# Analyzing viewers sentiment with deep learning.

**We will be developing a system that uses Deep Learning techniques to analyse whether a users movie review is positive or negative.**

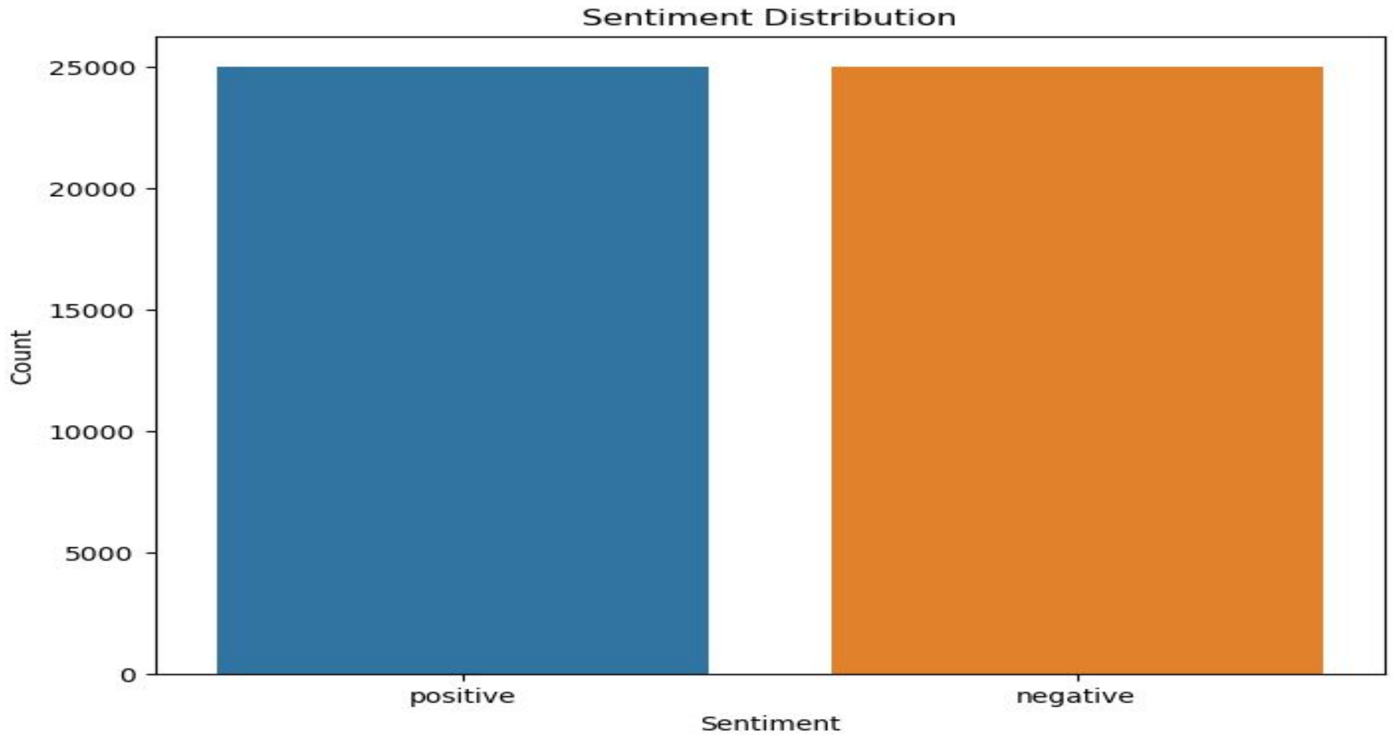
## Taking a look at the data.

Data Structure description: Structurally, our data set is pretty simple. It's one CSV file, used both for training and testing containing roughly 50 000 rows.
The CSV file contains two columns. Review and sentiment. The sentiment column contains the values as words, so "positive" and "negative"s

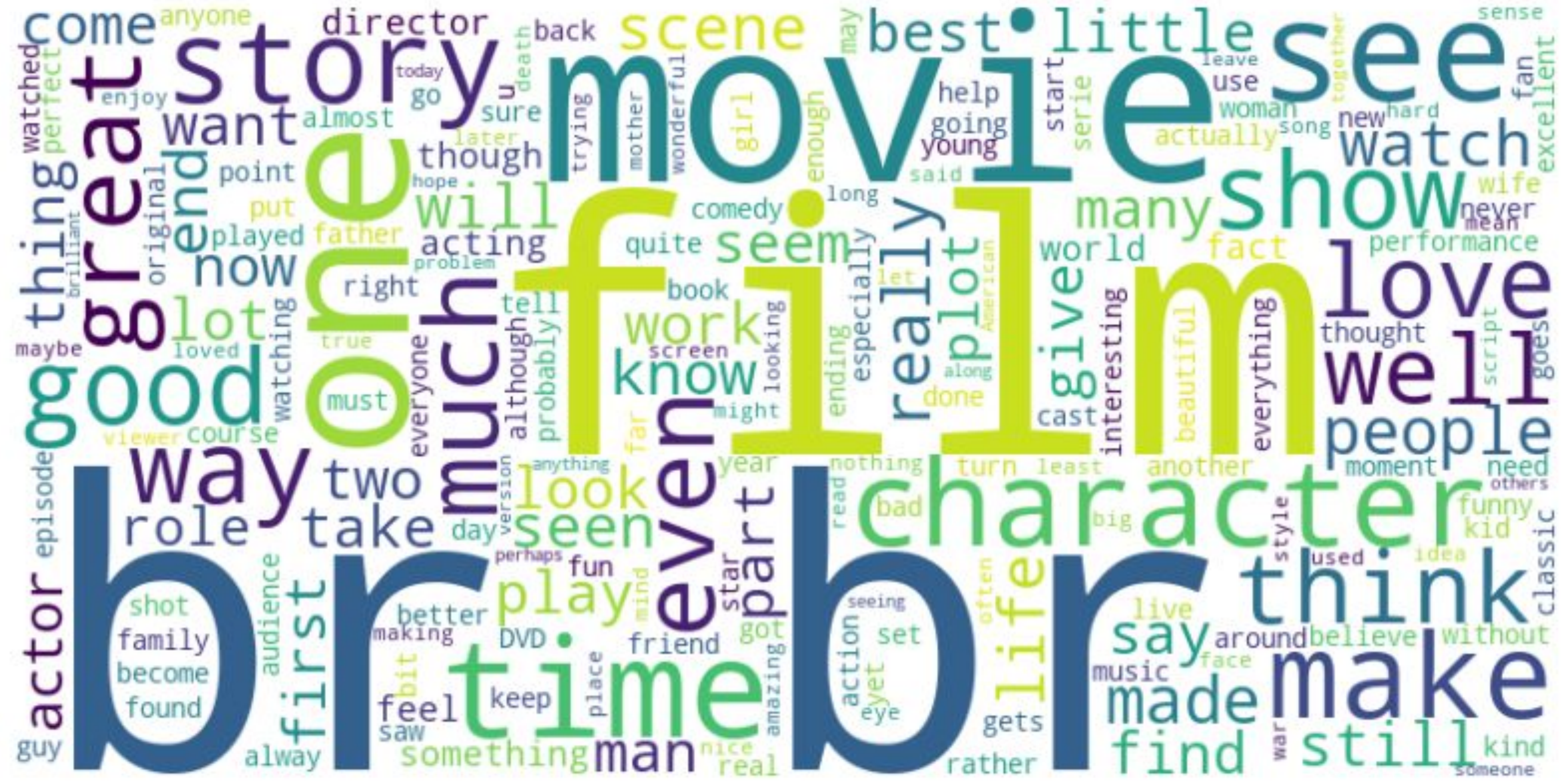
Data description: We are provided with a number of reviews for different movies. The Reviews are Html texts (contain a small amount of markup) that contain a viewers review about a movie they recently watched. The review usually contains a personal opinion about a movie with no structural requirements for the text.

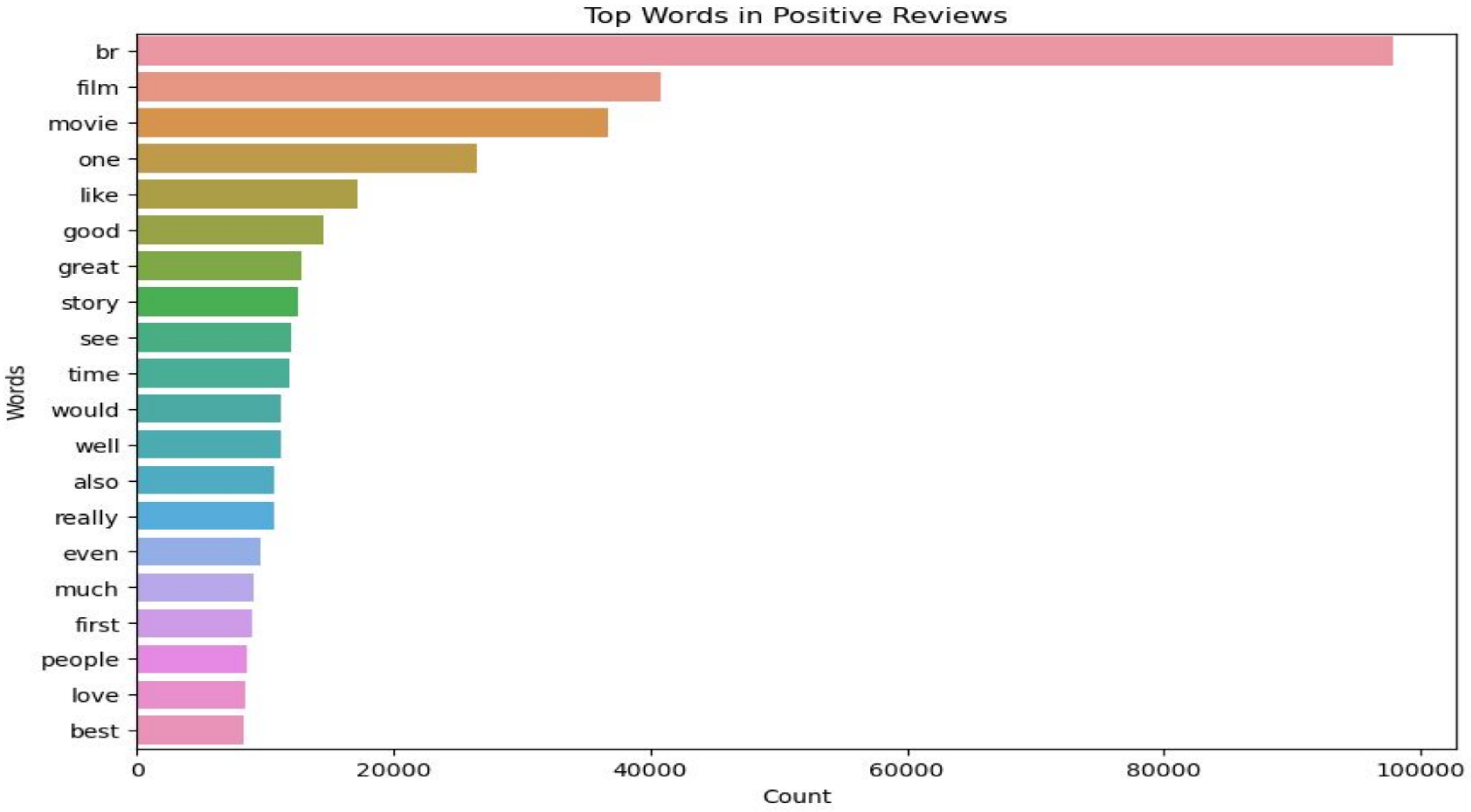
## Exploratory Data Analysis

For the EDA, Let’s get straight to the results. And take a look at some of the patterns



Positive Reviews Word Cloud




Top Words in Positive Reviews
br
film
movie
one
like
good
great
story
see
time
would
well
also
really
even
much
first
people
love
best
Words
0
20000
40000
60000
80000
100000
Count

## A simple model and data preparation

Prior to running a model we would usually clean and prepare data.

Our Data preprocessing will involve cleaning out html, lowercasing and removing stop words and punctuation.

We will also tokenize our reviews prior  to processing and convert the sentiment column into binary form.

## Our First Model.

A Simple RNN model. (Embedding layer, SimpleRnn + Dense Layer)

- Terrible Results. ~53% accuracy.
- Not really worth tuning considering the low base results.

## LSTM based model.

Just a reminder -
LSTM (Long Short-Term Memory) layer is a type of recurrent neural network (RNN) layer specifically designed to handle sequence data and learn long-term dependencies. LSTMs are particularly useful in tasks where the order and context of the data points are important, such as time series prediction, natural language processing, and speech recognition. Common for Sentiment analysis.

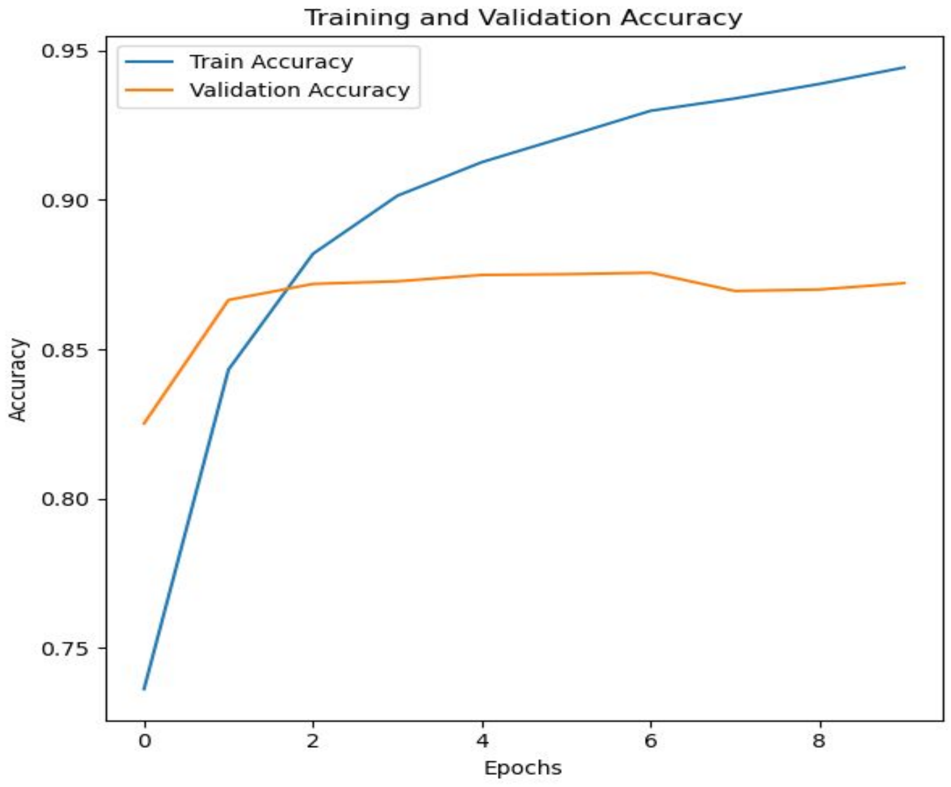
So our model will have the following layer workflow -
1) Embedding Layer
2) LSTM layer.
3) Dense Layer
4) Tune Hyperparameters

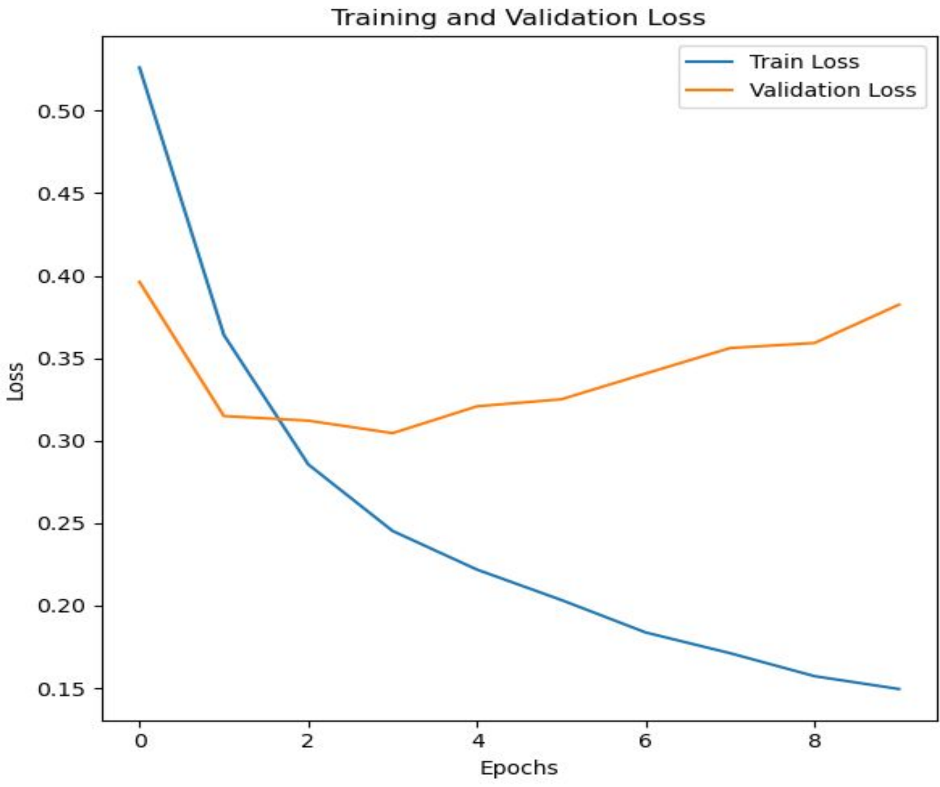
## Results

Accuracy - ~86% without hyper parameter tuning.

87% with hyperband tuner and 87.5% with random search tuner.

## Deeper analysis.





As we can see, our final model has solid and consistent results.
However, there is a noticeable gap between training and validation accuracy, with the validation accuracy plateauing around 85%, indicating that the model might be overfitting.

## Summary

In conclusion, i would like to say that we have managed to produce a model that shows great accuracy and adequate training speed.
Let's take a look at the things that could be improved and things that i am fond of in this project.

To improve:
1) I have not yet found a solution  that reduces the gap between training and validation accuracy.
2) I have tried several different approaches other than the ones included in this notebook. Most notebly a GRU layer instead of the LSTM.
Unfortunately, I have not managed to get a good result.
3) Maybe we could enhance the model by using pretrained embeddings.
4) Maybe we could use back translation to train the data on a larger and more varied dataset.

Things I like: I am generally satisfied with the results.