SENTIMENT ANALYSIS AMAZON BOOK REVIEWS

1 Group members

Kingshuk Basak | Roll-1611CS06 | Email-kingshuk.mtcs16@iitp.ac.in
Garima Sahu | Roll-1721EE04 | Email-garima.pee17@iitp.ac.in
Kingshuk Basak
Garima Sahu

2 Abstract of the project:

Sentiment has traditionally been considered a "deep" attribute of writing, often required the interpretation of figurative language to uncover the writer's intention. Sentiment Analysis is one of the main challenges in natural language processing. Recently, deep learning applications have shown impressive results across different NLP tasks. In this project we did Sentiment Analysis on Amazon Book Reviews. We will be using Deep Neural Feed Forward Network for this Sentiment Analysis. Sentiment Analysis can be of three types:

1. Positive
2. Negative

This analysis can be done using neural network.

Using this method we calculated the positive and negative sentiment. Then we compared it with the actual result to find the accuracy.

3 Introduction:

Sentiment mainly refers to feelings, emotions, opinion or attitude. With the rapid increase of usage of internet people frequently express their sentiment through social media, ratings and reviews. Due to this increase in the textual data, there is a need to sentiment classification and calculate insights for exploring business. Deep learning is very helpful in classifying and predicting whether a particular review is positive or negative. This process can be divided into two types: Supervised and Un-Supervised. We have used Supervised learning algorithm where each line of labeled data set is labeled with appropriate sentiment. We have classified positive sentiment as 1 and negative sentiment as 0 in this project. We have used Doc2Vec in this project. Doc2vec is an extension of word2vec that learns to correlate labels and words, rather than words with other words. The main purpose of Doc2Vec is associating arbitrary documents with labels, so labels are required.

4 Literature survey:

List of papers read:

1. Transfer Learning for Cross-Lingual Sentiment Classification with Weakly Shared Deep Neural Networks. By: Guangyou Zhou, Zhao Zeng, Jimmy Xiangji Huang, and Tingting He.
2. Neural Network Based Context Sentiment Analysis. By: S.Suruthi, M.Pradeeba, A.Sumaiya.
3. Convolution Neural Networks for Sentiment Classification. By: Yoon Kim.

4. Neural Networks for Sentiment Analysis. By: Brett Duncan and Yanqing Zhang.
5. An Approach to Sentiment Analysis using Artificial Neural Network with Comparative Analysis of Different Techniques. By: Pranali Borele, DIlipkumar A. Borikar
6. Convolutional Neural networks for Sentiment Classification. By: Yoon Kun.
7. Neural Network for Sentiment Analysis on Twitter. By: Brett Duncan and Yanqing Zhang.

4.1 Transfer Learning for Cross-Lingual Sentiment Classification with Weakly Shared Deep Neural Networks

In this paper they have used Weakly Shared Deep Neural Network for Sentiment Analysis on Amazon products.Data set was in four languages (English (E), French(F), German (G) and Japanese (J)) of three categories (Books(B), DVD (D), Music (M)).
They have compared their result with different methods.

1. TARGET BAG OF WORDS.
2. CROSS-LINGUAL LATENT SENTIMENT ANALYSIS.
3. CROSS-LINGUAL SRUCTURAL CORRESPONDENCE LEARNING.
4. MACHINE TRANSLATION
5. MIXTURE MODEL.
6. ORIENTED PRINCIPAL COMPONENT ANALYSIS.
7. TWO STEP METHOD
8. STATE OF ART MULTI-VIEW LEARNING METHOD.

Average accuracy: 80.10

4.2 Neural Network Based Context Sentiment Analysis

In this paper they have taken data from "Recognizing Stances in Idological On-line Debates". Process used:

1. Data processing.
2. Feature extraction.
3. Competitive layer Neural Network.

Average accuracy: 82.15

4.3 An Approach to Sentiment Analysis using Artificial Neural Network with Comparative Analysis of Different Techniques

In this paper they have introduced the methods for Sentiment Analysis on movie reviews.

1. Machine learning method.
2. Naive Bias.
3. Support Vector Machine.
4. K-Nearest Neighbor.
5. Maximum Entropy.

6. Artificial Neural Network.
7. Lexicon-based approach.
8. Rule based approach.

Methods done in the paper:

1. Pre-processing.
   (a) Stop word removal.
   (b) Symbol removal.
   (c) POS tagging.
2. Feature extraction.
3. Training and Classification.

4.4 Convolutional Neural networks for Sentiment Classification.

In this paper they have used Convolutional Neural Network to implement Sentiment Analysis on different data. Data sets:

1. MOVIE REVIEW.
2. STANFORD SENTIMENT TREE BANK.
3. SUBJECTIVE DATA SET.
4. TREC QUESTION SET.
5. CUSTOMER REVIEW.
6. MPQA DATA SET.

Then they have converted word to vector. Before doing that they have used regularization on the data. Then they implemented some method:

1. CNN-rand.
2. CNN-static.
3. CNN-non-static.
4. CNN-multichannel.

Average accuracy: 70.92-95.0

4.5 Neural Network for Sentiment Analysis on Twitter.

In this paper they have used Twitter data as data set. Methods done:

1. Pre-processing.
   (a) Removal of Punctuation and symbols.
   (b) Removal of @ words.
   (c) Removal of single character.
   (d) Removal of stop words.
2. Stemming.

3. Creating Vocabulary list.
4. Loading Training Labels.
5. Map Variable.
6. Numerical Training Vector.
7. Training the Neural Network.
8. Collecting Test Tweets.
9. Numerical Test Vector.

Average Accuracy: 74.15

5 Data sources:

We are taking data from amazon book reviews for the list of books:

Gone Girl, The Girl on the Train, The Fault in our Stars, Fifty Shades of Grey, Unbroken, The hunger games, The Goldfinch, The Martian.

Link for data: http://archive.ics.uci.edu/ml/datasets/Amazon+book+reviews

Data Set Information:

Gone Girl: 41.974
The Girl on the Train: 37.139
The Fault in our Stars: 35.844
Fifty Shades of Grey: 32.977
Unbroken: 25.876
The hunger games: 24.027
The Goldfinch: 22.861
The Martian: 22.571

Abstract: 213.335 book reviews for 8 different books. There are books which are scored very negatively in general and books which are scored very positively.

Data Set Characteristics:	Multivariate, Text	**Number of Instances:**	213335	**Area:**	Computer
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	4	**Date Donated**	2016-11-16
Associated Tasks:	Classification, Clustering	**Missing Values?**	N/A	**Number of Web Hits:**	27710

We collected data set from "http://jmcauley.ucsd.edu/data/amazon/" on book reviews as json file to create Doc2Vec.

6 Current status:

1. Data has been pre-processed.
2. Data Set is divided into 4 parts.
   (a) train

(b) train-target

(c) test

(d) test-target

3. Implementation using Feed Forward Neural Network using two Hidden Layer.
4. Used library in Python: NUMPY,GENSIM,DOC2VEC,TESORFLOW, MATPLOTLIB, HTML-PARSER, ARGPARSE.
5. Sentences converted into vector using Doc2vec.
6. We are saving the Doc2vec model in amazon.d2v.
7. We have implemented using Feed Forward Neural Network having two hidden layer.
8. Input to the Neural network is in the form of Vector of size 200.
9. Used Tanh in first hidden layer as activation function.
10. Used Relu in second hidden layer as activation function.
11. For regularization of data we used dropout.
12. We used Sparse Softmax Cross Entropy with Logits as loss function.
13. We used RMSProp as optimizer with learning rate 0.02.
14. We have used batch size=100.
15. We are running for various epoch values: 45, 100, 250, 500.
16. We have plotted graph Epoch vs Accuracy.
17. We have plotted graph Epoch vs Loss.
18. Link to the code in GitHub: https://github.com/KingGarima/SentimentAnalysisAmazon

7 Future Work:

We will implement using CNN for better accuracy.

8 Results:

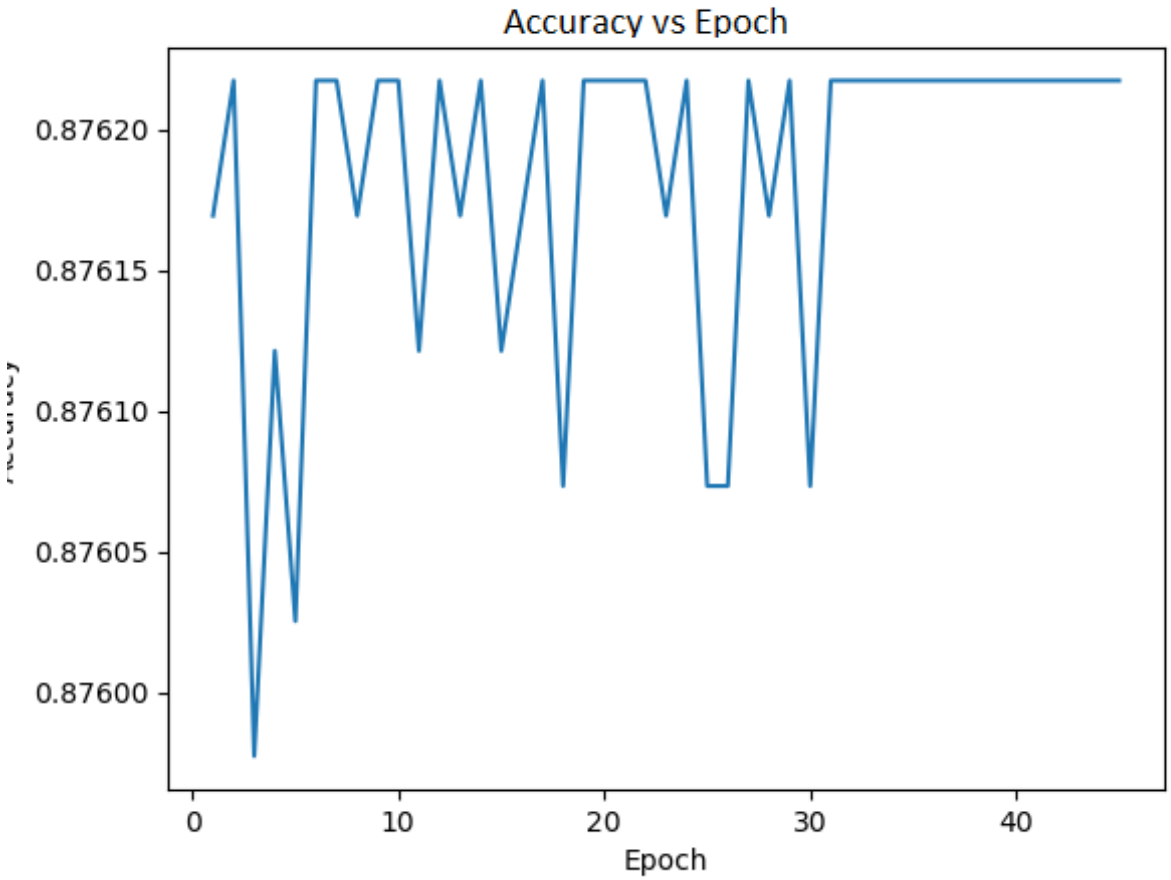

Figure 1: Epoch 45

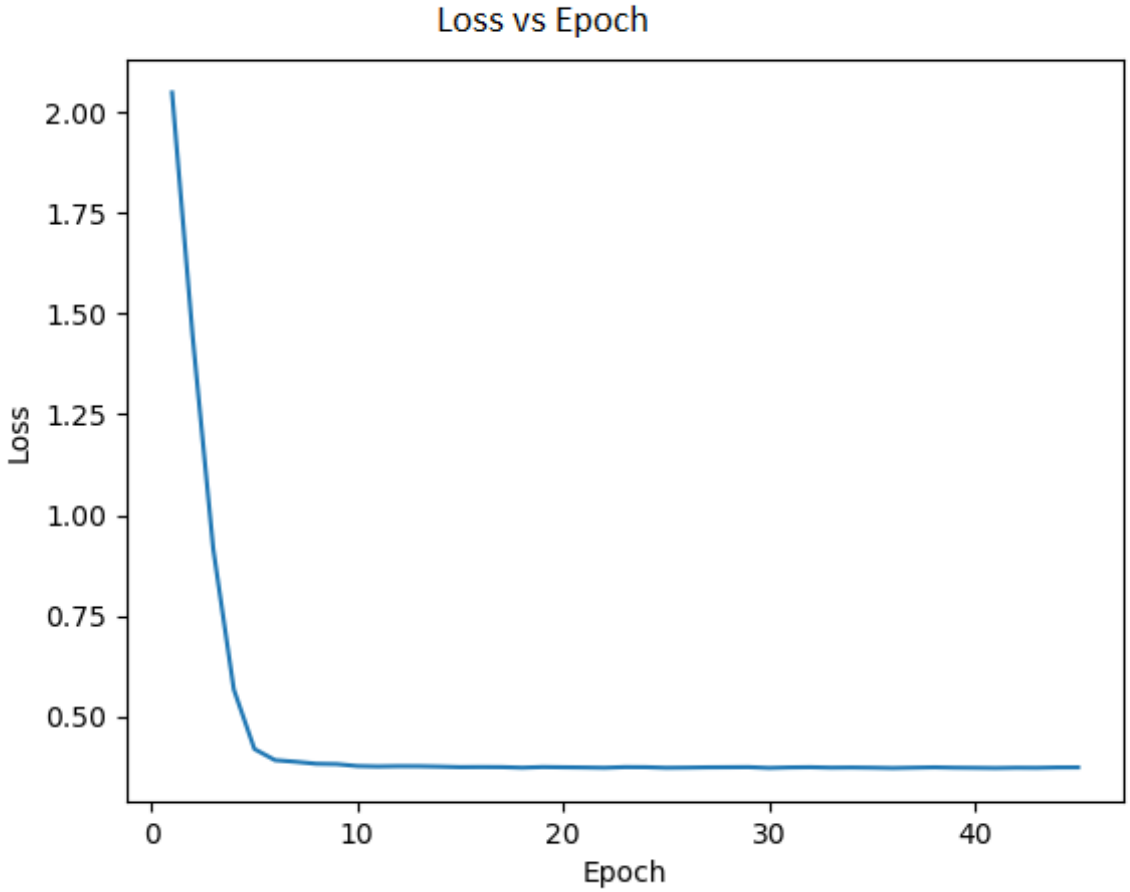

Figure 2: Epoch 45

```
('Epoch', 28, 'completed out of', 45, 'loss:', '0.375', 'accuracy:  ', 0.87616944)
('Epoch', 29, 'completed out of', 45, 'loss:', '0.375', 'accuracy:  ', 0.87621742)
('Epoch', 30, 'completed out of', 45, 'loss:', '0.373', 'accuracy:  ', 0.87607348)
('Epoch', 31, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 32, 'completed out of', 45, 'loss:', '0.375', 'accuracy:  ', 0.87621742)
('Epoch', 33, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 34, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 35, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 36, 'completed out of', 45, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 37, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 38, 'completed out of', 45, 'loss:', '0.375', 'accuracy:  ', 0.87621742)
('Epoch', 39, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 40, 'completed out of', 45, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 41, 'completed out of', 45, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 42, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 43, 'completed out of', 45, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 44, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 45, 'completed out of', 45, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Accuracy', 0.87621742)
```

Figure 3: Epoch 45

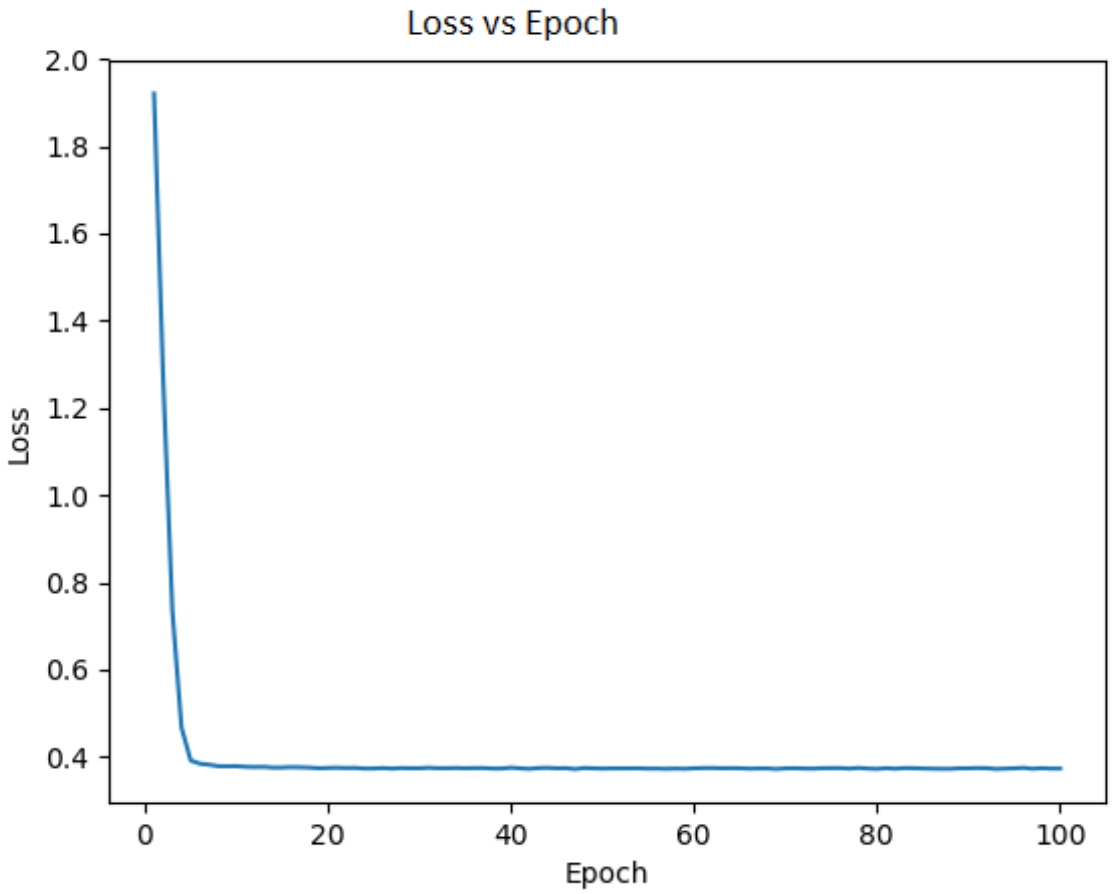

Figure 4: Epoch 100

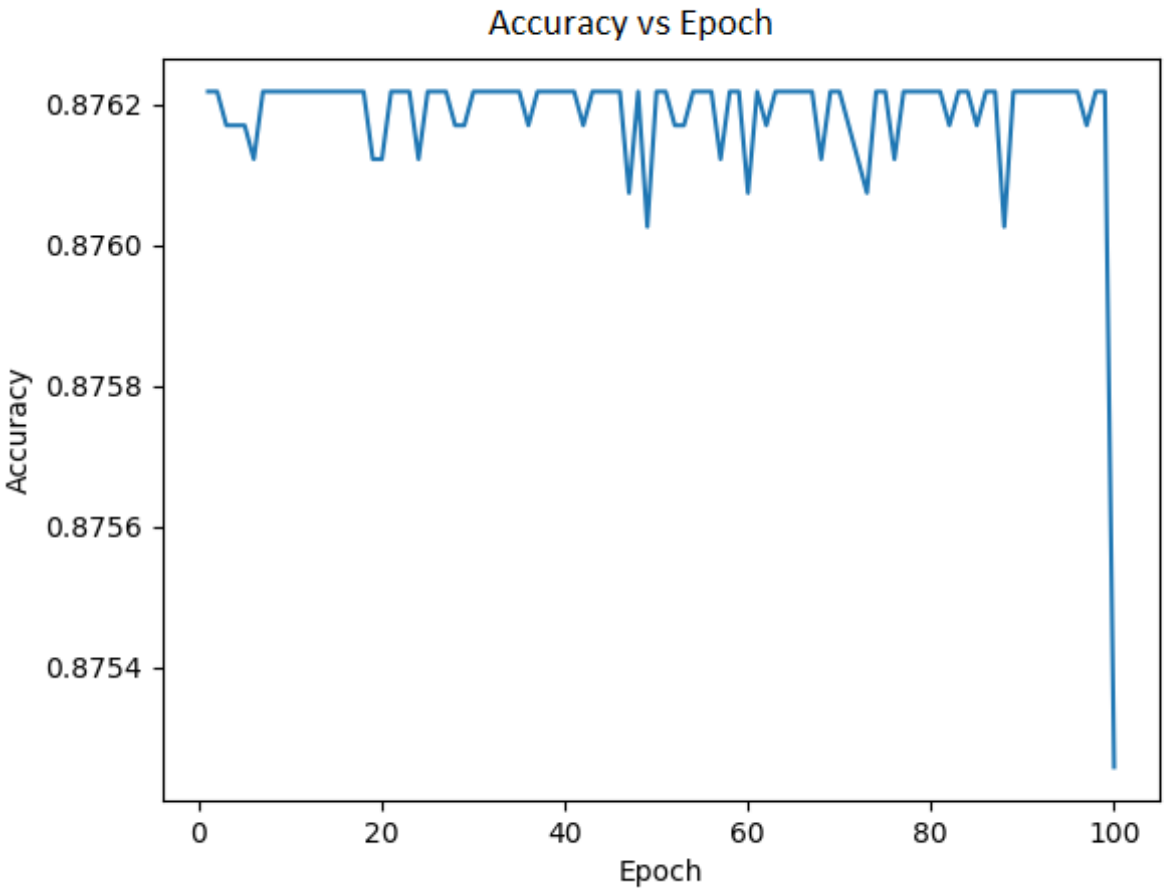

Figure 5: Epoch 100

```
('Epoch', 86, 'completed out of', 100, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 87, 'completed out of', 100, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 88, 'completed out of', 100, 'loss:', '0.372', 'accuracy:  ', 0.8760255)
('Epoch', 89, 'completed out of', 100, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 90, 'completed out of', 100, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 91, 'completed out of', 100, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 92, 'completed out of', 100, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 93, 'completed out of', 100, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 94, 'completed out of', 100, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 95, 'completed out of', 100, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 96, 'completed out of', 100, 'loss:', '0.375', 'accuracy:  ', 0.87621742)
('Epoch', 97, 'completed out of', 100, 'loss:', '0.372', 'accuracy:  ', 0.87616944)
('Epoch', 98, 'completed out of', 100, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 99, 'completed out of', 100, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 100, 'completed out of', 100, 'loss:', '0.373', 'accuracy:  ', 0.87525791)
('Accuracy', 0.87525791)
```

Figure 6: Epoch 100

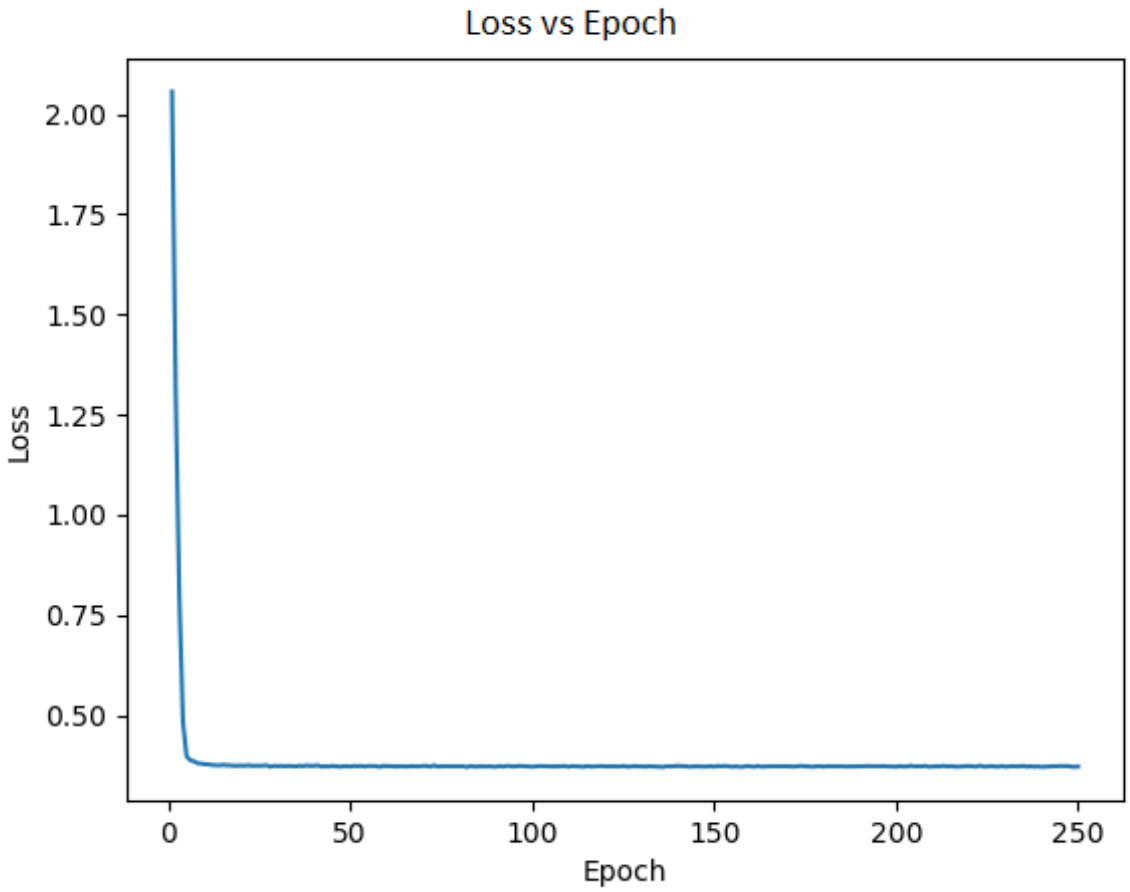

Figure 7: Epoch 250

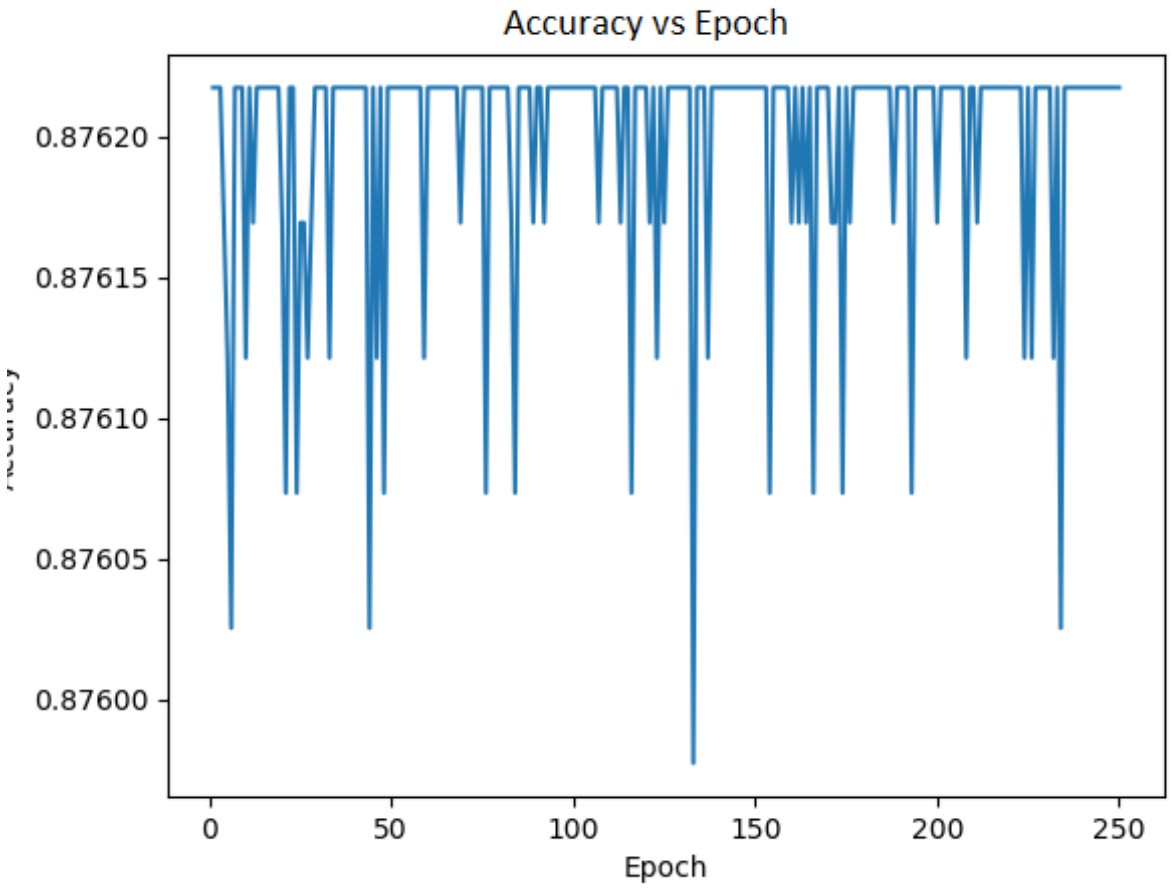

Figure 8: Epoch 250

```
('Epoch', 237, 'completed out of', 250, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 238, 'completed out of', 250, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 239, 'completed out of', 250, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 240, 'completed out of', 250, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 241, 'completed out of', 250, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 242, 'completed out of', 250, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 243, 'completed out of', 250, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 244, 'completed out of', 250, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 245, 'completed out of', 250, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 246, 'completed out of', 250, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 247, 'completed out of', 250, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 248, 'completed out of', 250, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 249, 'completed out of', 250, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 250, 'completed out of', 250, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Accuracy', 0.87621742)
```

Figure 9: Epoch 250

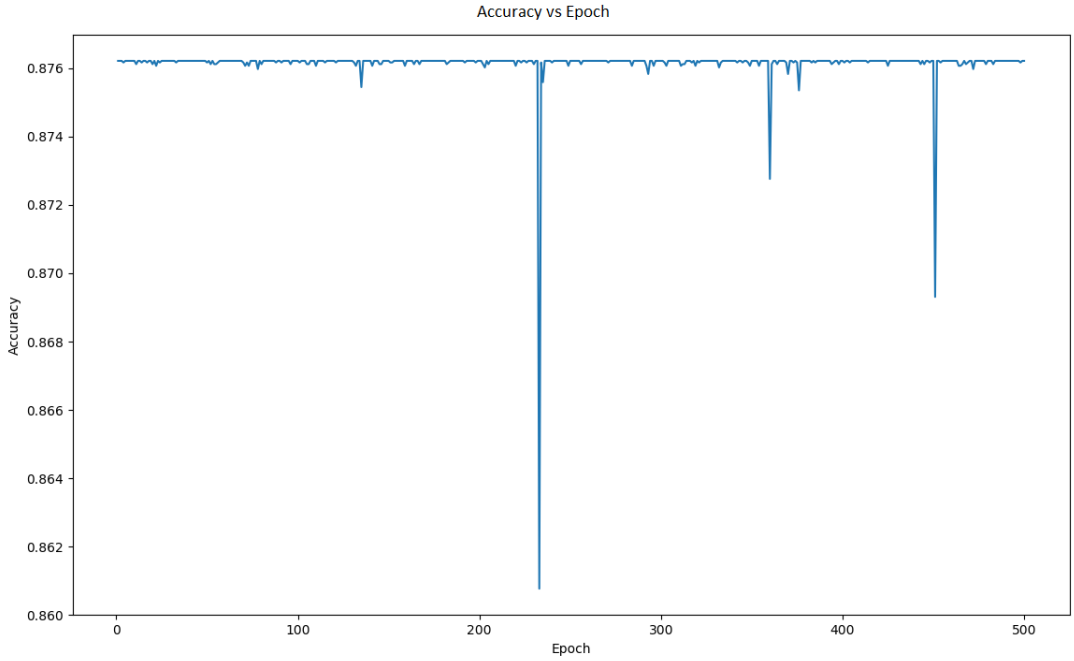

Figure 10: Epoch 500

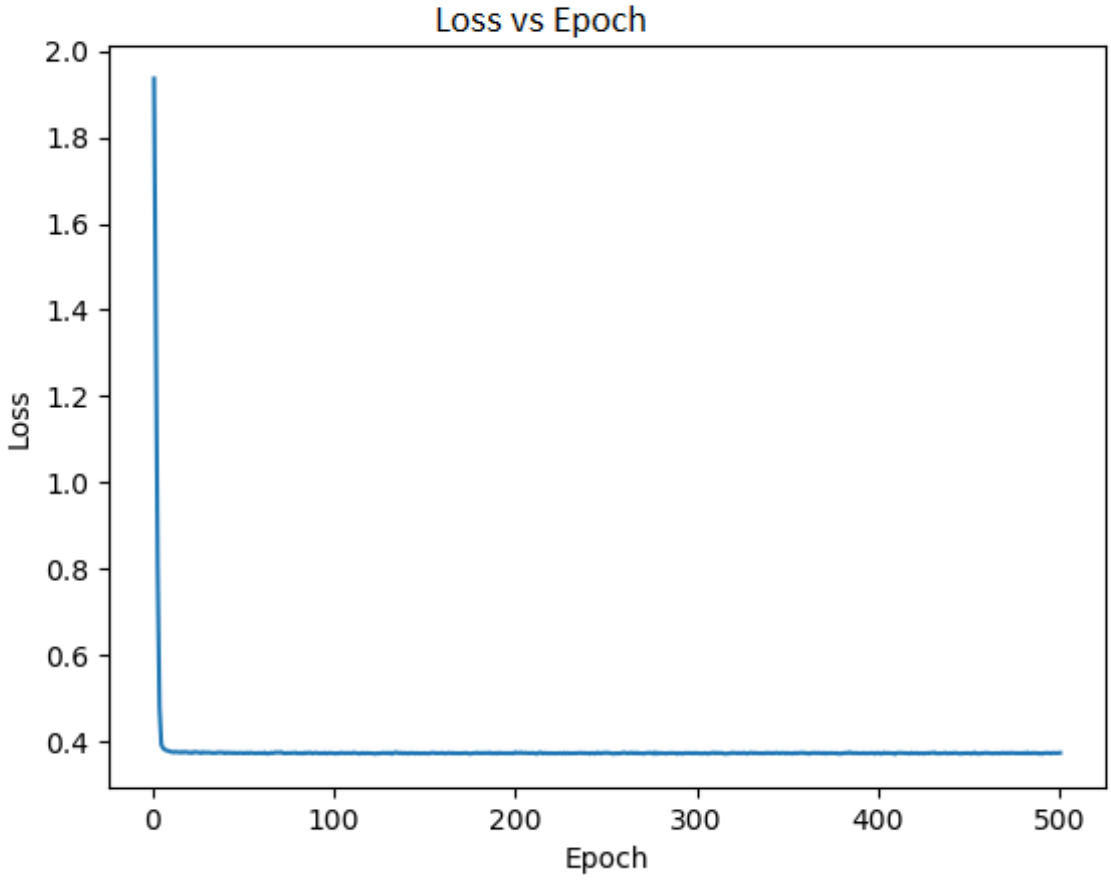

Figure 11: Epoch 500

```
('Epoch', 487, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 488, 'completed out of', 500, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 489, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 490, 'completed out of', 500, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 491, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 492, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 493, 'completed out of', 500, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 494, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 495, 'completed out of', 500, 'loss:', '0.372', 'accuracy:  ', 0.87621742)
('Epoch', 496, 'completed out of', 500, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 497, 'completed out of', 500, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Epoch', 498, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87616944)
('Epoch', 499, 'completed out of', 500, 'loss:', '0.373', 'accuracy:  ', 0.87621742)
('Epoch', 500, 'completed out of', 500, 'loss:', '0.374', 'accuracy:  ', 0.87621742)
('Accuracy', 0.87621742)
```

Figure 12: Epoch 500