Sentiment Analysis of Drug Reviews

Rajesh Khanna - 16CS10014
Sai Krishna Reddy - 16CS30018
S. Sasi Bhushan - 16CS30032
Uppada Vishnu - 16CS30037

Introduction

- Sentiment analysis is a type of subjectivity analysis which analyzes sentiment in the given text with the objective of understanding the sentiment polarities (i.e.positive, negative, or neutral) of the opinions regarding various aspects of a subject (Drug reviews).
- It is still considered as avery challenging problem since user generated content is described in various and complex ways using natural language

Motivation

- For sentiment analysis, most researchers have worked on general domains (such as electronic products, movies, and restaurants), but not extensively on health and medical domains.

- It has been shown that performing sentiment analysis on drug reviews is useful in many ways like:
  - When a new drug is released its review will be useful not only for patients but also for drug makers to get valuable feedback about the drug.
  - Highlight patient’s misconceptions and dissenting opinions about a drug.

Task Overview

- There is a huge corpus (Train-161297, Test-53766) containing the reviews of drugs given by the customers.
- Each review is rated on a scale of 1-10. The reviews rated 1-4 are negative, 5-6 are neutral and 7-10 are positive.
- In this task, we explore classifying each review as positive, negative or neutral.

Label	Review count
1	21619
2	6931
3	6513
4	5012
5	8013
6	6343
7	9456
8	18890
9	27531
10	50989

Model Implementations

- We have implemented 3 models:
  - TextCNN
  - RCNN
  - Seq2seq with attention.

- Each model is implemented using 3 embeddings:
  - GloVe
  - Word2vec
  - ELMo.

Model Architecture

TextCNN Model:

- Convolutional Neural Networks (CNN) are used for image classification or recognition etc., that use convolution in place of general matrix multiplication in at least one of their layers.

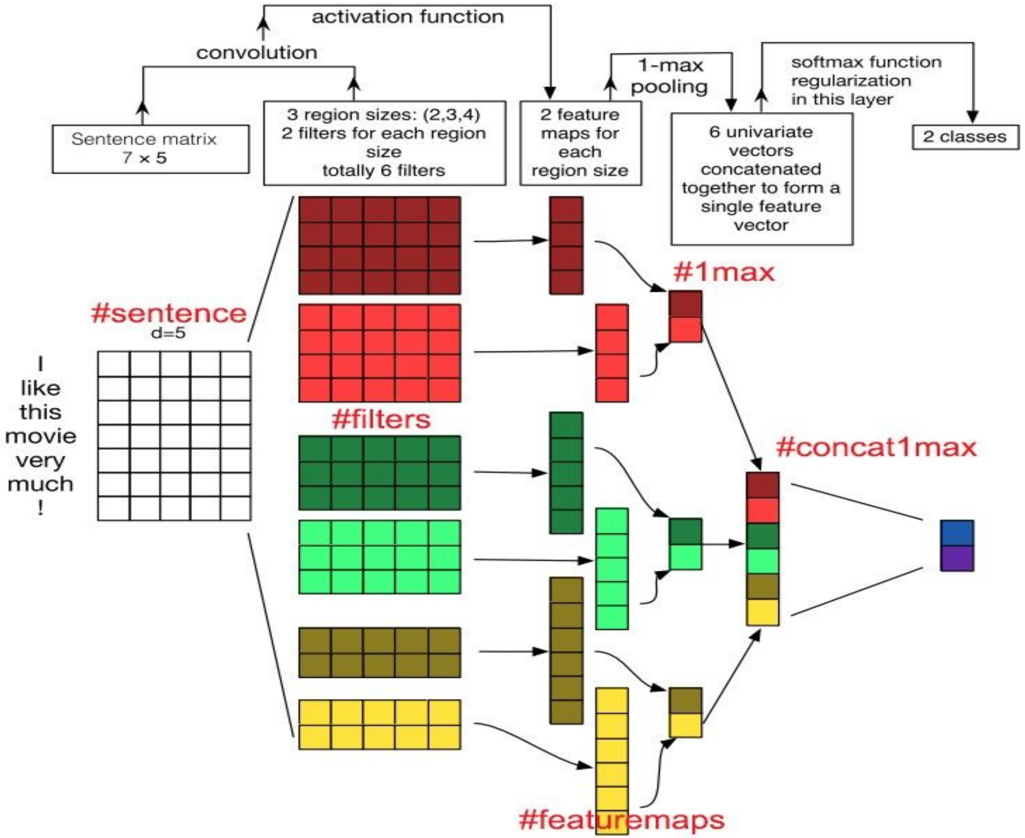
- TextCNNs are inspired from CNNs that is used for document classification (sentiment analysis) in which the document (review) is seen as an image i.e., just as we use CNN for images, we use textCNN for documents or reviews.

- The words in each review are converted to vectors using the embeddings and this review now becomes a 2d matrix (number of words * vector size) where as an image is a 2d matrix of pixel values.

- Instead of image pixels, the input to the tasks is sentences or documents (reviews) represented as a matrix. Each row of the matrix corresponds to one-word vector.

Model Architecture

A Sample textCNN that classifies the sentence ‘I like this movie very much!’.


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
#sentence
d=5
I
like
this
movie
very
much
!
#filters
#1max
#concat1max
#featuremaps

Model Architecture

RCNN Model:

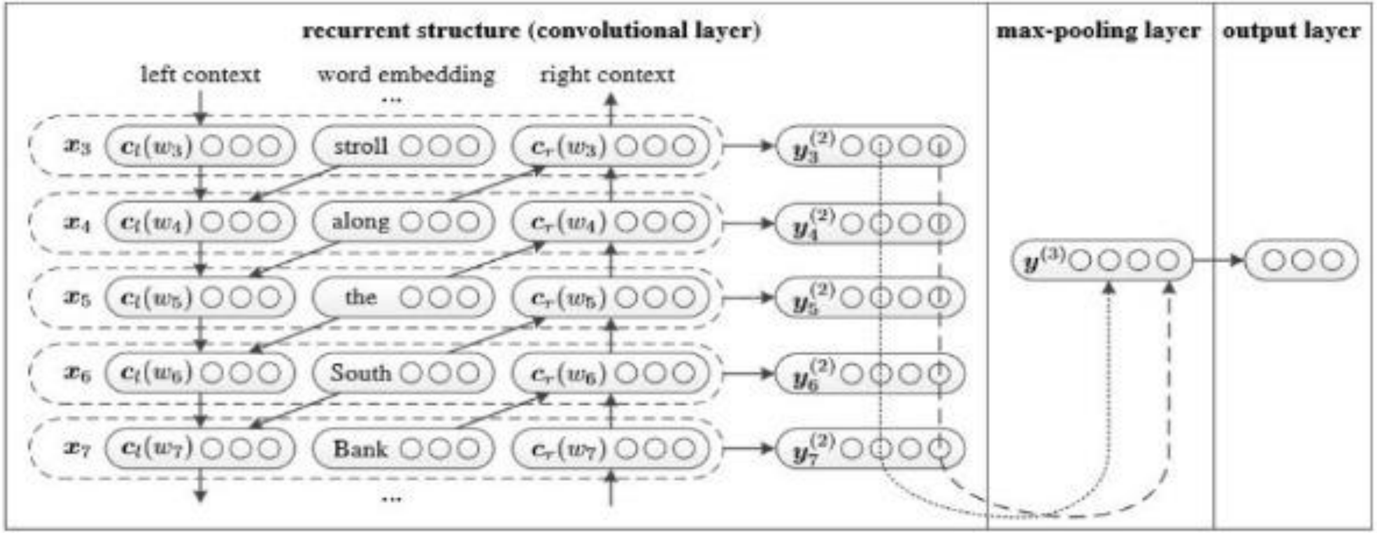
- RCNN (Recurrent Convolutional Neural Network) is used for text classification (Sentiment Analysis) with captures contextual information with the recurrent structure and constructs the representation of text using a convolutional neural network.
- Structure:1)recurrent structure (convolutional layer) 2)max pooling 3) fully connected layer+softmax.

- It learns representation of each word in the sentence or review with left side context and right side context.

- Representation current_word = [left_side_context_vector, current_word_embedding, right_side_context_vector].

- For left side context, it uses a recurrent structure, a non-linearity transform of previous word and left side previous context; similarly for the right side context.

Model Architecture


recurrent structure (convolutional layer)
max-pooling layer
output layer
left context
word embedding
right context
x_3 $c_l(w_3)$ stroll $c_r(w_3)$ $y_3^{(2)}$
x_4 $c_l(w_4)$ along $c_r(w_4)$ $y_4^{(2)}$
x_5 $c_l(w_5)$ the $c_r(w_5)$ $y_5^{(2)}$
x_6 $c_l(w_6)$ South $c_r(w_6)$ $y_6^{(2)}$
x_7 $c_l(w_7)$ Bank $c_r(w_7)$ $y_7^{(2)}$
$y^{(3)}$
...

The structure of the recurrent convolutional neural network. This figure is a partial example of the sentence “A sunset stroll along the South Bank affords an array of stunning vantage points”, and the subscript denotes the position of the corresponding word in the original sentence.

Model Architecture

Seq2seq with attention:

This model belongs to a family of encoder–decoders in which we encode a source sentence or a review into a fixed-length vector which is passed to the decoder which generates the required output.

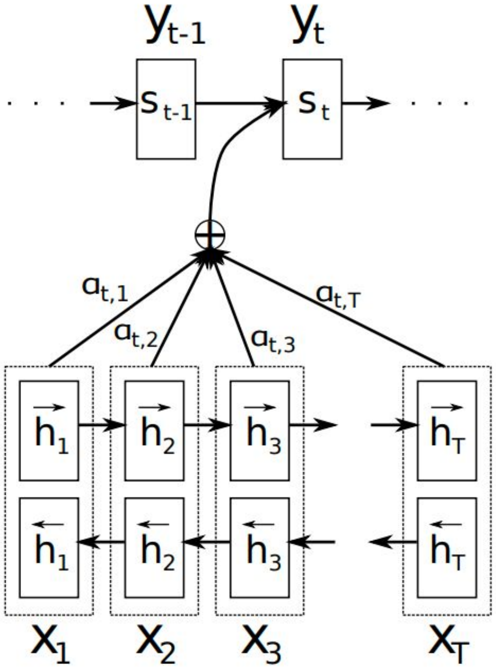
Since the sentence is very long some information about the sentence may be lost in the encoded in the small encoder vector. This bottle neck is solved with attention which directly transfers the information from the encoder words to the decoder.

Structure: 1)embedding 2)bi-GRU to get rich representation from source sentences(forward & backward). 3)decoder with attention.

Input: embedded words sequence of length 100
Output: One word embedding which is either positive, negative or neutral.

Model Architecture

The Seq2seq model trying to generate the prediction y_t from the decoder given a source review $(x_1, x_2, . . . , x_T)$



Word Embeddings

GloVe:

- This model is an Unsupervised algorithm for obtaining vector representations for words. This is achieved by mapping words into a meaningful space where the distance between words is related to semantic similarity.

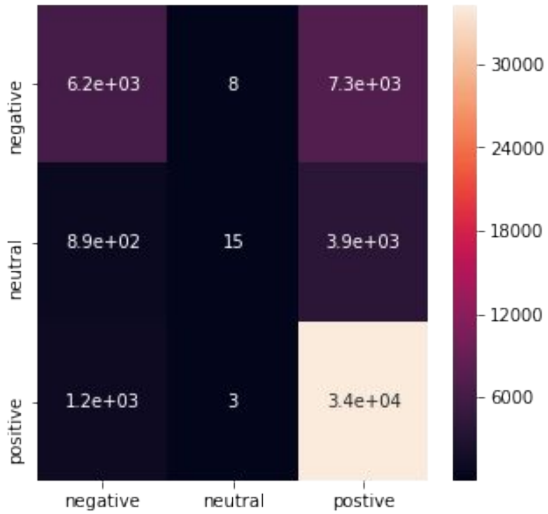
Word2vec:

- Word2vec model takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space.

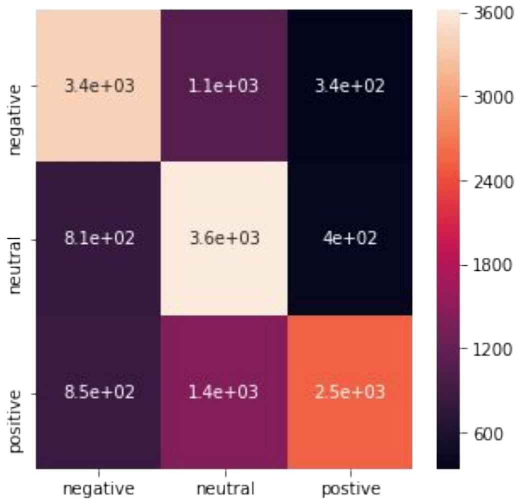
ELMo:

- ELMo is a deep contextualized word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts. These word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus.

Why Balance Data ?



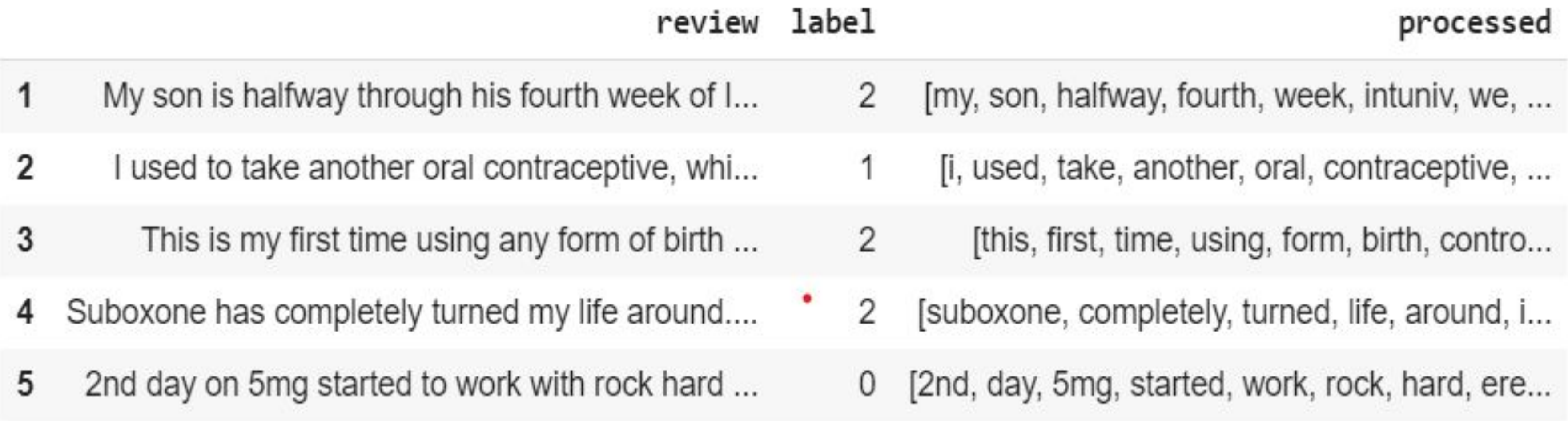
Unbalanced Data



Balanced Data

The Data is balanced so that we have equal data in all classes and works better during testing.

Pre-Processing



	review	label	processed
1	My son is halfway through his fourth week of I...	2	[my, son, halfway, fourth, week, intuniv, we, ...
2	I used to take another oral contraceptive, whi...	1	[i, used, take, another, oral, contraceptive, ...
3	This is my first time using any form of birth ...	2	[this, first, time, using, form, birth, contro...
4	Suboxone has completely turned my life around....	2	[suboxone, completely, turned, life, around, i...
5	2nd day on 5mg started to work with rock hard ...	0	[2nd, day, 5mg, started, work, rock, hard, ere...

Results

Summary of Results

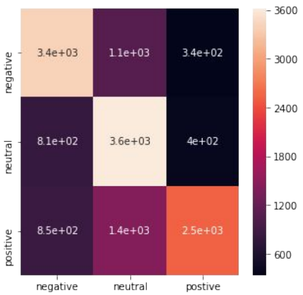
Model	Embeddings	Accuracy
TextCNN	GloVe	66%
	Word2vec	65%
	ELMo	63%
RCNN	GloVe	63%
	Word2vec	64%
	ELMo	58%
Seq2seq	GloVe	53%
	Word2vec	59%
	ELMo	57%

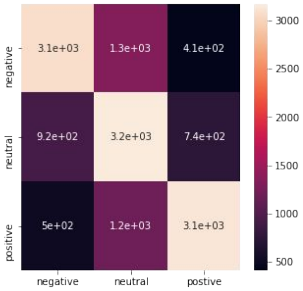
Glove

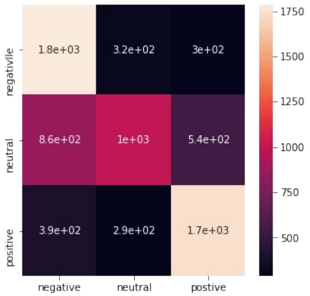
Word2Vec

Elmo

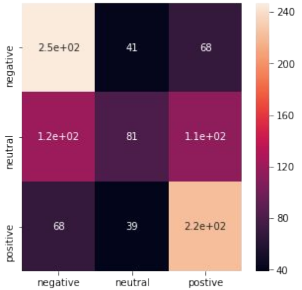
TextCNN

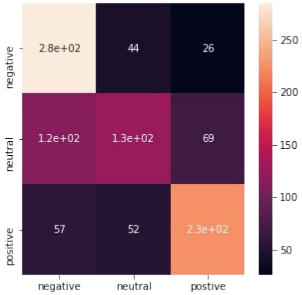


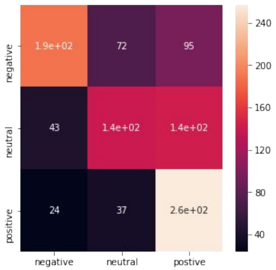




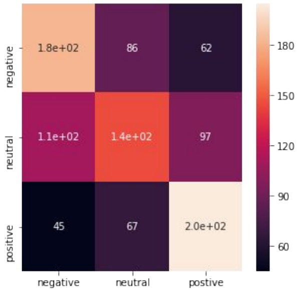
RCNN

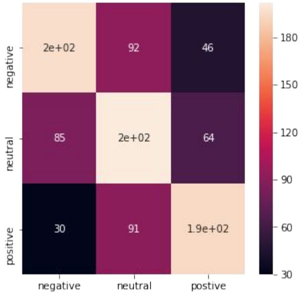


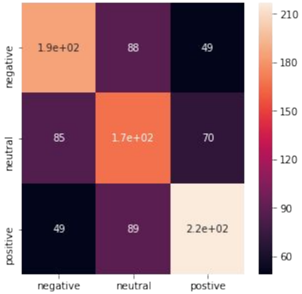




Seq2Seq
Attention







Results

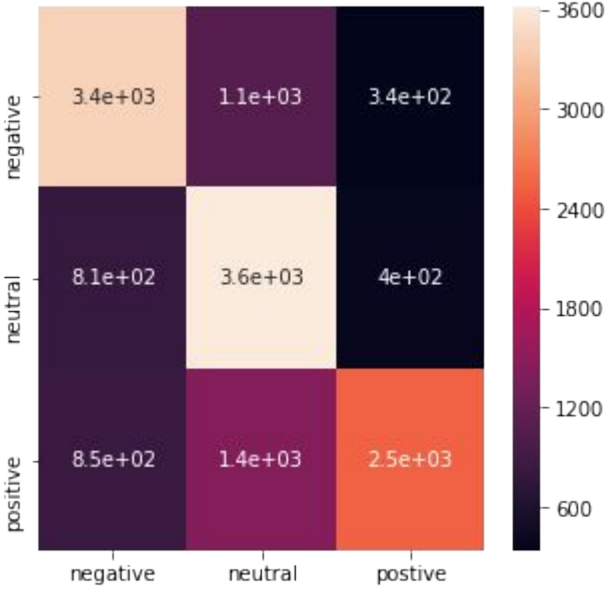
TextCNN with Glove Embeddings:

We took Glove-50 (50 dim vectors) for the embeddings.

class	precision	recall	f1-score	support
0	0.67	0.71	0.69	4829
1	0.59	0.75	0.66	4829
2	0.78	0.53	0.63	4829

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 66%.



Heatmap of the Results

Results

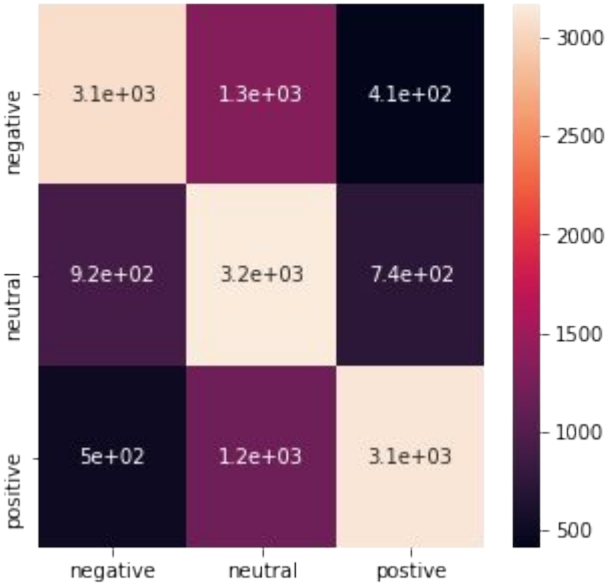
TextCNN with Word2Vec Embeddings:

We took 300 dim vectors for the embeddings.

class	precision	recall	f1-score	support
0	0.68	0.64	0.66	4829
1	0.56	0.66	0.60	4829
2	0.73	0.65	0.69	4829

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 65%.



Heatmap of the Results

Results

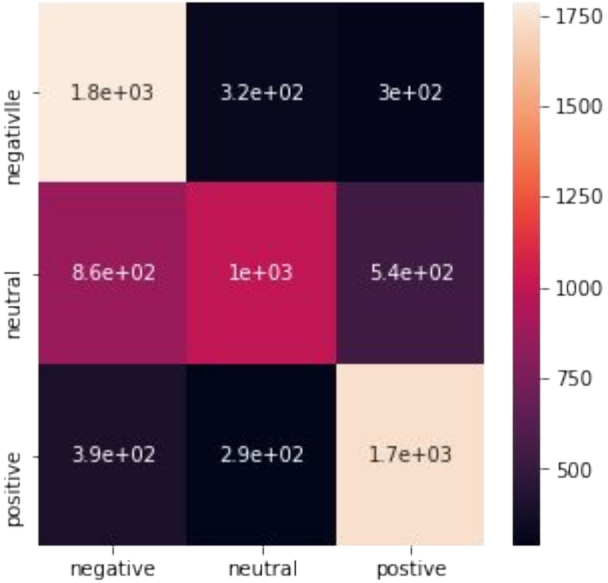
TextCNN with ELMo Embeddings:

We took 1024 dim vectors for the embeddings.

class	precision	recall	f1-score	support
0	0.59	0.74	0.65	2413
1	0.62	0.42	0.50	2405
2	0.67	0.72	0.69	2425

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 63%.



Heatmap of the Results

Results

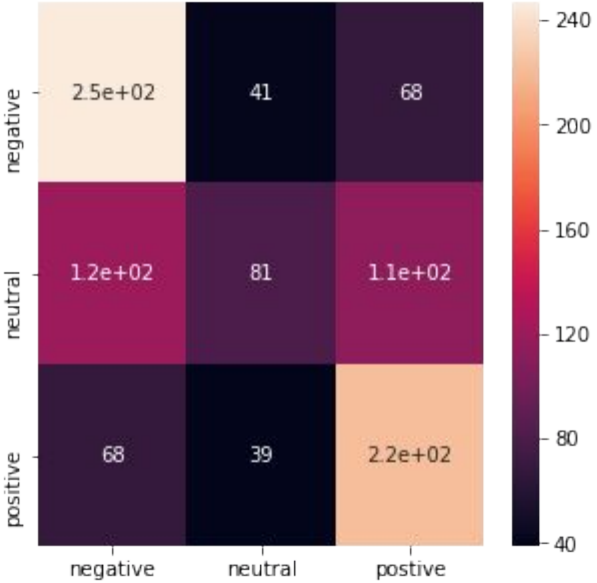
RCNN with Glove Embeddings:

We took Glove-50 (50 dim vectors) for the embeddings.

class	precision	recall	f1-score	support
0	0.57	0.69	0.62	356
1	0.50	0.26	0.34	315
2	0.55	0.67	0.61	329

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 55%.



Heatmap of the Results

Results

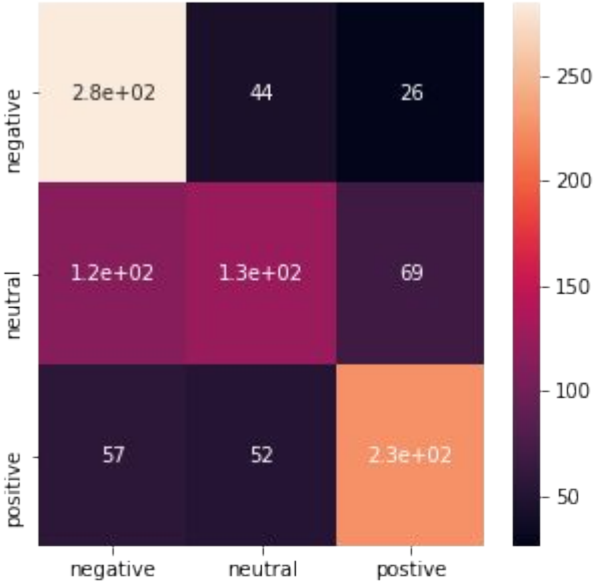
RCNN with Word2Vec Embeddings:

We took 300 dim vectors for the embeddings.

class	precision	recall	f1-score	support
0	0.62	0.80	0.70	355
1	0.57	0.41	0.47	310
2	0.70	0.67	0.69	335

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 64%.



Heatmap of the Results

Results

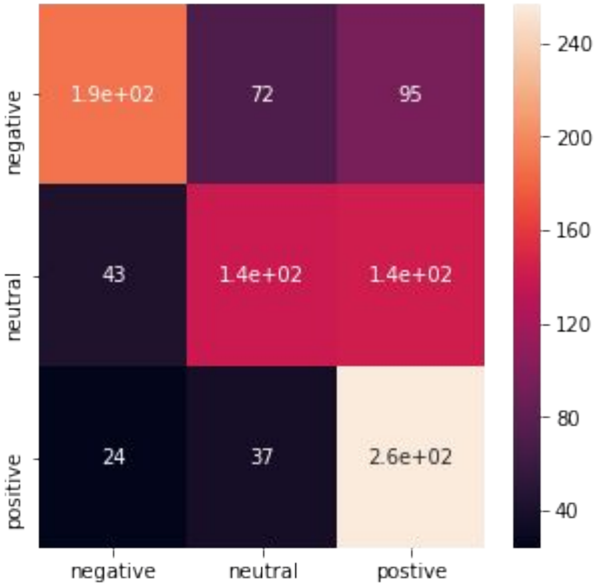
RCNN with ELMo Embeddings:

We took 1024 dim vectors for the embeddings.

class	precision	recall	f1-score	support
0	0.74	0.53	0.62	355
1	0.56	0.43	0.49	327
2	0.52	0.81	0.63	318

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 58%.



Heatmap of the Results

Results

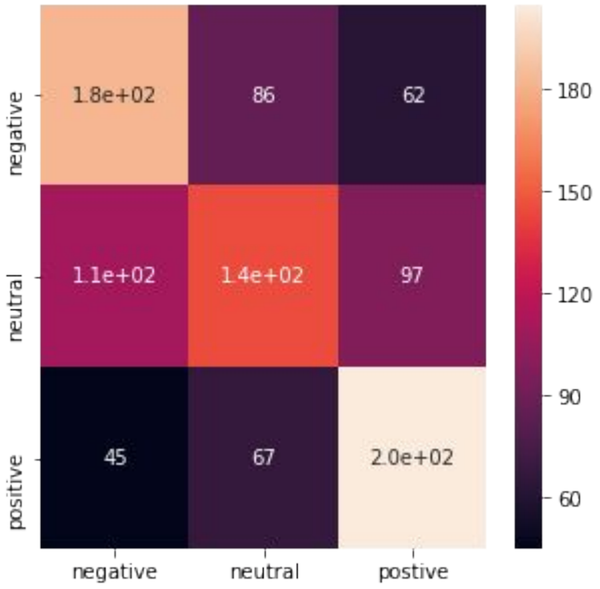
Seq2seq with Glove Embeddings:

We took Glove-50 (50 dim vectors) for the embeddings.

class	precision	recall	f1-score	support
0	0.54	0.65	0.55	331
1	0.49	0.41	0.49	352
2	0.56	0.65	0.60	317

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 53%.



Heatmap of the Results

Results

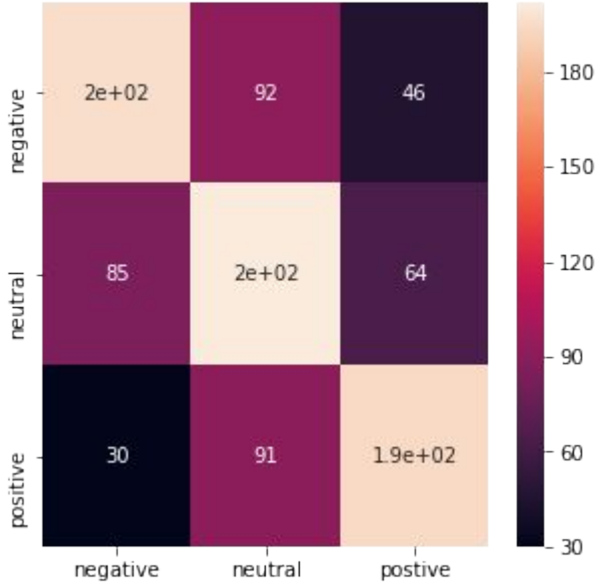
Seq2seq with Word2Vec Embeddings:

We took 300 dim vectors for the embeddings.

class	precision	recall	f1-score	support
0	0.63	0.59	0.61	334
1	0.52	0.58	0.55	351
2	0.64	0.62	0.63	315

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 59%.



Heatmap of the Results

Results

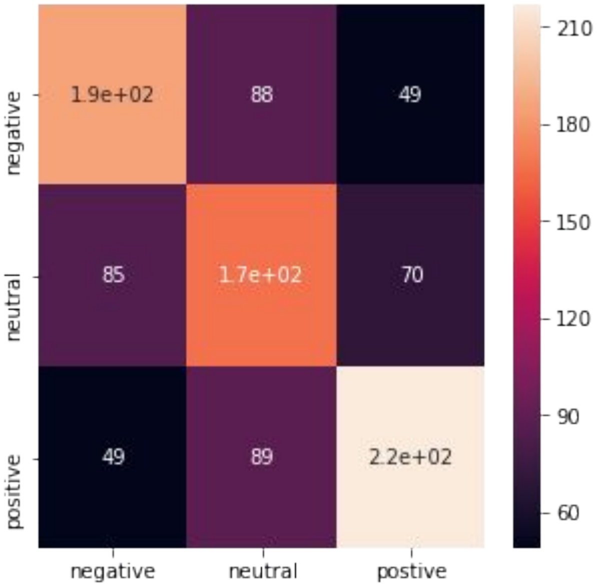
Seq2seq with ELMo Embeddings:

We took 1024 dim vectors for the embeddings.

class	precision	recall	f1-score	support
0	0.58	0.58	0.58	323
1	0.49	0.52	0.50	322
2	0.65	0.61	0.63	344

The Classes are 0 (negative), 1 (neutral) and 2 (positive).

The accuracy of this model is 57%.



Heatmap of the Results

Ablation Analysis

- Words that were given high attention by the Attention model with corresponding counts of sentences in which they occurred.

Negative		Neutral		Positive	
bleeding	98	feel	12	love	374
pain	80	started	8	life	367
my	65	bleeding	6	my	211
feel	61	like	6	works	209
bad	56	eat	39	great	182
horrible	47	started	34	love	89
		UNK	34	years	87
		bad	34	great	69

Ablation Analysis

- We observed high accuracy for unbalanced than balanced data due to data being skewed, with only 9 % of whole labels in class 1.
- Training time model were in the order TextCNN < RCNN <<< Seq2Seq with attention.
- The order of training time with respect to embeddings was observed to be Glove(50) < Word2Vec(300) < Elmo(1024). This can be attributed to the increasing length of vectors.
- We observe that F1- score of RCNN is higher than TextCNN in negative and positive sentiment predictions. This can be attributed to the sequential structure of RCNN which is not present in TextCNN.

Thank You