Complex Sentiment Analysis using Recursive Autoencoders

Shashank Sashihithlu, Siddhi S. Soman
Stanford University
{sashihit, ssoman}@ stanford.edu

Advised by Richard Socher

Abstract

We use a machine learning framework based on recursive autoencoders to make sentence level predictions regarding the subject, object and sentiment of the sentence. Our model learns vector representations of multi-word phrases.
The syntactic structure of the sentence is incorporated into our model by using the Stanford Parser tree thereby enabling it to make sentiment predictions with respect to subject, object, etc.

1 Introduction

Deep learning involves learning a hierarchy of internal representations using a stack of multiple modules, each of which is trainable and can implement complex non-linear functions. Deep learning allows us to go from low level representations to high level representations and is more efficient and accurate than shallow architectures such as kernel machines. A deep architecture achieves this efficiency by trading space for time and using sparse feature encoding. We build a hierarchy of modules, where stage k measures the "compatibility" between features at level k-1 and level k, where the compatibility may be measured as the negative of log likelihood or energy. The objective at each stage is to find a set of features that would minimize the energy. The input to stage k+1 is the feature vector of stage k. Deep belief networks have been shown to perform significantly better than other techniques for problems such as sentiment analysis, image classification and handwriting analysis.

Sentiment analysis (together with opinion mining) is becoming a promising topic and has important applications in the fields of social media, customer relationship management etc. Our project aims at performing complex sentiment analysis using deep learning techniques. Previous work in [2] looked at the problem of extracting sentiments using recursive autoencoders. We tried to extend this work by performing more complex sentiment analysis which includes extracting the subject, object and the sentiment. We plan to use data comprising of 'opinionated text'. The recursive neural network architecture described in [3] jointly learns how to parse and how to represent phrases in a continuous vector space of features. This allows us to capture syntactic and compositional semantic information from natural language sentences. We use this architecture combined with the auto encoder scheme to perform complex sentiment analysis.

2 Dataset

We use the MPQA dataset which consists of opinionated text. The MPQA Opinion Corpus contains news articles from a wide variety of news sources manually annotated for opinions and other private states (i.e., beliefs, emotions, sentiments, speculations, etc.). The corpus contains 535 documents, a total of 11,114 sentences. The articles in the corpus are from 187 different foreign and U.S. news sources which date from June 2001 to May 2002.

The dataset contains annotations of different types like express subjectivity, direct-subjective, objective speech event etc. These also have

nested annotations like polarity (attribute for marking the polarity of the expression, in context, according to the nested-source), intensity etc. We try to use the annotations within this dataset to extract the different parts of the sentence like subject, object along with the sentiment.

3 Recursive Autoencoders

We represent words as continuous vectors of parameters. We represent a sentence (or any n-gram) as an ordered list of these vectors $(x_1, x_2, \dots, x_m)$. Now, assume that we are given a list of word vectors $x = (x_1, x_2, \dots, x_m)$ 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 an input word vector or a non-terminal node in the tree. For the example in Fig.1, we have the following triplets: $((y_1 \rightarrow x_3 x_4); (y_2 \rightarrow x_2 y_1); (y_3 \rightarrow x_1 y_2))$. 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 $(c1; c2) = (x3; x4)$:

$$p = f(W^{(1)}[c1; c2] + b^{(1)});$$

where we multiplied a matrix of parameters $W^{(1)} \in R^{n \times 2n}$ by the concatenation of the two children. One way of assessing how well this n-dimensional vector represents its children is to try to reconstruct the children in a reconstruction layer:

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

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:

$$E_{rec}([c1; c2]) = \frac{1}{2}\|[c_1; c_2] - [c_1'; c_2']\|^2;$$

Essentially these same steps are repeated for all triplets and the reconstruction error at nodes of the tree is found.

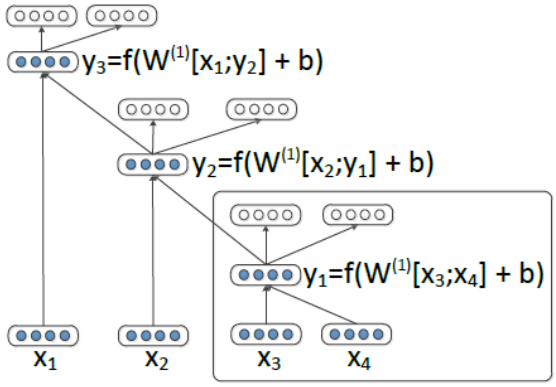

Fig 1: RAE on a binary tree

So far, the RAE was completely unsupervised and induced general representations that capture the semantics of multi-word phrases. We extend RAEs to a semi-supervised setting in order to predict a sentence or phrase level target distribution t. This is done by adding on top of each parent node a simple softmax layer to predict class distributions:

$$d(p; \theta) = softmax(W^{label}p);$$

Assuming there are K labels, $d \in \mathbb{R}^K$ is a K-dimensional multinomial distribution and $\sum_{k=1} d_k = 1$. Let t_k be the kth element of the multinomial target label distribution t for one entry. The softmax layer's outputs are interpreted as conditional probabilities $d_k = p(k|[c1; c2])$, hence the cross-entropy error is :

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

Using this cross-entropy error for the label as well as the reconstruction error, we get the total error for sentence, label pairs(x, t) in the corpus as:

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

where E is summed over all nodes for a particular sentence and is a weighted average of the reconstruction error and the cross-entropy error. This error function is minimized with respect to all the parameters by using L-BFGS.

4 Model

We use recursive autoencoders to model our data and for making predictions. The previous work included a greedy approach for structure prediction using unsupervised RAE. 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. Hence though this approach might be faster, it does not necessarily capture the entire syntactic structure.

Owing to this we chose not to use the greedy approach of tree construction. Instead we use the Stanford parser tree output as our tree representation. This representation captures the syntactic structure of the sentence. This helps us correlate the syntactic and semantic information of the sentence and helps us make better predictions.

e.g.

Sentence:

The Kimberley Provincial Hospital said it would probably know by Tuesday whether one of its patients had Congo Fever.

Parsed output:

```
(ROOT
  (S
    (NP (DT The) (NNP Kimberley) (NNP
Provincial) (NNP Hospital))
    (VP (VBD said)
      (SBAR
        (S
          (NP (PRP it))
          (VP (MD would)
            (ADVP (RB probably))
            (VP (VB know)
              (PP (IN by)
                (NP (NNP Tuesday)))
              (SBAR (IN whether)
                (S
                  (NP
                    (NP (CD one))
                    (PP (IN of)
                      (NP (PRP its) (NNS patients))))
                  (VP (VBD had)
                    (NP (NNP Congo) (NN
Fever)))))))))
    (. .)))
```

In our model, words are represented as a continuous vector of parameters. Our model learns a vector representation of phrases and sentences using RAE as shown in the figure. Our model is also extended to learn a sentence or phrase level target distribution in a semi-supervised setting using an additional softmax regression layer.

5 Methodology

We used the annotations and information provided by the MPQA dataset for training our model. The first step was extracting the sentences and corresponding annotations from the MPQA dataset. The data was then pre-processed to the format required by our model.

5.1. Data pre-processing:

The MPQA dataset required substantial pre-processing as we had to extract out the labels we needed for all the words, phrases and sentences within the dataset. This helped us in designing our model to predict sentiments with respect to the various parts of the sentence like subject and object.

e.g.

Sentence: The Kimberley Provincial Hospital said it would probably know by Tuesday whether one of its patients had Congo Fever.

Pre-processed data:

Element Label	Word	Basic word
subj	The	the
subj	Kimberley	kimberley
subj	Provincial	provincial
subj	Hospital	hospital
...	said	say
subj	it	it
obj	would	would
obj	probably	probably
obj	know	know
...	by	by
...	Tuesday	tuesday
obj	whether	whether
subj	one	one
...	of	of
...	its	its
...	patients	patient
...	had	have
obj	Congo	congo
obj	Fever	fever
punc	.	.

Similarly we processed the dataset to assign subject, object and sentiment labels to each word in each sentence for the dataset.

5.2. Tree representation of the sentence:

As mentioned before, the previous approach utilized a greedy algorithm to find the best possible tree representation for a sentence. However since this does not incorporate the syntactic structure of the sentence in making predictions we choose not to utilize this approach. Instead we parse each sentence and using the Stanford Parser convert each sentence into its parsed form. We then convert this parsed tree to a binary form to determine the hierarchical structure of the sentence. We then use this in our recursive autoencoder model for making predictions.

5.3. Labelling:

Using the above tree structure we calculate the multi label distribution and assign the different labels. Specifically for this problem we assign two kinds of labels to the different elements. The first one is the SOS (subject, object, sentiment classes) label which holds the probability of a particular phrase or word consisting of subject, object and sentiment clause. The second label gives us more information about the sentiment if the element has one. Currently we give it a simple labelling indicating whether the element has a positive, negative or neutral sentiment.

5.4. Training and Results:

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

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

To compute this gradient, we first find reconstruction error at all nodes and then derivatives for these trees are computed efficiently via backpropogation through structure. Because 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 intend to use L-BFGS

run over the complete training data to minimize the objective. It should be noted that the expected objective of the project is not fully met yet. However we believe that given more time, we can tweak the parameters to get good results as the model and algorithm we use might prove to significantly outperform previous techniques used for complex sentiment analysis.

6 Conclusion and Future Work

We tried to use recursive autoencoder techniques to make predictions about the sentence structure as well as the sentiment of the sentence with respect to the subject and object. Training the data using deep learning techniques to make predictions is a hard problem and we believe that given more time we will really be able to make a significant contribution towards solving this problem. Hopefully we should be able to continue with this work and achieve good results as we keep on conducting more experiments. As an example of an additional factor that could be taken into consideration, we could take into account the form of the verb (e.g. active passive voice, tense) while making predictions. The current model may be easily extended to include this due to the use of the Stanford Parser tree output in the model which incorporates the entire syntactic structure of the sentence. Further experiments could make use of these additional factors to make better predictions.

7 References

1. http://nlp.stanford.edu/pubs/SocherLinNgManning_ICML2011.pdf

2. http://www.socher.org/uploads/Main/SocherPenningtonHuangNgManning_EMNLP2011.pdf

3. http://www.socher.org/uploads/Main/2010SocherManningNg.pdf

4. http://www.cs.pitt.edu/mpqa/annotation.html

5. http://cogcomp.cs.illinois.edu/page/resources/data