SMU
SINGAPORE MANAGEMENT UNIVERSITY

School of
Computing and
Information Systems

ISSS609 Text Analytics – Project Proposal

Applying Text Analytics on Unstructured Movie Reviews

Group 4

Team Members

Name	ID	Email	Role	Peer Evaluation[1]
CLARENCE SAN REN YI	clarencesan.2021	clarencesan.2021@mitb.smu.edu.sg		
FANG SIYU	siyu.fang.2021	siyu.fang.2021@mitb.smu.edu.sg		
HUANG JING	huangjing.2021	huangjing.2021@mitb.smu.edu.sg		
KHOO WEI LUN	weilun.khoo.2021	weilun.khoo.2021@mitb.smu.edu.sg		
LONG NU	nu.long.2021	nu.long.2021@mitb.smu.edu.sg		
MA ANBO	anbo.ma.2021	anbo.ma.2021@mitb.smu.edu.sg		

[1] Enter the relative contribution of the members: Enter 100% for every member if the contributions are equal.

Introduction

This project aims to apply text analytics and Natural Language Processing (NLP) techniques to extract meaningful insights from unstructured, textual movie reviews data. In doing so, we aim to streamline the process of evaluating releases for studios and streaming platforms, where the tracking and evaluation of catalogs must be performed regularly. We will demonstrate a proof of concept using a two-pronged approach: the first phase utilises sentiment analysis to demonstrate how to distill large amounts of textual into key consumer sentiment factors. The second dives deeper into the data to perform a text summarisation of the reviews to facilitate ease of analysis by abstracting meaning from lengthy text.

We expect our analysis to serve two main target stakeholders. The first are users of streaming platforms, who will be able to inform their viewing decision through a high-level summary from other viewer's feedback. The second are producers, content and catalog managers, marketers, and various business functions within these streaming platforms, who will be able to extract insights on a faster scale to adjust their content curation strategy in a timely manner.

Data Overview

Our data consists of 50,000 movie reviews that have been made publicly available through the Stanford AI lab for the purposes of benchmarking common NLP tasks. It contains 25,000 positive and 25,000 negative reviews. There are 49,582 observations after removing duplicates.

We intend to perform textual cleaning and preprocessing steps, which include but are not limited to: removing duplicates, lowercasing, removing html artifacts, removing punctuation, removing stop words, stemming / lemmatization and vectorization.

Table 1: Schema of the 50k movie reviews dataset

Column Name	Description	Field type
review	Movie reviews in text	String
sentiment	“positive” or “negative”	String

Additional details from a quick exploratory data analysis can be found in appendix *Exploratory Data Analysis*.

Analytics Tasks

Sentiment analysis is a supervised natural language processing technique that aims to identify the overall polarity of a given text (Bengfort, Bilbro & Ojeda 2018, p. 284). It is commonly utilised by companies to analyze data at scale and in real-time, discover insights for data-driven decisions and automate business processes. Companies such as Apple, Google, KFC and TripAdvisor are using sentiment analysis for reputation management, market research, product enhancement and customer analysis to elevate customer satisfaction (Sudhir & Suresh 2021, p. 207).

Our use case intends for streaming platforms to apply sentiment analysis to monitor the level of buzz and overall consumer sentiment of their content catalog. This has applications for reviewing, benchmarking and further analyses. For instance, the insights gained can guide the curating decisions of content managers to ensure that their organisations are providing content that are appropriate and wanted by consumers.

Text summarisation is a natural language processing technique that aims to distill vital information from longer text into an abridged and fluent version, while keeping the essence of the original text material. There are two main approaches to text summarisation:

1. Extractive summarisation, which identifies a subset of important words from sentences that rank higher on similarity to arrange them in a comprehensive manner
2. Abstractive summarisation, which selects words based on semantic understanding to generate a new summary (Munot & Govilkar 2014, p. 35).

In the past decade, the applications of text summarisation have made it easier and faster to condense large amounts of textual information into digestible sizes, analyse large amounts of search results, and perform web and email summarisation (Allahyari et al. 2017, p. 5). We propose that the characteristics of movie reviews, which are often lengthy, dialogue based, and contain overlapping ideas, render it an appropriate case for text summarisation (Allahyari et al. 2017, p. 5). More specifically, streaming platforms stand to gain time efficiencies from reducing the overall content of their reviews. These summaries in turn provide insights into customer preferences, which can serve as a guidepost for content curation.

Methodology

We propose the following methodology for the preceding analytics tasks:

Sentiment analysis: Preprocess the data, which includes an exploratory data analysis, generating of word / sentence vectors, train-validation-test splitting, benchmarking and experimentation.

Text summarisation: Preprocess the data, perform an extractive summarisation and finetune on an abstractive summarisation. For a more detailed process flow for both tasks, please refer to appendix *Workflow Charts*.

Train-validation-test splitting will be performed prior to modelling the Sentiment Analysis task. This is essential to avoid any data leakage, a phenomenon that occurs when a model is exposed to information in the train data that it will not have at inference (Tingle 2019). We also intend to implement a baseline solution before proceeding to finetune our results. This provides us with a set of indicators to better assess outcomes and effectiveness.

The overall accuracy and the F1 score, which is the harmonic mean of the precision and the recall, will be used to evaluate the individual model performance on the sentiment analysis task. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) will be used to evaluate the performance of the text summarisation method. The ROUGE benchmark, which compares a model's output to a reference summary (human translation) is the standard automatic evaluation measure for text summarisation tasks (Ganesan 2018; referencing Lin 2004). This metric is computed by calculating the number of overlapping words between the model output and the reference summary (Ganesan 2018, p. 1).

Indepth Details

Sentiment Analysis: We will utilise a Random Forest ensemble and Logistic Regression model as a traditional machine learning baseline. Random Forest ensembles are popular baseline approaches for sentiment analysis due to their robustness and ease of interpretability (Parmar, Bhanderi & Shah 2014). In addition to employing a bootstrapping mechanism to sample the data, the Random Forest algorithm also randomly selects a subset of features to build each weak learner (Bruce & Bruce 2017, p. 261) before aggregating these decisions to generate a single prediction (Geron 2017, p. 78).

This bagging characteristic has the net effect of reducing the overall variance of predictions; as weak learners are trained independently from each other, the ensemble model often learns correlations in the data that are not overfitting to any esoteric structure in the training data (Geron 2017, p. 199).

Equation. 2: Mathematical formulation of a single decision tree; each tree selects the best attribute that maximises some information gain criterion; commonly used criterions are GINI impurity and entropy gain

$$argmax\ IG(X) = I(Y) - I(Y \mid X)$$

The Logistic Regression model is a popular discriminative model that is frequently used to estimate the probability of an instance belonging to a particular class (Geron 2017, p. 9). In the binary case, the model computes a weighted sum of features and outputs a value between 0 to 1 for the positive class through passing the output logits through the sigmoid function (Bruce & Bruce 2017, p. 233). We will utilise Logistic Regression as a second benchmark model due to its computational speed and ease of interpretability (Bruce & Bruce 2017, p. 233).

Equation 3: Mathematical specification of the Logistic Regression model; σ is the sigmoid function

$$\hat{y} = \sigma\left(\mathrm{x}^{\mathrm{T}}\theta\right), \text{ where } \sigma(z) = \frac{1}{1+e^{-z}}$$

We aim to generate performant results through the application of two training techniques. The first is to train a deep learning model that is inspired by sequence-aware networks, such as Long-Short-Term Memory (LSTM) neural networks (an adaptation over the standard recurrent neural network architecture). Sequence-aware networks, such as RNNs, utilise feedback connections to maintain some internal memory state (quote Goodfellow). This effectively allows RNNs to retain information from previous inputs; as a trivial example, some state at *t-1* is mapped into the hidden unit of the state at *t* (Goodfellow, Bengio & Courville 2016, p. 376).

The generic RNN structure is applicable to sequential data of arbitrary lengths, such as long-form textual data or audio recordings, which makes them ideal for NLP tasks that require context from surrounding inputs (Geron 2017, p. 575). The practical challenge of the vanilla specification is that it is necessarily lossy over long sequences; this has computational implications for training time (to train a deep RNN requires many time steps) and the practical limitation that RNNs cannot retain the "long-term memory" of previous states (Geron 2017, p. 609).

Equation 4: Mathematical specification of a single recurrent cell, where ϕ is some activation function; every recurrent neuron learns two distinct sets of weights: one for the present inputs and the other for the outputs from the previous time step

$$y_t = \phi({x_t}^T \cdot w_x + {y_{t-1}}^T \cdot w_y + b)$$

To address these challenges, we intend to add a gated cell, known as the LSTM cell, which manages two states, a long-term and a short-term state, through the use of three logistic sigmoid gates and one tanh layer (Goodfellow, Bengio & Courville 2016, p. 412). The main advantage of using LSTM cells is that it addresses the vanishing gradient issue through its gated design: which limits the information that can be passed onto the next cell (a forget gate) (Goodfellow, Bengio & Courville 2016, p. 412).

Equation 5: The key characteristic of LSTM is the addition of the gated design: Mathematical specification of the forget gate unit, which learn to output a value between 0 and 1; $\boldsymbol{b}, \boldsymbol{U}, \boldsymbol{W}$ are the biases, input weights and recurrent weights for the forget gates; the external input gate is computed similarly, but with its own parameters (Goodfellow, Bengio & Courville 2016, p. 412)

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right)$$

The second training technique we propose to improve on baseline performance is to introduce transfer learning through using pre-trained models. The main idea to motivate transfer learning is that many text-based problems share an underlying, low-level representation which can be transferable across tasks (Goodfellow, Bengio & Courville 2016, p. 536). Several pre-trained word and sentence embeddings are available for transfer learning; the current most popular strategy is to fine-tune a BERT (Bidirectional Encoder Representations from Transformers) model, or any of its variations (Maltoudoglou et al. 2020). BERT-base, for instance, is trained on the *BookCorpus* (800 million words) and *English Wikipedia* (2,500 million words) (Devlin et al. 2019).

In addition to its rich representation, BERT-base relies on a special language training process known as masked-language modelling (MLM): word predictions are generated by masking a word within a sequence and training the model to bidirectionally use the neighbouring words to predict the masked word (Muller 2022). This contrasts with older language models, such as GloVe (Global Vectors for Word Representation) and Word2Vec, which relies on a unidirectional masking during the training process (Muller 2022). This bidirectional encoding allows BERT and its derivative to achieve state-of-the-art performance for many NLP tasks (Muller 2022).

Text summarisation: approaches can be largely condensed into two main techniques: the extractive approach attempts to select a subset of words that represent the critical points while masking unimportant ones, while the abstractive variant learns to paraphrase and compose a summary from the input (Zhang et al. 2019, p. 789).

Our extractive implementation will utilise a statistical approach to determining word importance by substituting each word found in the input with their corresponding weighted frequency (Bengfort, Bilbro & Ojeda 2018, p. 184). A threshold for sentence / phrase inclusion is computed such that each sequence of words is assigned a normalised importance score based on their frequency of occurrence within the document (Geron 2017, p. 611). This approach has the advantage of ease of computation and interpretation; however, it is reliant on existing terms found within the documents, and is unable to generate new summaries based on the input data.

Previous studies (Zhang et al. 2019) have shown that utilising a pre-trained, contextualised language model as the decoder results in better performance at inference time. This has been mainly attributed by Zhang et al. (2019) by the existing difficulties for decoders to learn summary representations and contextual cues without a large corpus of data (Zhang et al. 2019, p. 789). Our abstractive approach is inspired by the Pretraining-Based Natural Language Generation design proposed in Zhang et al. (2019). The overall architecture of the model is best represented by the two-stage encoder-decoder process. First, we will use BERT to map the input sequence to word embeddings. Then, a summary draft is generated using a left-context-only decoder. Finally, the generated summary draft is used to refine the content of the summary.

Eq. 6: Objective function of the left-context-only decoder: to minimize the NLL of the conditional probability; note the key difference with the refine decoder

$$L_{dec} = \sum_{t=1}^{|a|} -\log P\left(a_t = y_t^* \mid a_{<t}, H\right)$$

Eq. 7: Objective function of the refine decoder, which receives the generated summary draft as inputs and outputs a refined summary

$$L_{refine} = \sum_{t=1}^{|y|} -\log P\left(y_t = y_t^* \mid a_{\neq t}, H\right)$$

References

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). Text summarization techniques: a brief survey. *ARXIV*, preprint arXiv:1707.02268. Retrieved from: https://arxiv.org/abs/1707.02268

Bengfort, B., Bilbro, R., Ojeda, T. (2018). *Applied Text Analysis Python Language Aware*, O'REILLY.

Devlin, J., Chang, M.W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *Applied Intelligence*. doi:10.1007/s10489-022-03511-6, Retrieved from: https://arxiv.org/pdf/1810.04805v2.pdf

Ganesan, K. (2018). *ROUGE 2.0:* Updated and Improved Measures for Evaluation of Summarization Tasks, *ARXIV*. arXiv:1803.01937v1, Retrieved from: https://doi.org/10.48550/arXiv.1803.01937

Géron, Aurélien. (2017). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, MIT Press. 2017 edition

Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*, MIT Press. Retrieved from: http://www.deeplearningbook.org

Lin, C.Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In Text Summarization Branches Out, pp. 74–81, *Association for Computational Linguistics*. Retrieved from: https://aclanthology.org/W04-1013/

Maltoudoglou, L., Paisios, A., & Papadopoulos, H. (2020). BERT-based Conformal Predictor for Sentiment Analysis. *Conformal and Probabilistic Prediction and Applications* (pp. 269-284). PMLR.

Muller, Britney. (2022). BERT 101-State Of The Art NLP Model Explained, *Huggingface*, Retrieved from: https://huggingface.co/blog/bert-101

Munot, N., & Govilkar, S. S.. (2004). Comparative study of Text Summarization Methods, *International Journal of Computer Applications*, 102(12).

Parmar, H., Bhanderi, S., & Shah, G. (2014). Sentiment mining of movie reviews using Random Forest with Tuned Hyperparameters. In *International Conference on Information Science* (pp. 1-6). *Kerala*.

Tingle, Max. (2019). Preventing Data Leakage in Your Machine Learning Model, *Towards Data Science*, Retrieved from: https://towardsdatascience.com/preventing-data-leakage-in-your-machine-learning-model-9ae54b3cd1fb

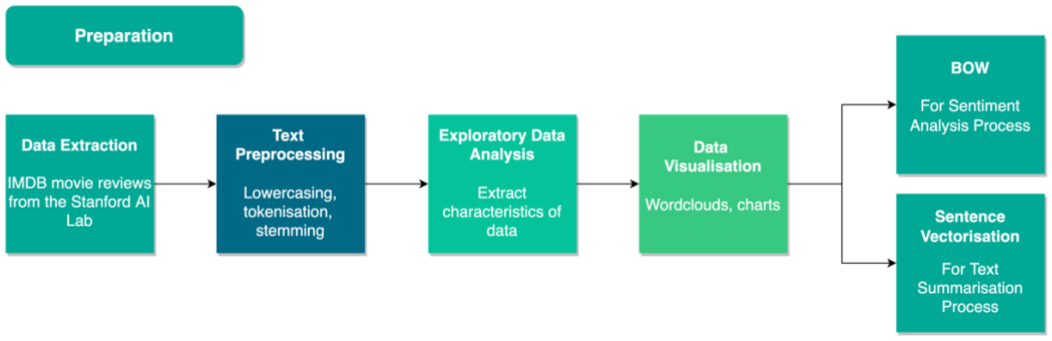
Sudhira, Prajval & Suresh, D. Varun Varun. (2021). Comparative study of various approaches, applications and classifiers for sentiment analysis, *Science Direct*, Retrieved from: https://www.sciencedirect.com/science/article/pii/S2666285X21000327

Zhang, H., Xu, J., & Wang, J. (2019). Pretraining-based Natural Language Generation for Text Summarization. *arXiv* preprint arXiv:1902.09243.

Appendix

Indepth Details

Figure 1: Preprocessing workflow



Preparation
Data Extraction
IMDB movie reviews from the Stanford AI Lab
Text Preprocessing
Lowercasing, tokenisation, stemming
Exploratory Data Analysis
Extract characteristics of data
Data Visualisation
Wordclouds, charts
BOW
For Sentiment Analysis Process
Sentence Vectorisation
For Text Summarisation Process

Figure 2: Sentiment Analysis workflow

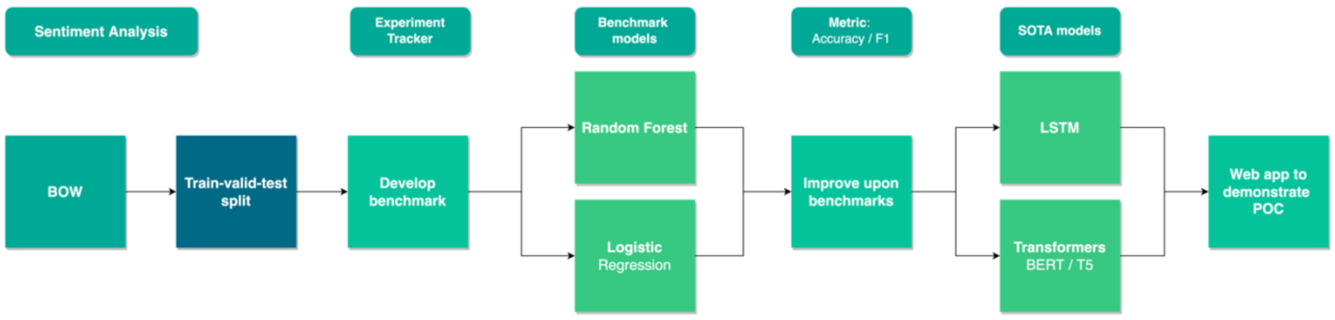
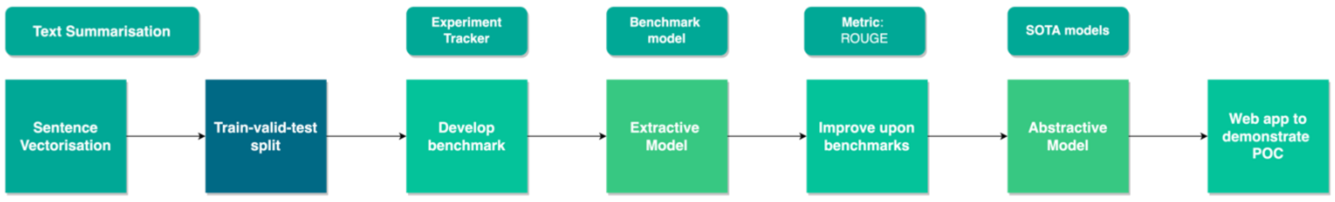


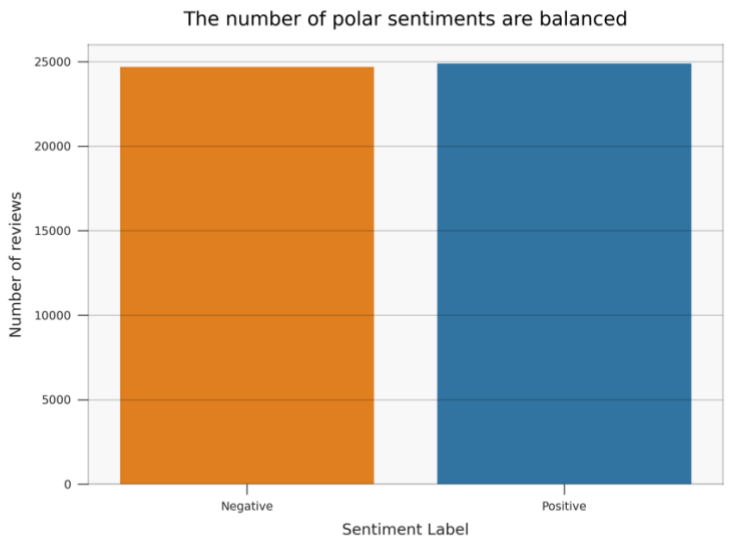
Figure 3: Text Summarisation workflow



Exploratory Data Analysis

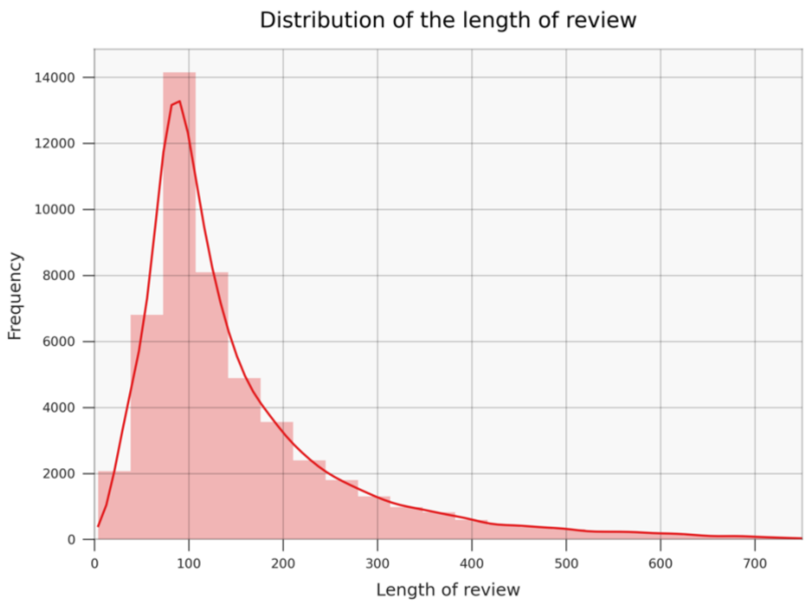
The positive and negative sentiment labels in the dataset is balanced.

Figure 4: Countplot of the number of positive and negative reviews after removing duplicates



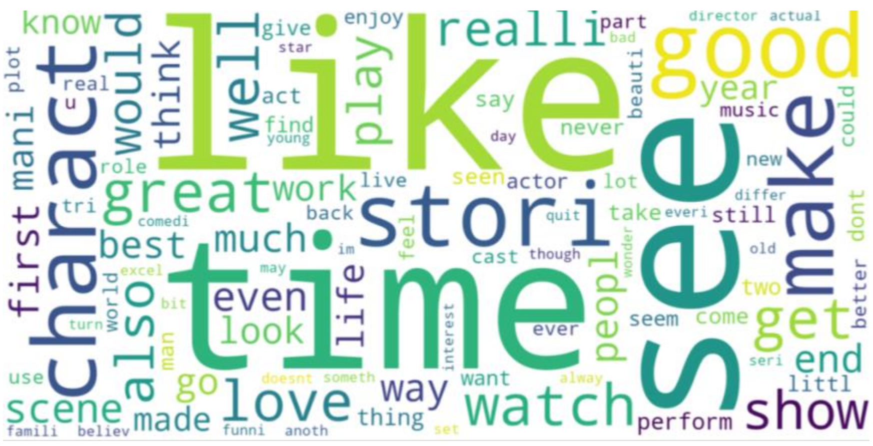
The chart below shows that most of the reviews are under 200 with 75th percentile at 142. Among the longer reviews, 26 reviews have more than 600 words and 4 have more than 1000 words.

Figure 5: Distribution of the length of clean review text



Word cloud of positive reviews reveals the most frequent words used in positive reviews are words such as like, good, love, see, time.

Figure 6: Word cloud of positive reviews reveals the most frequent words used in positive reviews are words such as like, good, love, see, time.



From the figure below, we can observe that the frequent words used in negative reviews are words such as like, even, make. It is surprising that "like" is in the list, but upon further investigation, we that it is not used in the same way as in positive reviews. In negative reviews, "like" is often used to describe the movie. For example, "So the tale goes like this", "Sort of like a high school film project" and "it's like a million other miles of bad". This is an indication that the sentiment conveyed by the word "like" depends on how it is used in the sentence.

Figure 7: Word cloud of negative reviews

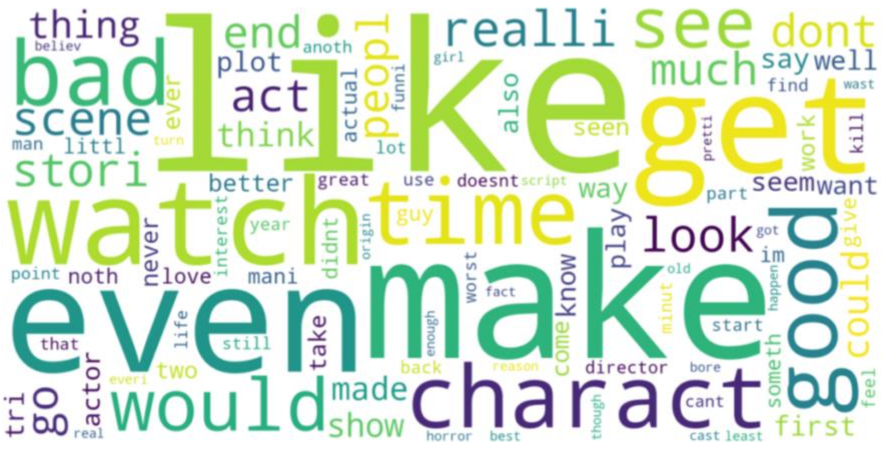
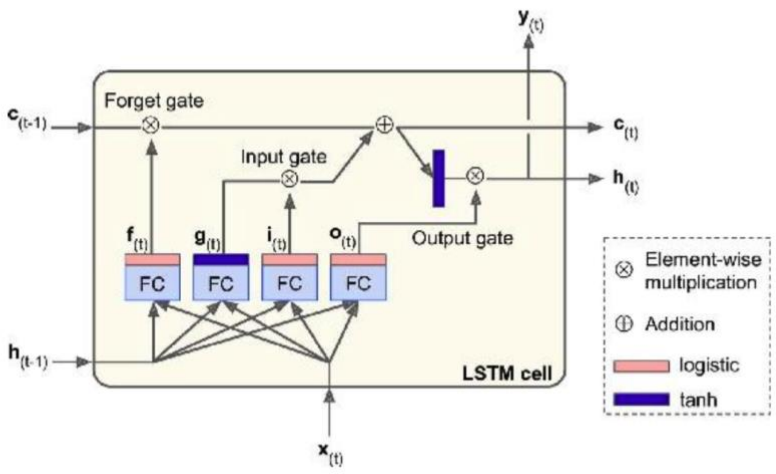


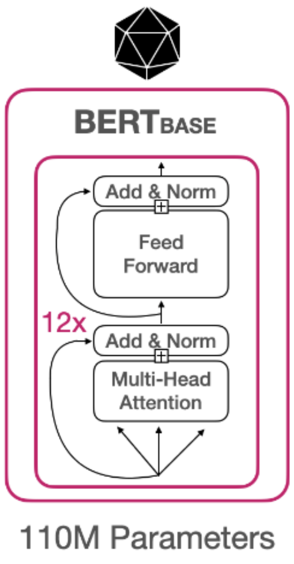
Figure 8: Architecture of a Long-Short-Term-Memory Cell consisting of three logistic gates and one tanh gate (Geron 2017, p. 688)



BERT

BERT base has 12 transformer layers, 12 attention heads, and 110M parameters

Figure 9: Overview of the BERT Base architecture



BERTBASE
Add & Norm
Feed Forward
12x
Add & Norm
Multi-Head Attention
110M Parameters