Convolutional Neural Network for Text Classification

{Anais Addad, Brian Zhen, Gavin Poe, Ling Xie - Github: bzhen/stat198-gicf }

Introduction

Sentiment analysis is one of the key challenges in extracting meaning from online user generated content. In this work, we focus on Amazon reviews for the "cell phones and assessories" category from a dataset of 42.8 million reviews collected by McAuley Leskovec between May 1996 and 2014. The goal is to binary classify each review as positive or negative using a Convolutional Neural Network (CNN).

CNNs use several layers of convolutions over the input layer to compute the output where different filters, automatically learned by CNN, are applied to each layer. The framework consists of four steps: (1) embed each word as a row in a matrix; (2) tune filter sizes over rows; (3) max-pool to produce a feature vector; (4) add softmax classification layer for prediction.

Amazon Dataset

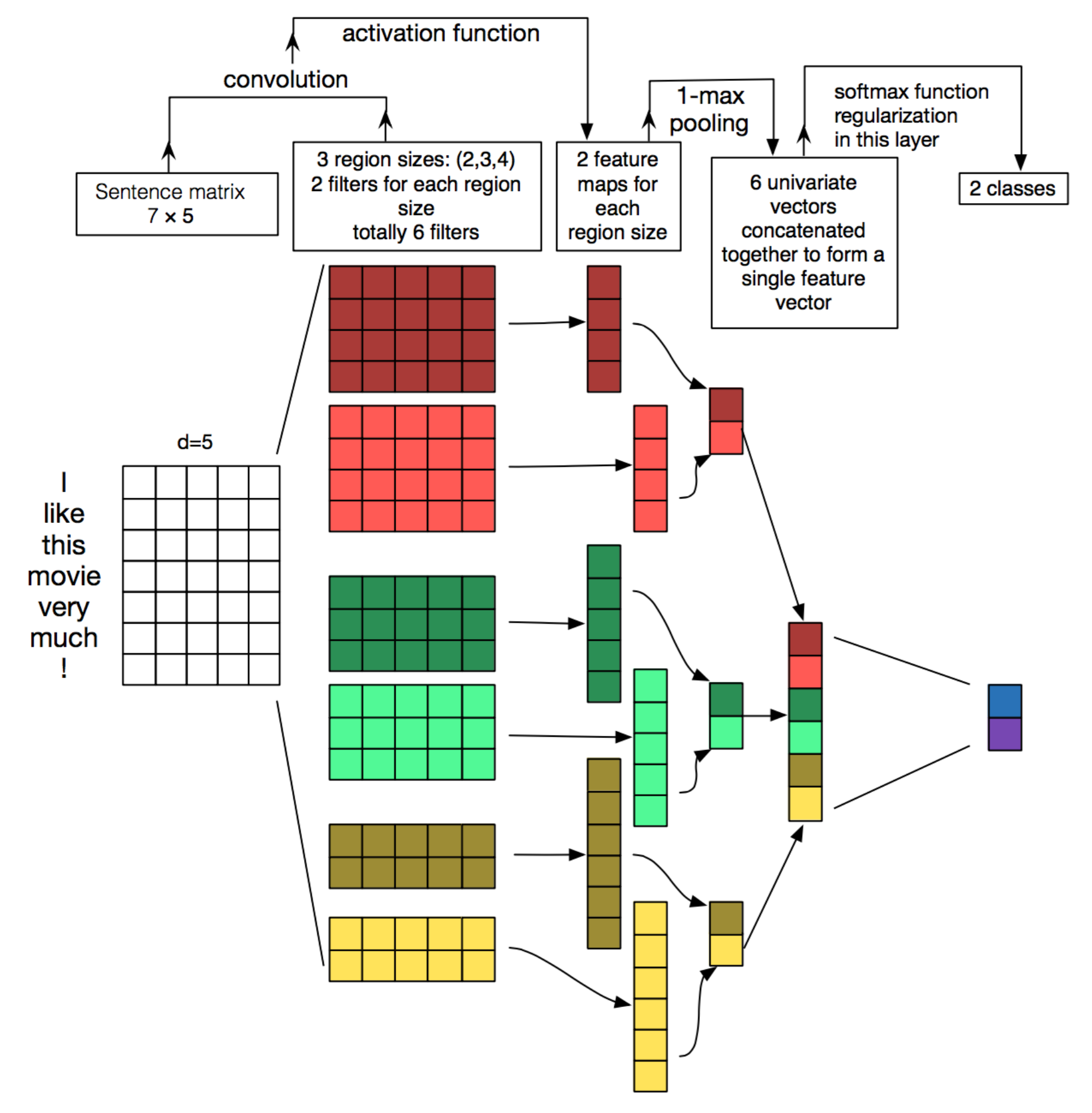
```
{
  "reviewerID": "A2SUAM1J3GNN3B",
  "asin": "0000013714",
  "reviewerName": "J. McDonald",
  "helpful": [2, 3],
  "reviewText": "I bought this for my husband who plays the
piano.  He is having a wonderful time playing these old hymns.
The music  is at times hard to read because we think the book
was published for singing from more than playing from.  Great
purchase though!",
  "overall": 5.0,
  "summary": "Heavenly Highway Hymns",
  "unixReviewTime": 1252800000,
  "reviewTime": "09 13, 2009"
}
```

Model

1. **Data Processing**
   - Clean text data
   - Make each review the same length by appending <PAD> tokens
   - Build vocabulary index V an map each word to integer between 0 and V
2. **Convolutional Neural Networks (CNN)**
   - Word Embeddings: each row of matrix represents a word
   - Filter of different sizes slide over full rows of matrix
   - Every filter performs a convolution on sentence matrix and generates feature maps

Model (cont.)

- Pooling of maps to reduce output dimensionality
- Final softmax layer receives feature vector as input used for binary classification


activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 × 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I like this movie very much !

3. **TensorFlow training procedure**
   - Minimize our loss function using Adam optimizer
   - Keep track of our accuracy and loss
   - Iterate over batches of our data by calling the `train_step` function
   - Run training procedure with default parameters: 128-dimensional embedding, filter sizes of 1,2,3 and 4, dropout of 0.5 ad 128 filter per sizes
   - Train/Test Split: 9000/1000
4. **Training and Test Data**
   - *Train Data*: we assumed that reviews with a rating of 1 are negative and those with a rating of 5 are positive; we adjusted our samples to have the same amount of positive and negative reviews
   - *Test Data*: we tested both on a manually labelled (0/1) dataset of sentences from Amazon, IMDB and Yelp and we also used a data set similar to that of our training set

Contact Information

Research supervised by Professor Deborah Nolan, UC Berkeley - Department of Statistics

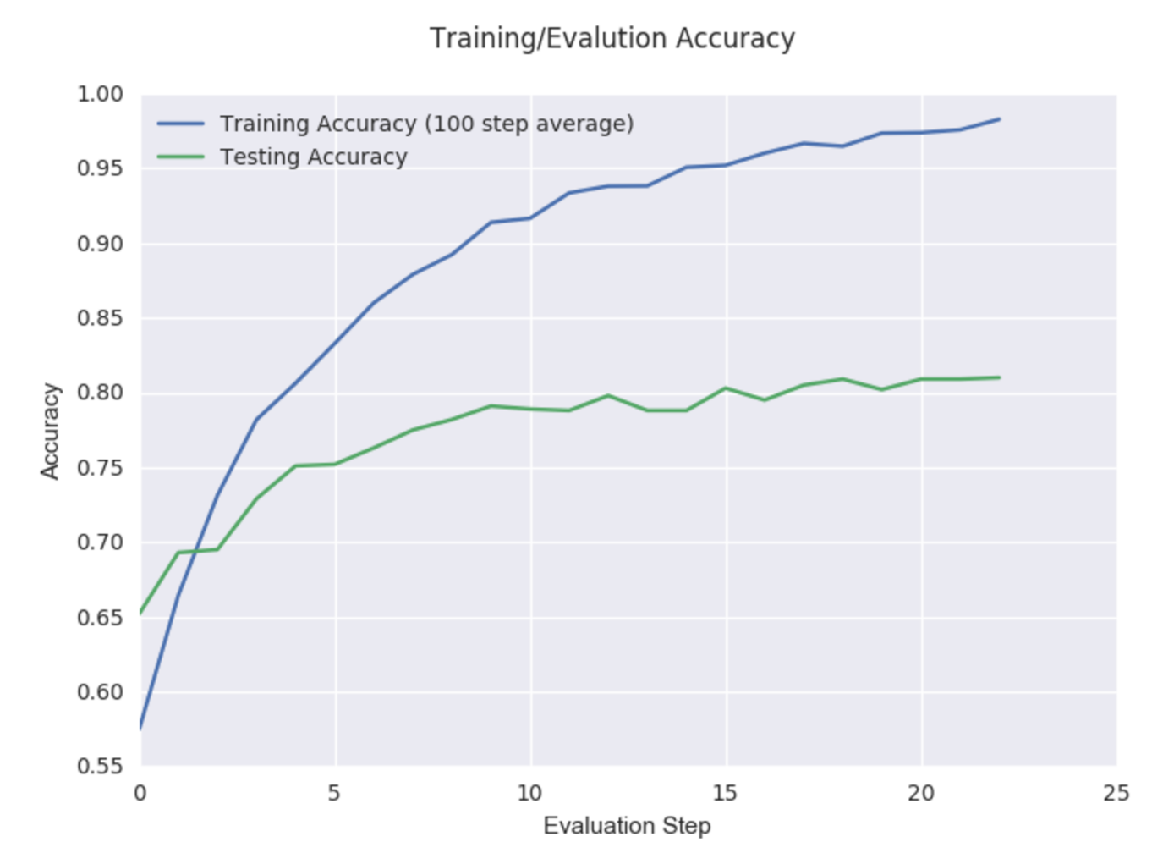
Anais Addad: B.A. Economics 2017, email: anaisaddad@berkeley.edu
Brian Zhen: B.A. Computer Science 2018, email: bzhen@berkeley.edu
Gavin Poe: B.A. Computer Science 2019, email: gpoe@berkeley.edu
Ling Xie: B.A. Statistics 2019, email: xieling@berkeley.edu

Results 1

Key Parameters

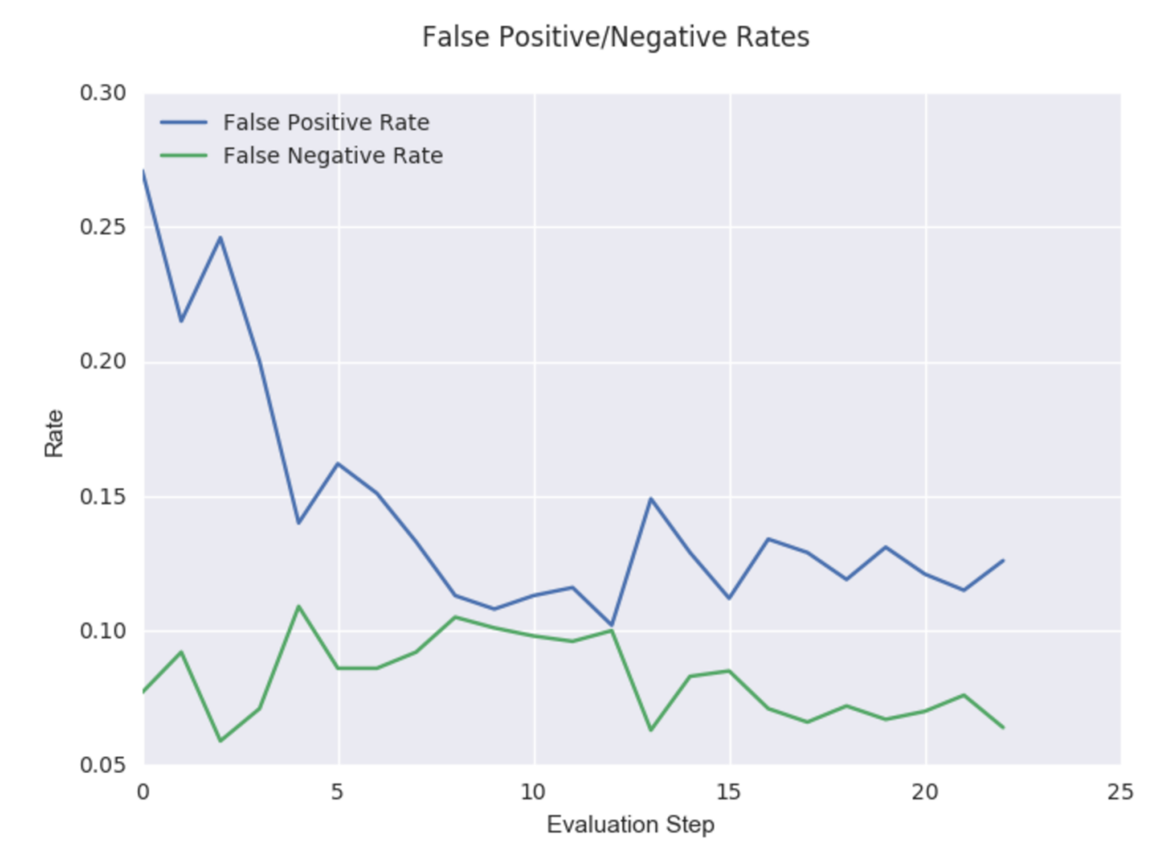
BATCH_SIZE	64
DEV_SAMPLE_PERCENTAGE	0.1
DROPOUT_KEEP_PROB	0.5
EMBEDDING_DIM	128
FILTER_SIZES	3,4,5
NUM_EPOCHS	44
NUM_FILTERS	128
Vocabulary Size	10084
Train/Dev Split	9000/1000

batch_size: number of samples propagated through network
dev_sample_percentage: fraction of data used for evaluation
dropout_keep_prob: fraction of neurons to keep enabled
filter_sizes: number of words covered by filter
num_epochs: number of passes through data
num_filters: number of filters per filter_size



These are the parameters that we used to train our model. This graph shows how our smoothed overall accuracy improved over time for the train and test dataset. We can see that our train data reaches an accuracy of about 95% while our test data reaches a maximum accuracy of about 80%.

Results 2 and Random Forest Baseline



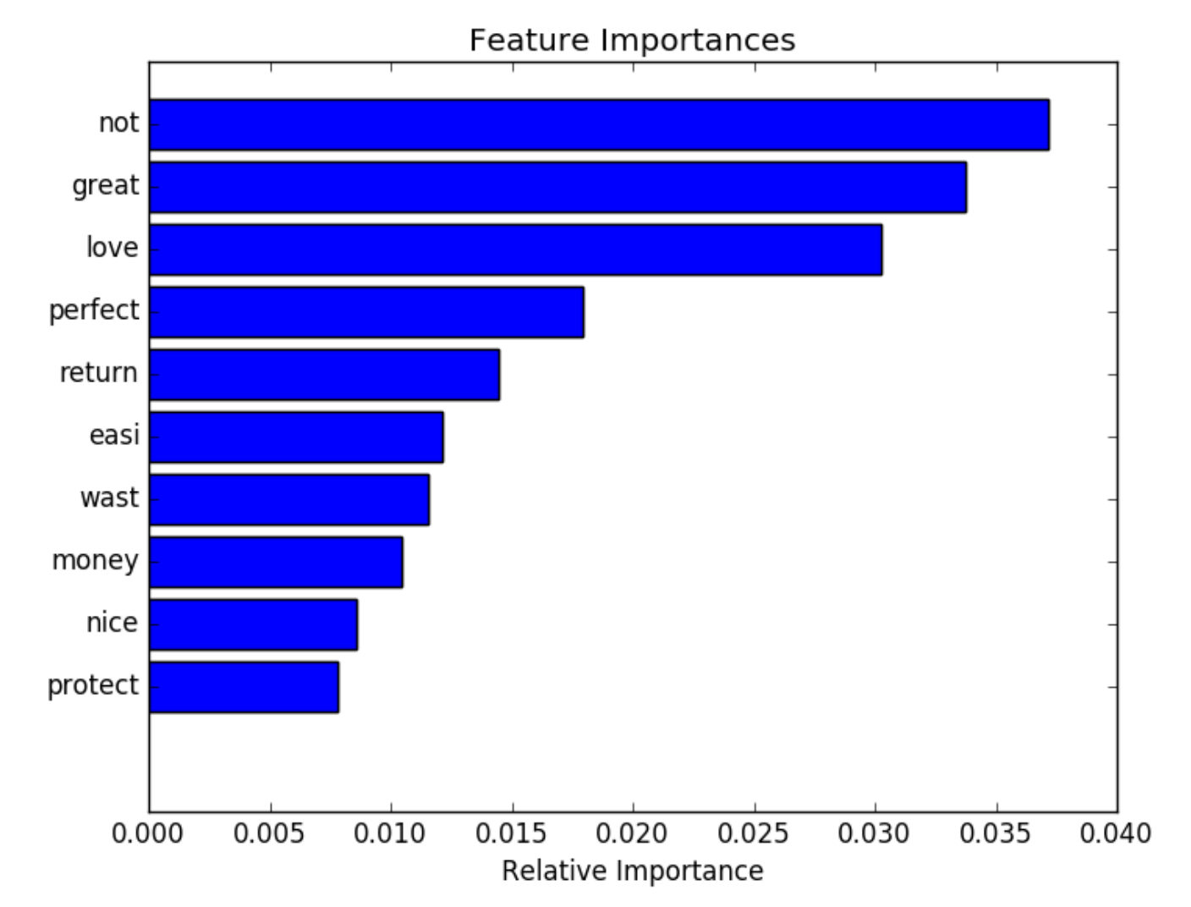
The false positive rate improves significant, whereas the false negative rate shows relatively little improvement. After step 12, both the false positive and false negative rates converge to around 12% and 6%, respectively. These results reveal that the model predicts negative reviews more accurately than positive reviews.

False positive (negative, but predicted positive): "it was an inexpensive piece , but i would still have expected better quality"

False negative (positive, but predicted negative): "it quit working after i 'd used it for about 18 months , so i just purchased another one because this is the best headset i 've ever owned"

Second Prediction Method: Random Forest

This graph shows the ten most "important" features and their corresponding importance. Our overall accuracy for the random forest method is about 84% using the manually labeled data for our test set. The random forest tree we initialized contains 100 trees, and the bag of words contains the 5000 most frequent words.



Conclusion

After testing two different machine learning techniques: deep neural networks implemented through TensorFlow and the random forest, we conclude that the random forest method is overall more accurate by 3% in extrapolating our model to predict sentiment of different sentences from different sources than the TensorFlow model (81%). We were inspired to pursue convolutional neural networks methods for text classification after reading the paper "From Group to Individual Labels using Deep Features" from D. Kotzias, M. Denil, N. De Freitas and P. Smyth who achieved a high accuracy of 88% in their results by implementing CNN and minimizing a group-instance cost function. However, we have come to the conclusion that the random forest method is preferred in predicting binary sentiment analysis of Amazon reviews through supervised learning.