterministic choice of collapsing the remaining two states into one parent to obtain $(p_{(1,(2,(3,4)))})$ or $(p_{((1,2),(3,4))})$ respectively. The tree is then recovered by unfolding the collapsing decisions.

The resulting tree structure captures as much of the single-word information as possible (in order to allow reconstructing the word vectors) but does not necessarily follow standard syntactic constraints. We also experimented with a method that finds better solutions to Eq. 5 based on CKY-like beam search algorithms (Socher et al., 2010; Socher et al., 2011) but the performance is similar and the greedy version is much faster.

Weighted Reconstruction. One problem with simply using the reconstruction error of both children equally as describe in Eq. 4 is that each child could represent a different number of previously collapsed words and is hence of bigger importance for the overall meaning reconstruction of the sentence. For instance in the case of $(x_1, p_{(2,(3,4))})$ one would like to give more importance to reconstructing p than x_1. We capture this desideratum by adjusting the reconstruction error. Let n_1, n_2 be the number of words underneath a current potential child, we re-define the reconstruction error to be $E_{rec}([c_1; c_2]; \theta) =$

$$\frac{n_1}{n_1+n_2}\left|\left|c_1 - c_1'\right|\right|^2 + \frac{n_2}{n_1+n_2}\left|\left|c_2 - c_2'\right|\right|^2 \quad (6)$$

Length Normalization. One of the goals of RAEs is to induce semantic vector representations that allow us to compare n-grams of different lengths. The RAE tries to lower reconstruction error of not only the bigrams but also of nodes higher in the tree. Unfortunately, since the RAE computes the hidden representations it then tries to reconstruct, it can just lower reconstruction error by making the hidden layer very small in magnitude. To prevent such undesirable behavior, we modify the hidden layer such that the resulting parent representation always has length one, after computing p as in Eq. 2, we simply set: $p = \frac{p}{||p||}$.

2.4 Semi-Supervised Recursive Autoencoders

So far, the RAE was completely unsupervised and induced general representations that capture the semantics of multi-word phrases.In this section, we extend RAEs to a semi-supervised setting in order to predict a sentence- or phrase-level target distribution t.[1]

One of the main advantages of the RAE is that each node of the tree built by the RAE has associated with it a distributed vector representation (the parent vector p) which could also be seen as features describing that phrase. We can leverage this representation by adding on top of each parent node a simple $\mathrm{softmax}$ layer to predict class distributions:

$$d(p; \theta) = \mathrm{softmax}(W^{label} p). \quad (7)$$

Assuming there are K labels, $d \in \mathbb{R}^K$ is a K-dimensional multinomial distribution and $\sum_{k=1} d_k = 1$. Fig. 3 shows such a semi-supervised RAE unit. Let t_k be the kth element of the multinomial target label distribution t for one entry. The $\mathrm{softmax}$ layer's outputs are interpreted as conditional probabilities $d_k = p(k|[c_1; c_2])$, hence the cross-entropy error is

$$E_{cE}(p, t; \theta) = -\sum_{k=1}^{K} t_k \log d_k(p; \theta). \quad (8)$$

[1]For the binary label classification case, the distribution is of the form $[1, 0]$ for class 1 and $[0, 1]$ for class 2.

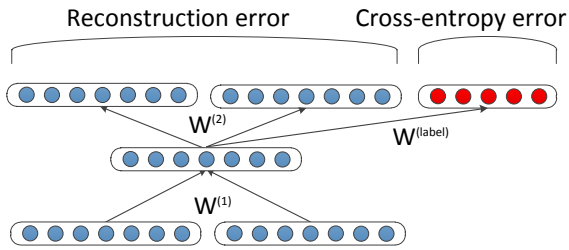


Figure 3: Illustration of an RAE unit at a nonterminal tree node. Red nodes show the supervised *softmax* layer for label distribution prediction.

Using this cross-entropy error for the label and the reconstruction error from Eq. 6, the final semi-supervised RAE objective over (sentences,label) pairs (x, t) in a corpus becomes

$$J = \frac{1}{N}\sum_{(x,t)} E(x,t;\theta) + \frac{\lambda}{2}||\theta||^2, \qquad (9)$$

where we have an error for each entry in the training set that is the sum over the error at the nodes of the tree that is constructed by the greedy RAE:

$$E(x,t;\theta) = \sum_{s \in T(RAE_\theta(x))} E([c_1;c_2]_s, p_s, t, \theta).$$

The error at each nonterminal node is the weighted sum of reconstruction and cross-entropy errors, $E([c_1;c_2]_s, p_s, t, \theta) =$

$$\alpha E_{rec}([c_1;c_2]_s;\theta) + (1-\alpha)E_{cE}(p_s, t;\theta).$$

The hyperparameter α weighs reconstruction and cross-entropy error. When minimizing the cross-entropy error of this softmax layer, the error will backpropagate and influence both the RAE parameters and the word representations. Initially, words such as *good* and *bad* have very similar representations. This is also the case for Brown clusters and other methods that use only cooccurrence statistics in a small window around each word. When learning with positive/negative sentiment, the word embeddings get modified and capture less syntactic and more sentiment information.

In order to predict the sentiment distribution of a sentence with this model, we use the learned vector representation of the top tree node and train a simple logistic regression classifier.

3 Learning

Let $\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(1)}, W^{label}, L)$ be the set of our model parameters, then the gradient becomes:

$$\frac{\partial J}{\partial \theta} = \frac{1}{N}\sum_{(x,t)} \frac{\partial E(x,t;\theta)}{\partial \theta} + \lambda\theta. \qquad (10)$$

To compute this gradient, we first greedily construct all trees and then derivatives for these trees are computed efficiently via backpropagation through structure (Goller and Küchler, 1996). Because the algorithm is greedy and the derivatives of the supervised cross-entropy error also modify the matrix $W^{(1)}$, this objective is not necessarily continuous and a step in the gradient descent direction may not necessarily decrease the objective. However, we found that L-BFGS run over the complete training data (batch mode) to minimize the objective works well in practice, and that convergence is smooth, with the algorithm typically finding a good solution quickly.

4 Experiments

We first describe the new experience project (EP) dataset, results of standard classification tasks on this dataset and how to predict its sentiment label distributions. We then show results on other commonly used datasets and conclude with an analysis of the important parameters of the model.

In all experiments involving our model, we represent words using 100-dimensional word vectors. We explore the two settings mentioned in Sec. 2.1. We compare performance on standard datasets when using randomly initialized word vectors (random word init.) or word vectors trained by the model of Collobert and Weston (2008) and provided by Turian et al. (2010).[2] These vectors were trained on an unlabeled corpus of the English Wikipedia. Note that alternatives such as Brown clusters are not suitable since they do not capture sentiment information (*good* and *bad* are usually in the same cluster) and cannot be modified via backpropagation.

[2] http://metaoptimize.com/projects/wordreprs/

Corpus	K	Instances	Distr.(+/-)	Avg$\|W\|$
MPQA	2	10,624	0.31/0.69	3
MR	2	10,662	0.5/0.5	22
EP	5	31,675	.2/.2/.1/.4/.1	113
EP≥ 4	5	6,129	.2/.2/.1/.4/.1	129

Table 1: Statistics on the different datasets. K is the number of classes. Distr. is the distribution of the different classes (in the case of 2, the positive/negative classes, for EP the rounded distribution of total votes in each class). $|W|$ is the average number of words per instance. We use EP≥ 4, a subset of entries with at least 4 votes.

4.1 EP Dataset: The Experience Project

The confessions section of the experience project website[3] lets people anonymously write short personal stories or "confessions". Once a story is on the site, each user can give a single vote to one of five label categories (with our interpretation):

1 Sorry, Hugs: User offers condolences to author.
2. You Rock: Indicating approval, congratulations.
3. Teehee: User found the anecdote amusing.
4. I Understand: Show of empathy.
5. Wow, Just Wow: Expression of surprise,shock.

The EP dataset has 31,676 confession entries, a total number of 74,859 votes for the 5 labels above, the average number of votes per entry is 2.4 (with a variance of 33). For the five categories, the numbers of votes are $[14,816; 13,325; 10,073; 30,844; 5,801]$. Since an entry with less than 4 votes is not very well identified, we train and test only on entries with at least 4 total votes. There are 6,129 total such entries. The distribution over total votes in the 5 classes is similar: $[0.22; 0.2; 0.11; 0.37; 0.1]$. The average length of entries is 129 words. Some entries contain multiple sentences. In these cases, we average the predicted label distributions from the sentences. Table 1 shows statistics of this and other commonly used sentiment datasets (which we compare on in later experiments). Table 2 shows example entries as well as gold and predicted label distributions as described in the next sections.

Compared to other datasets, the EP dataset contains a wider range of human emotions that goes far beyond positive/negative product or movie reviews. Each item is labeled with a multinomial distribution over interconnected response categories. This is in contrast to most other datasets (including multi-aspect rating) where several distinct aspects are rated independently but on the same scale. The topics range from generic happy statements, daily clumsiness reports, love, loneliness, to relationship abuse and suicidal notes. As is evident from the total number of label votes, the most common user reaction is one of empathy and an ability to relate to the authors experience. However, some stories describe horrible scenarios that are not common and hence receive more offers of condolence. In the following sections we show some examples of stories with predicted and true distributions but refrain from listing the most horrible experiences.

For all experiments on the EP dataset, we split the data into train (49%), development (21%) and test data (30%).

4.2 EP: Predicting the Label with Most Votes

The first task for our evaluation on the EP dataset is to simply predict the single class that receives the most votes. In order to compare our novel joint phrase representation and classifier learning framework to traditional methods, we use the following baselines:

Random Since there are five classes, this gives 20% accuracy.

Most Frequent Selecting the class which most frequently has the most votes (the class *I understand*).

Baseline 1: Binary BoW This baseline uses logistic regression on binary bag-of-word representations that are 1 if a word is present and 0 otherwise.

Baseline 2: Features This model is similar to traditional approaches to sentiment classification in that it uses many hand-engineered resources. We first used a spell-checker and Wordnet to map words and their misspellings to synsets to reduce the total number of words. We then replaced sentiment words with a sentiment category identifier using the sentiment lexica of the Harvard Inquirer (Stone, 1966) and LIWC (Pennebaker et al., 2007). Lastly, we used tf-idf weighting on the bag-of-word representations and trained an SVM.

[3] http://www.experienceproject.com/confessions.php

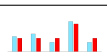
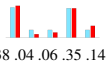
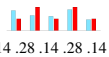
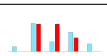
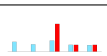
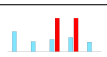
KL	Predicted&Gold	V.	Entry (Shortened if it ends with ...)
.03	 .16 .16 .16 .33 .16	6	I reguarly shoplift. I got caught once and went to jail, but I've found that this was not a deterrent. I don't buy groceries, I don't buy school supplies for my kids, I don't buy gifts for my kids, we don't pay for movies, and I dont buy most incidentals for the house (cleaning supplies, toothpaste, etc.)...
.03	 .38 .04 .06 .35 .14	165	i am a very succesfull buissnes man.i make good money but i have been addicted to crack for 13 years.i moved 1 hour away from my dealers 10 years ago to stop using now i dont use daily but once a week usally friday nights. i used to use 1 or 2 hundred a day now i use 4 or 5 hundred on a friday.my problem is i am a funcational addict...
.05	 .14 .28 .14 .28 .14	7	Hi there, Im a guy that loves a girl, the same old bloody story... I met her a while ago, while studying, she Is so perfect, so mature and yet so lonely, I get to know her and she get ahold of me, by opening her life to me and so did I with her, she has been the first person, male or female that has ever made that bond with me,...
.07	 .27 .18 .00 .45 .09	11	be kissing you right now. i should be wrapped in your arms in the dark, but instead i've ruined everything. i've piled bricks to make a wall where there never should have been one. i feel an ache that i shouldn't feel because i've never had you close enough. we've never touched, but i still feel as though a part of me is missing. ...
.05		23	Dear Love, I just want to say that I am looking for you. Tonight I felt the urge to write, and I am becoming more and more frustrated that I have not found you yet. I'm also tired of spending so much heart on an old dream. ...
.05		5	I wish I knew somone to talk to here.
.06		24	I loved her but I screwed it up. Now she's moved on. I'll never have her again. I don't know if I'll ever stop thinking about her.
.06		5	i am 13 years old and i hate my father he is alwas geting drunk and do's not care about how it affects me or my sisters i want to care but the truthis i dont care if he dies
.13		6	well i think hairy women are attractive
.35		5	As soon as I put clothings on I will go down to DQ and get a thin mint blizzard. I need it. It'll make my soul feel a bit better :)
.36		6	I am a 45 year old divoced woman, and I havent been on a date or had any significant relationship in 12 years...yes, 12 yrs. the sad thing is, Im not some dried up old granny who is no longer interested in men, I just can't meet men. (before you judge, no Im not terribly picky!) What is wrong with me?
.63		6	When i was in kindergarden i used to lock myself in the closet and eat all the candy. Then the teacher found out it was one of us and made us go two days without freetime. It might be a little late now, but sorry guys it was me haha
.92		4	My paper is due in less than 24 hours and I'm still dancing round my room!

Table 2: Example EP confessions from the test data with KL divergence between our predicted distribution (light blue, left bar on each of the 5 classes) and ground truth distribution (red bar and numbers underneath), number of votes. The 5 classes are [Sorry, Hugs ;You Rock; Teehee;I Understand;Wow, Just Wow]. Even when the KL divergence is higher, our model makes reasonable alternative label choices. Some entries are shortened.

Baseline 3: Word Vectors We can ignore the RAE tree structure and only train softmax layers directly on the pre-trained words in order to influence the word vectors. This is followed by an SVM trained on the average of the word vectors.

We also experimented with latent Dirichlet allocation (Blei et al., 2003) but performance was very low. Lastly, we ran extensive cross-validation for each of the baseline's regularization parameters.

Table 3 shows the results for predicting the class with the most votes. Even the approach that is based on sentiment lexica and other resources is outperformed by our model by almost 3%, showing that for tasks involving complex broad-range human sentiment, the often used sentiment lexica lack in coverage and traditional bag-of-words representations are not powerful enough.

Method	Accuracy
Random	20.0
Most Frequent	38.1
Baseline 1: Binary BoW	46.4
Baseline 2: Features	47.0
Baseline 3: Word Vectors	45.5
RAE (our method)	**50.1**

Table 3: Accuracy of predicting the class with most votes.

4.3 EP: Predicting Sentiment Distributions

We now turn to evaluating our distribution-prediction approach. In both this and the previous maximum label task, we backprop using the gold multinomial distribution as a target. Since we maximize likelihood and because we want to predict a distribution that is closest to the distribution of labels that people would assign to a story, we evaluate using KL divergence: $KL(g||p) = \sum_i g_i \log(g_i/p_i)$, where g is the gold distribution and p is the predicted

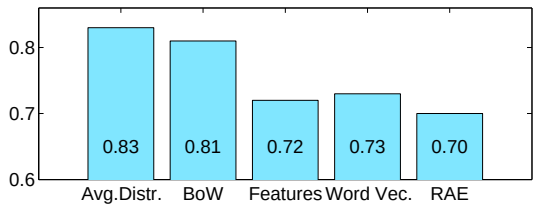


Figure 4: Average KL-divergence between gold and predicted sentiment distributions (lower is better).

one. We report the average KL divergence, where a smaller value indicates better predictive power. To get an idea of the values of KL divergence, predicting random distributions gives a an average of 1.2 in KL divergence, predicting simply the average distribution in the training data give 0.83. Fig. 4 shows that our RAE-based model outperforms the other baselines. Table 2 shows EP example entries with predicted and gold distributions, as well as numbers of votes.

4.4 Binary Polarity Classification

In order to compare our approach to other methods we also show results on commonly used sentiment datasets: movie reviews[4] (MR) (Pang and Lee, 2005) and opinions[5] (MPQA) (Wiebe et al., 2005).We give statistical information on these and the EP corpus in Table 1.

We compare to the state-of-the-art system of (Nakagawa et al., 2010), a dependency tree based classification method that uses CRFs with hidden variables. We use the same training and testing regimen (10-fold cross validation) as well as their baselines: majority phrase voting using sentiment and reversal lexica; rule-based reversal using a dependency tree; Bag-of-Features and their full Tree-CRF model. As shown in Table 5, our algorithm outperforms their approach on both datasets. For the movie review (MR) data set, we do not use any hand-designed lexica. An error analysis on the MPQA dataset showed several cases of single words which never occurred during training. Correctly classifying these instances can only be the result of having them in the original sentiment lexicon. Hence, for the experiment on MPQA we added the same sentiment lexicon that (Nakagawa et al., 2010) used in their system to our training set. This improved accuracy from 86.0 to 86.4.Using the pre-trained word vectors boosts performance by less than 1% compared to randomly initialized word vectors (setting: random word init). This shows that our method can work well even in settings with little training data. We visualize the semantic vectors that the recursive autoencoder learns by listing n-grams that give the highest probability for each polarity. Table 4 shows such n-grams for different lengths when the RAE is trained on the movie review polarity dataset.

On a 4-core machine, training time for the smaller corpora such as the movie reviews takes around 3 hours and for the larger EP corpus around 12 hours until convergence. Testing of hundreds of movie reviews takes only a few seconds.

[4] www.cs.cornell.edu/people/pabo/movie-review-data/

[5] www.cs.pitt.edu/mpqa/

Method	MR	MPQA
Voting with two lexica	63.1	81.7
Rule-based reversal on trees	62.9	82.8
Bag of features with reversal	76.4	84.1
Tree-CRF (Nakagawa et al,'10)	77.3	86.1
RAE (random word init.)	76.8	85.7
RAE (our method)	**77.7**	**86.4**

Table 5: Accuracy of sentiment classification on movie review polarity (MR) and the MPQA dataset.

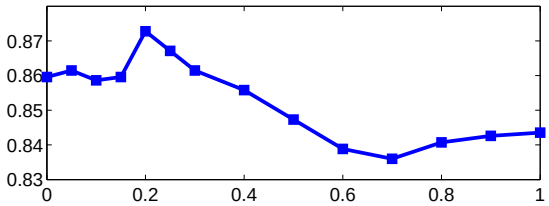


Figure 5: Accuracy on the development split of the MR polarity dataset for different weightings of reconstruction error and supervised cross-entropy error: $err = \alpha E_{rec} + (1-\alpha)E_{cE}$.

4.5 Reconstruction vs. Classification Error

In this experiment, we show how the hyperparameter α influences accuracy on the development set of one of the cross-validation splits of the MR dataset. This parameter essentially trade-off the supervised and unsupervised parts of the objective. Fig. 5 shows that a larger focus on the supervised objective is important but that a weight of $\alpha = 0.2$ for the reconstruction error prevents overfitting and achieves the highest performance.

n	Most negative n-grams	Most positive n-grams
1	bad; boring; dull; flat; pointless; tv; neither; pretentious; badly; worst; lame; mediocre; lack; routine; loud; bore; barely; stupid; tired; poorly; suffers; heavy;nor; choppy; superficial	touching; enjoyable; powerful; warm; moving; culture; flaws; provides; engrossing; wonderful; beautiful; quiet; socio-political; thoughtful; portrait; refreshingly; chilling; rich; beautifully; solid;
2	how bad; by bad; dull .; for bad; to bad; boring .; , dull; are bad; that bad; boring ,; , flat; pointless .; badly by; on tv; so routine; lack the; mediocre .; a generic; stupid ,; abysmally pathetic	the beautiful; moving,; thoughtful and; , inventive; solid and; a beautiful; a beautifully; and hilarious; with dazzling; provides the; provides.; and inventive; as powerful; moving and; a moving; a powerful
3	. too bad; exactly how bad; and never dull; shot but dull; is more boring; to the dull; dull, UNK; it is bad; or just plain; by turns pretentious; manipulative and contrived; bag of stale; is a bad; the whole mildly; contrived pastiche of; from this choppy; stale material.	funny and touching; a small gem; with a moving; cuts, fast; , fine music; smart and taut; culture into a; romantic , riveting; ... a solid; beautifully acted .; , gradually reveals; with the chilling; cast of solid; has a solid; spare yet audacious; ... a polished; both the beauty;
5	boring than anything else.; a major waste ... generic; nothing i hadn't already; ,UNK plotting;superficial; problem ? no laughs.; ,just horribly mediocre .; dull, UNK feel.; there's nothing exactly wrong; movie is about a boring; essentially a collection of bits	reminded us that a feel-good; engrossing, seldom UNK,; between realistic characters showing honest; a solid piece of journalistic; easily the most thoughtful fictional; cute, funny, heartwarming; with wry humor and genuine; engrossing and ultimately tragic.;
8	loud, silly, stupid and pointless.; dull, dumb and derivative horror film.; UNK's film, a boring, pretentious; this film biggest problem ? no laughs.; film in the series looks and feels tired; do draw easy chuckles but lead nowhere.; stupid, infantile, redundant, sloppy	shot in rich , shadowy black-and-white , devils an escapist confection that 's pure entertainment .; , deeply absorbing piece that works as a; ... one of the most ingenious and entertaining; film is a riveting , brisk delight .; bringing richer meaning to the story 's;

Table 4: Examples of n-grams ($n = 1, 2, 3, 5, 8$) from the test data of the movie polarity dataset for which our model predicts the most positive and most negative responses.

5 Related Work

5.1 Autoencoders and Deep Learning

Autoencoders are neural networks that learn a reduced dimensional representation of fixed-size inputs such as image patches or bag-of-word representations of text documents. They can be used to efficiently learn feature encodings which are useful for classification. Recently, Mirowski et al. (2010) learn dynamic autoencoders for documents in a bag-of-words format which, like ours, combine supervised and reconstruction objectives.

The idea of applying an autoencoder in a recursive setting was introduced by Pollack (1990). Pollack's recursive auto-associative memories (RAAMs) are similar to ours in that they are a connectionst, feed-forward model. However, RAAMs learn vector representations only for fixed recursive data structures, whereas our RAE builds this recursive data structure. More recently, (Voegtlin and Dominey, 2005) introduced a linear modification to RAAMs that is able to better generalize to novel combinations of previously seen constituents. One of the major shortcomings of previous applications of recursive autoencoders to natural language sentences was their binary word representation as discussed in Sec. 2.1.

Recently, (Socher et al., 2010; Socher et al., 2011) introduced a max-margin framework based on recursive neural networks (RNNs) for labeled structure prediction. Their models are applicable to natural language and computer vision tasks such as parsing or object detection. The current work is related in that it uses a recursive deep learning model. However, RNNs require labeled tree structures and use a supervised score at each node. Instead, RAEs learn hierarchical structures that are trying to capture as much of the the original word vectors as possible. The learned structures are not necessarily syntactically plausible but can capture more of the semantic content of the word vectors. Other recent deep learning methods for sentiment analysis include (Maas et al., 2011).

5.2 Sentiment Analysis

Pang et al. (2002) were one of the first to experiment with sentiment classification. They show that simple bag-of-words approaches based on Naive Bayes, MaxEnt models or SVMs are often insufficient for predicting sentiment of documents even though they work well for general topic-based document classification. Even adding specific negation words, bigrams or part-of-speech information to these models did not add significant improvements. Other document-level sentiment work includes (Turney, 2002; Dave et al., 2003; Beineke et al., 2004; Pang and Lee, 2004). For further references, see (Pang and Lee, 2008).

Instead of document level sentiment classification, (Wilson et al., 2005) analyze the contextual polarity of phrases and incorporate many well designed features including dependency trees. They also show improvements by first distinguishing be-

tween neutral and polar sentences. Our model naturally incorporates the recursive interaction between context and polarity words in sentences in a unified framework while simultaneously learning the necessary features to make accurate predictions. Other approaches for sentence-level sentiment detection include (Yu and Hatzivassiloglou, 2003; Grefenstette et al., 2004; Ikeda et al., 2008).

Most previous work is centered around a given sentiment lexicon or building one via heuristics (Kim and Hovy, 2007; Esuli and Sebastiani, 2007), manual annotation (Das and Chen, 2001) or machine learning techniques (Turney, 2002). In contrast, we do not require an initial or constructed sentiment lexicon of positive and negative words. In fact, when training our approach on documents or sentences, it jointly learns such lexica for both single words and n-grams (see Table 4). (Mao and Lebanon, 2007) propose isotonic conditional random fields and differentiate between local, sentence-level and global, document-level sentiment.

The work of (Polanyi and Zaenen, 2006; Choi and Cardie, 2008) focuses on manually constructing several lexica and rules for both polar words and related content-word negators, such as "prevent cancer", where *prevent* reverses the negative polarity of *cancer*. Like our approach they capture compositional semantics. However, our model does so without manually constructing any rules or lexica.

Recently, (Velikovich et al., 2010) showed how to use a seed lexicon and a graph propagation framework to learn a larger sentiment lexicon that also includes polar multi-word phrases such as "once in a life time". While our method can also learn multi-word phrases it does not require a seed set or a large web graph. (Nakagawa et al., 2010) introduced an approach based on CRFs with hidden variables with very good performance. We compare to their state-of-the-art system. We outperform them on the standard corpora that we tested on without requiring external systems such as POS taggers, dependency parsers and sentiment lexica. Our approach jointly learns the necessary features and tree structure.

In multi-aspect rating (Snyder and Barzilay, 2007) one finds several distinct aspects such as food or service in a restaurant and then rates them on a fixed linear scale such as 1-5 stars, where all aspects could obtain just 1 star or all aspects could obtain 5 stars independently. In contrast, in our method a single aspect (a complex reaction to a human experience) is predicted not in terms of a fixed scale but in terms of a *multinomial distribution* over several interconnected, sometimes mutually exclusive emotions. A single story cannot simultaneously obtain a strong reaction in different emotional responses (by virtue of having to sum to one).

6 Conclusion

In this paper we presented a novel algorithm that can accurately predict sentence-level sentiment distributions. Without using any hand-engineered resources such as sentiment lexica, parsers or sentiment shifting rules, our model achieves state-of-the-art performance on commonly used sentiment datasets. Furthermore, we introduce a new dataset that contains distributions over a broad range of human emotions. Our evaluation shows that our model can more accurately predict these distributions than other models.

Acknowledgments

We gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, AFRL, or the US government. This work was also supported in part by the DARPA Deep Learning program under contract number FA8650-10-C-7020.
We thank Chris Potts for help with the EP data set, Raymond Hsu, Bozhi See, and Alan Wu for letting us use their system as a baseline and Jiquan Ngiam, Quoc Le, Gabor Angeli and Andrew Maas for their feedback.

References

P. Beineke, T. Hastie, C. D. Manning, and S. Vaithyanathan. 2004. Exploring sentiment summarization. In *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*.

Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research.*, 3:993–1022.

Y. Choi and C. Cardie. 2008. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *EMNLP*.

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167.

S. Das and M. Chen. 2001. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific Finance Association Annual Conference (APFA)*.

K. Dave, S. Lawrence, and D. M. Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of WWW*, pages 519–528.

X. Ding, B. Liu, and P. S. Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the Conference on Web Search and Web Data Mining (WSDM)*.

J. L. Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7(2-3):195–225.

A. Esuli and F. Sebastiani. 2007. Pageranking wordnet synsets: An application to opinion mining. In *Proceedings of the Association for Computational Linguistics (ACL)*.

C. Goller and A. Küchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of the International Conference on Neural Networks (ICNN-96)*.

G. Grefenstette, Y. Qu, J. G. Shanahan, and D. A. Evans. 2004. Coupling niche browsers and affect analysis for an opinion mining application. In *Proceedings of Recherche d'Information Assistée par Ordinateur (RIAO)*.

D. Ikeda, H. Takamura, L. Ratinov, and M. Okumura. 2008. Learning to shift the polarity of words for sentiment classification. In *IJCNLP*.

S. Kim and E. Hovy. 2007. Crystal: Analyzing predictive opinions on the web. In *EMNLP-CoNLL*.

A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. 2011. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of ACL*.

Y. Mao and G. Lebanon. 2007. Isotonic Conditional Random Fields and Local Sentiment Flow. In *NIPS*.

P. Mirowski, M. Ranzato, and Y. LeCun. 2010. Dynamic auto-encoders for semantic indexing. In *Proceedings of the NIPS 2010 Workshop on Deep Learning*.

T. Nakagawa, K. Inui, and S. Kurohashi. 2010. Dependency tree-based sentiment classification using CRFs with hidden variables. In *NAACL*, HLT.

B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*.

B. Pang and L. Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, pages 115–124.

B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *EMNLP*.

J. W. Pennebaker, R.J. Booth, and M. E. Francis. 2007. Linguistic inquiry and word count: Liwc2007 operators manual. University of Texas.

L. Polanyi and A. Zaenen. 2006. Contextual valence shifters.

J. B. Pollack. 1990. Recursive distributed representations. *Artificial Intelligence*, 46:77–105, November.

C. Potts. 2010. On the negativity of negation. In David Lutz and Nan Li, editors, *Proceedings of Semantics and Linguistic Theory 20*. CLC Publications, Ithaca, NY.

B. Snyder and R. Barzilay. 2007. Multiple aspect ranking using the Good Grief algorithm. In *HLT-NAACL*.

R. Socher, C. D. Manning, and A. Y. Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

R. Socher, C. C. Lin, A. Y. Ng, and C. D. Manning. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *ICML*.

P. J. Stone. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. The MIT Press.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, pages 384–394.

P. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *ACL*.

L. Velikovich, S. Blair-Goldensohn, K. Hannan, and R. McDonald. 2010. The viability of web-derived polarity lexicons. In *NAACL*, HLT.

T. Voegtlin and P. Dominey. 2005. Linear Recursive Distributed Representations. *Neural Networks*, 18(7).

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39.

T. Wilson, J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/EMNLP*.

H. Yu and V. Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *EMNLP*.