h e g

Haute école de gestion
Genève

Attention is All Your Business Needs: A Sentiment Analysis of Customer Reviews

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

par :

Komivi Jarod Manuel Assiamua

Conseiller au travail de Bachelor :

Kalousis Alexandros, Chargé de cours HES

Genève, 12 août 2024

Haute École de Gestion de Genève (HEG-GE)

Filière informatique de gestion

Hes·SO GENÈVE
Haute Ecole Spécialisée
de Suisse occidentale

Declaration

Ce travail de Bachelor est réalisé dans le cadre de l’examen final de la Haute école de gestion de Genève, en vue de l’obtention du titre « Bachelor of Science HES-SO en informatique de gestion ».

L’étudiant a envoyé ce document par email à l'adresse remise par son directeur de mémoire afin qu’il l’analyse à l’aide du logiciel de détection de plagiat COMPILATIO.

L’étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J’atteste avoir réalisé seul le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève , le 12 août 2024

Komivi Jarod Manuel Assiamua

Acknowledgements

I extend my deepest gratitude to my thesis advisor, Mr. Kalousis Alexandros, whose unwavering support and invaluable guidance have been fundamental to the success of this research. His profound expertise and insightful feedback have profoundly shaped this work.

I am also profoundly grateful to my family for their unconditional love, patience, and understanding. Their unwavering support has been a constant source of strength and motivation. I extend my heartfelt thanks to God for the blessings and opportunities that have led me to this point.

I would also like to thank those who have read the thesis and provided valuable feedback. Their insights and suggestions have greatly enriched this work.

Lastly, I would like to acknowledge the significant impact of Mr. Christopher Manning's NLP with deep learning courses (CS 224N) at Stanford Engineering. The knowledge and insights gained from these courses have been crucial in realising this work.

Abbreviations

- WOM: Word of Mouth
- E-WOM: Electronic Word of Mouth
- CGM: Consumer-Generated Media
- SEO: Search Engine Optimisation
- AI: Artificial Intelligence
- NLP: Natural Language Processing
- NN: Neural Network
- RNN: Recurrent Neural Network
- BERT: Bidirectional Encoder Representations from Transformers
- GPT: Generative Pre-trained Transformer
- LSTM: Long Short-Term Memory
- GD: Gradient Descent
- SGD: Stochastic Gradient Descent
- GloVe: Global Vectors for Word Representation
- TF-IDF: Term Frequency-Inverse Document Frequency
- Word2Vec: Word to Vector
- SVM: Support Vector Machine
- FNN: Feedforward Neural Network

Contents

Abstract

Customer reviews have become indispensable business resources. They offer valuable insights into consumer experiences and preferences and are crucial in shaping business strategies. To derive critical insights from reviews, Sentiment Analysis, a Natural Language Processing tool, aims to evaluate the underlying sentiment behind a text. However, due to the complexity of human language, traditional methods often struggle with understanding nuances. This bachelor's thesis, "Attention is All Your Business Needs: A Sentiment Analysis of Customer Reviews," explores applying Deep learning techniques to enhance sentiment analysis of customer reviews. The paper begins by introducing the concept of customer participation, which, throughout the years, has resulted in a growing number of customer reviews. It then introduces the field of Natural Language Processing (NLP), demonstrating how the attention mechanism can significantly enhance a model's accuracy and overall performance. Simulating a commission to develop a sentiment analysis model for an investment firm has led to the development of a sentiment analysis model through two approaches. The model aims to analyse customer reviews and feedback to predict market trends and sentiment, aiding in making informed investment decisions. The paper concludes by showcasing various use cases of sentiment analysis by big tech companies, emphasising how other businesses can learn from these examples to stay competitive in the market.

Scope

This paper focuses on enhancing sentiment analysis of customer reviews through deep learning techniques, specifically the attention mechanism. The following areas are considered beyond the scope of this memoir:

- Regularisation for overfitting
- Dependency parsing
- Coreference resolution
- Word agreement
- Class imbalances
- Cross-lingual transfer learning
- Linear models

These topics are excluded to maintain a focused exploration of the primary research objectives.

Figures

1. Introduction

Understanding customer opinions through reviews is critical for success in today’s business world. My thesis, **"Attention is All Your Business Needs: A Sentiment Analysis of Customer Reviews,"** explores how deep learning through Attention Mechanisms enhances models' capabilities to analyse these reviews.

This research applies advanced Natural Language Processing (NLP) techniques to better understand and leverage customer feedback sentiments. By focusing on attention-based models, I demonstrate their ability to analyse text and interpret sentiments that support informed business decisions. To illustrate the practical applications, I present a simulated project for an investment firm, highlighting the real-world potential of these models. Additionally, the thesis examines how major tech companies use similar approaches to maintain a competitive edge, offering valuable insights for businesses across various sectors.

The primary objective of this study is to showcase how sophisticated AI tools can transform vast amounts of customer review data into actionable insights, driving strategic business decisions. By bridging the gap between cutting-edge AI research and practical business applications, this thesis aims to provide a roadmap for companies seeking to harness the power of sentiment analysis in their decision-making processes.

2. Customer Reviews

Customer behavioural patterns are highly complex and vary from one individual to another (Vishesh 2020). Research examining customers' decision-making for a standard repeated purchase (Hoyer 1984) found that consumers tend to apply straightforward choice rules that provide satisfaction while allowing a quick and effortless decision. The ***word-of-mouth communication (WOM)*** has been cited as the most critical influence in purchasing essential consumption products or adopting personal services (De Bruyn).

The Cambridge Dictionary defines word-of-mouth as "The process of telling people you know about a particular product or service, usually because you think it is good and want to encourage them to try it" (C. E. Dictionary s.d.). Word-of-mouth communication is present in most social interactions; it is everywhere, from chatting about vacations to gossiping about co-workers (Berger 2014).

Communication is one of the deepest needs of human beings and is central to a civilised society, argues (Gallager 2008). In psychology, understanding how people can influence others is the aim of ***Social Psychology***, "The scientific attempt to explain how the thoughts, feelings and behaviours of individuals are influenced by the actual, imagined, or implied presence of other human beings" (Fiske 2018). The Canadian psychologist and author ***Jordan B. Peterson,*** in his book « ***Beyond Order*** », explains the reason for this need: "We depend on content communication with others to keep our minds organised; we mostly think by talking, we must submit the strategies and tactics we formulated to the judgement of others to ensure their efficiency and resilience. Without the intermediation of the social world, we would be overwhelmed by the world" (Peterson 2021).

Consequently, this has encouraged companies and firms to adopt ***word-of-mouth marketing*** by encouraging their customers to share their experiences about the product and the brand (Bernoff, Schadler 2010). Furthermore, the rise of the Internet and social media has introduced a new kind of communication: ***the electronic word-of-mouth (e-WOM).***

The e-WOM has completely revolutionised consumer behaviour by vastly expanding customers' options for gathering information from other customers (*Electronic word-of-mouth via consumer-opinion platforms: What motivates consumers to articulate themselves on the Internet?*). To demonstrate the scale of the e-WOM compared to traditional WOM, (Huete-Alcocer 2017) chooses four criteria:

- Credibility
- Privacy
- Diffusion speed
- Accessibility

Figure 1 - Differences between WOM and E-WOM

	WOM	**E-WOM**
Credibility	The receiver of the information knows the communicator (positif influence on credibility).	Anonimity between the communicator and the receiver of the information (negative influence on credibility).
Privacy	The conversation is private, interpersonals (via dialogs) and concluded in real time.	The information is publicly shared and can be viewed by anyone at any time.
Diffusion speed	Messages are spread slowly. Users must be present when the information is being shared.	Messages are conveyed more quickly between users thanks to internet.
Accessibility	Less accessible.	Easily accessible.

(Huete-Alcocer 2017)

Now and then, through blogs, forums, review websites, emails, social media, YouTube channels and third-party websites, customers can spread information by creating and sharing their content in what is identified as ***consumer-generated media (CGM)*** (*Effects of Restaurant Satisfaction and Knowledge Sharing Motivation on eWOM Intentions*). This profound shift, driven by the proliferation of digital platforms, has allowed CGM to leverage the concept of customer participation.

2.1 Who Is A Customer?

The Cambridge Business English Dictionary defines a customer as "a person or an organisation that buys a product or a service" (C. E. Dictionary s.d.) In recent years, businesses have emphasised the concept of customer satisfaction, with customers as the principal factor in their chains. Nonetheless, it has been a hurdle to come to a consensus about defining a customer because different classes of customers can be identified from one industry to another.

It turns out that a customer is not only one who obtains a product or service through a direct transaction but also through an indirect transaction; subsequently, every party involved in a chain of transactions directed to a final customer is also a customer. ***Robert Simons,*** Professor of Business Administration at Harvard Business School, argues that businesses should choose the best primary customer. He demonstrates this by giving the example of Amazon, which, among its four types of customers (consumers, sellers, enterprises, and content providers), devotes most of its resources to pleasing consumers even if it means that the rest of its customers feel shortchanged (Simons 2014). It is a "student" for a "school", a "patient "for a "hospital", a "client" for a "shop", and a "stakeholder" for a "company" claims (Caplan 2002).

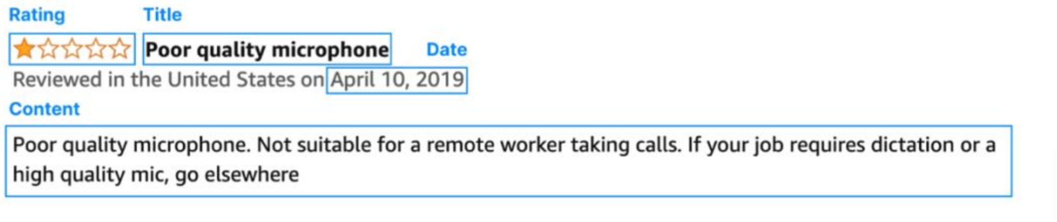
Research conducted by the ***MIT Sloan Magazine review*** on various industries and companies about their customer participation highlights that 82% of companies actively encouraged customers to recommend the company to others. In contrast, only 18 % tended to encourage customers to volunteer constructive ideas and suggestions to improve products and services. The research points out that businesses have been reluctant to give the power of communication to their customers, which results in a feeble reciprocal relationship. "Customer-to-customer reviews and customer-to-business interactions can influence a customer to buy more of a company's products and services; hence, encouraging them to provide feedback and suggestions helps tie them more closely to the business and strengthen the relationship", claims (Merlo, Eisingerich, Auh 2013). This emphasises the importance of letting the customer review the business.

2.2 What Is A Review?

From a customer perspective, "A review is an evaluation of a product or service made by someone who has purchased and used, or had experience with a product or service"(*Customer review* 2024). A typical review includes the customer's opinion, rating, and details about their experiences (Bhasin 2023, p. 91). Amazon has been one of the pioneers in encouraging customer participation through online reviews since 1995 (*2023 Ratings & Reviews Report: Amazon Edition* 2023; *Amazon improves the customer reviews experience with AI*).

A classic representation of a customer review on Amazon over a particular product follows:

Figure 2 - A customer review on Amazon



Poor quality microphone

Reviewed in the United States on April 10, 2019

Poor quality microphone. Not suitable for a remote worker taking calls. If your job requires dictation or a high quality mic, go elsewhere

(*Amazon.com*)

In this not-so-content review, we can identify four main components :

- **The rating**: a note which sums up the customer's overall experience.
- **The title**: the reason for the review.
- **The date**: the date of the review.
- **The content**: the literary expressions of the customer's feelings about the product.

Provided reviews represent valuable insights into the quality and performance of products and services, influencing purchasing decisions and brand perceptions.

2.3 The Power of Reviews

In a European Commission study, the World Tourism Organisation (UNWTO) reports that 82% of respondents read consumer reviews before shopping. In contrast, another survey completed by Australian consumers suggests that 40% ‘always’ or ‘often’ used online reviews to assess customer services (*Recommendations on the Responsible Use of Ratings and Reviews on Digital Platforms*). Tripadvisor, the gigantic American online travel agency, recorded 26 million reviews submitted to their site in 2020 alone when their 2023 transparency review estimated that number to be more than 30 million reviews posted by more than 17.4 million Tripadvisor members, with 40% of travellers viewing the reviews as significantly helpful (*2023 Tripadvisor Review Transparency Report*).

On the other hand, the 2023 Amazon report indicates more than 500 million reviews, with 82% of adults checking the reviews, product pages with reviews consulted 3.5 times more frequently, and sales increased by 26% with a one-star rating increase (*McAuley-Lab/Amazon-Reviews-2023 · Datasets at Hugging Face* 2024; Push-Pull 2022). Since 1995, the firm has registered 1.5 billion reviews from 125 million customers...(*3 steps Amazon is taking to stop fake reviews*). On its side, Google holds more than half of online reviews, with an increase of 65% in 2020 compared to 2019, with more than 60% of consumers likely to check Google reviews before they visit the business in person. Statistics also showcase that Google reviews are among the most critical factors in local search engine optimisation(SEO)(*Google Reviews: The Complete Guide for Businesses*).

These numbers highlighted how critical customer reviews have become for businesses. Reviews are precious resources, offering essential insights into their products and services' perception. They are crucial in building branding, trust, customer engagement, and leveraging revenue (Katole 2022; Mumuni et al. 2020).

Now that we have highlighted opportunities to leverage business through online reviews, how do we analyse and derive information from millions of reviews? We must analyse each review individually to detect the underlying sentiment experienced by the customer. This seems repetitive and gruelling. Thankfully, when faced with a redundant task that we can do but will take too much time, we can program a machine to do it for us (Azencott 2022). This is where ***artificial intelligence*** (AI) intervenes.

2.4 Fake Customers Reviews

Before starting with artificial intelligence, a case about online reviews' drawbacks must be made. The problem with allowing anyone to participate is that not everyone is driven by the mindset of ameliorating the process. Thus, fake online reviews have been and still are a major concern for businesses.

The TripAdvisor transparency report defines a fake review as "One submitted by someone who is either biased in some way and/or did not have a personal experience with the business they reviewed"(*2023 Tripadvisor Review Transparency Report*).

Fake reviews serve several purposes, from review boosting to review vandalism. Spotting them is crucial. In 2022, Tripadvisor spotted more than 1.3 million fake reviews, whereas Amazon blocked more than 200 million reviews in the same year (*3 steps Amazon is taking to stop fake reviews*; *2023 Tripadvisor Review Transparency Report*). On its side, Google has removed more than 170 million policy-violating reviews and 12 million fake businesses, which was reported as an increase of 45% compared to 2022.

Therefore, businesses must consider fake reviews, especially when their core decision is based on information derived from them. Review algorithm checkers, moderators, artificial intelligence, and review helpfulness have been tools against fake reviews, but improvements still must be made (*3 steps Amazon is taking to stop fake reviews*).

Appendix 1 - Fake reviews

3. Artificial Intelligence

The word "Intelligence" is defined by the Cambridge English Dictionary as "The ability to learn, understand, and make judgments or have opinions that are based on reason", whereas "Artificial" is defined as "Made by people, often as a copy of something natural"(C. E. Dictionary s.d.). Therefore, we can attempt to define artificial intelligence as a machine that can act like intelligent beings. This suggests a machine that can learn, understand, think and make decisions (Negnevitsky 20).

Throughout History, various definitions based on the dimension of thought and actions have been proposed for AI. These definitions are presented in Figure 3 as follows:

Thinking Humanly “The exciting new effort to make comput- ers think . . . *machines with minds*, in the full and literal sense.” (Haugeland, 1985) “The automation of activities that we associate with human thinking, activities such as decision-making, problem solv- ing, learning . . .” (Bellman, 1978)	**Thinking Rationally** “The study of mental faculties through the use of computational models.” (Charniak and McDermott, 1985) “The study of the computations that make it possible to perceive, reason, and act.” (Winston, 1992)
Acting Humanly “The art of creating machines that per- form functions that require intelligence when performed by people.” (Kurzweil, 1990) “The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991)	**Acting Rationally** “Computational Intelligence is the study of the design of intelligent agents.” (Poole *et al.*, 1998) “AI ...is concerned with intelligent be- havior in artifacts.” (Nilsson, 1998)

Figure 3 - Definitions of AI Throughout History (Russell, Norvig 2016)

Nowadays, AI is commonly regarded as the field of computer science and engineering focused on creating systems capable of performing tasks that typically require human intelligence (Negnevitsky 20). The question of 'Can machines think?' is a highly complex one. In 1950, ***Alan Turing*** introduced the concept of ***''The Imitation Game''***, to define and explore whether machines can act in a way that is indistinguishable from humans (Turing 2009).

The Imitation game has been ingrained as the **Turing Test**. To pass the **total Turing Test,** the system needs to possess the following capabilities :

- **natural language processing (NLP):** to communicate successfully with human languages.
- **knowledge representation (KR):** to store what it knows or hears.
- **automated reasoning (AR):** to use the stored information to answer questions and draw new conclusions.
- **machine learning (ML):** to adapt to new circumstances and to detect and extrapolate patterns.
- **computer vision (CV):** to perceive objects.
- **robotics:** to manipulate objects and move about.

(Turing 2009; Russell, Norvig 2016)

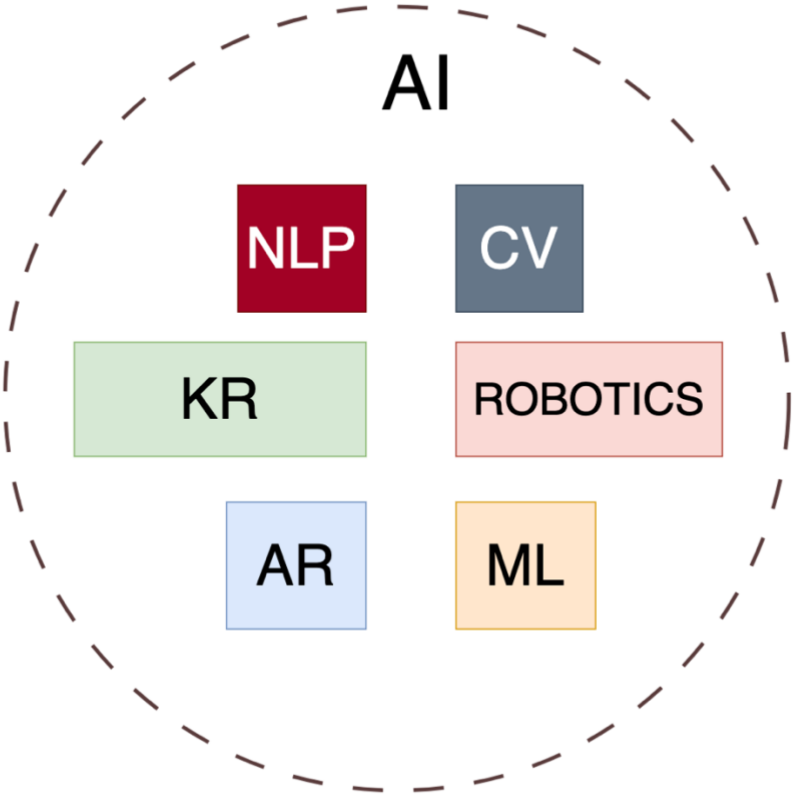


Figure 4 - AI Tools (By the author)

In our quest to derive information from millions of reviews, we require an AI tool capable of understanding human language (as complex as it can be) and classifying the sentiment each review radiates. This is the realm of Natural Language Processing (Tejwani 2014).

3.1 Natural Language Processing

Natural language processing (NLP) is a field of science and engineering that develops and studies automatic systems that understand and generate natural languages. NLP combines the principles of linguistics and engineering methods.

Human languages enable us to share and store complex information, and the complexity of human language is unique among species (Manning 2022).

Although it has become usual nowadays to interact with a computational system in human language so that the machine can understand us and we can comprehend the machine's response, it is a staggering advancement because we are still nowhere close to developing learning machines that have a fraction of the acquisition ability of children to learn languages (Manning 2022).

3.1.1 NLP Tasks

NLP techniques are used for a multitude of tasks, such as :

- **Machine Translation:** Automatically converting text from one language to another.
- **Question Answering:** Building systems that automatically answer questions humans pose in natural language. (Search engine, virtual assistant)
- **Text Summarisation:** Creating summaries from large texts while preserving the most essential information.
- **Named Entity Recognition (NER):** Identifying and classifying entities within text into categories such as names of people, organisations, locations, and dates.
- **Sentiment Analysis:** Determining the emotional tone behind a body of text.

And much more ...(Ranjan et al. 2016)

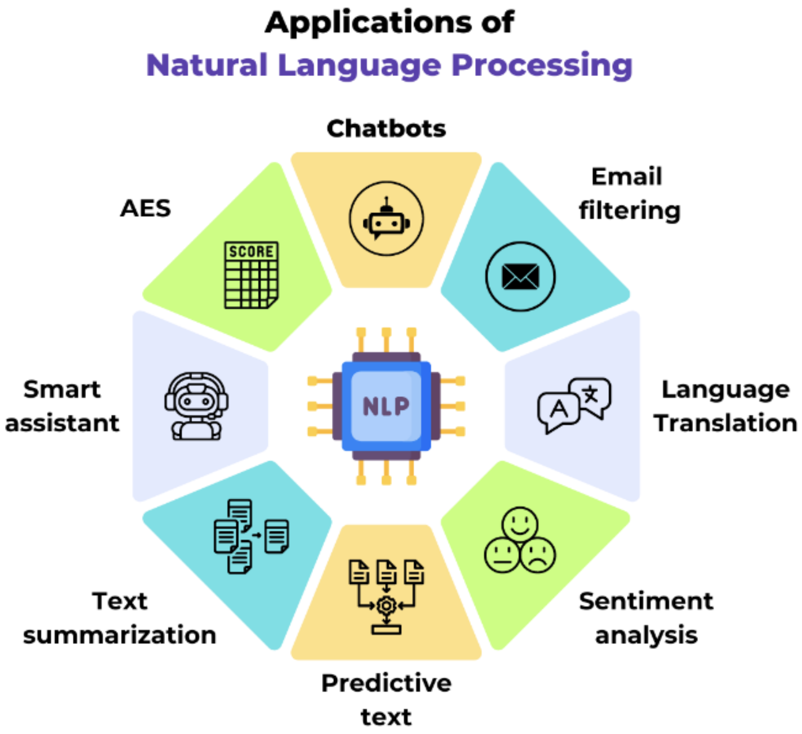


Figure 5 - NLP Applications(Quan 2023)

One fundamental problem in building language-learning machines is the question of representation: How should we represent language in a computer so that the computer can process it and generate a response? Attempts to solve this quest are diverse and have evolved throughout the years (Manning 2022).

3.1.2 Word Representation

3.1.2.1 Denotational Semantics

One attempt was to represent each word by all the things it represents, also known as **denotational semantics**. The idea is to feed the computer with all instances or objects a word denotes. For instance, the word "drink" will be represented by all instances that refer to a drink:

Drink = {Tea, Water, Coffe, Juice...}

Figure 6 - Denotational semantics (Engineering s.d.)

The idea was that a computer could "understand" and generate meaning based on these associations by compiling comprehensive lists of possible referents for each word. Unfortunately, natural language is highly context-dependent, with words and sentences often having multiple meanings that change based on context. Hence, denotational semantics struggles with this variability and the dynamic nature of language.

3.1.3 Independent Vector

Another attempt was to represent each word as a distinct symbol, with a one-hot vector representing each word**.** For instance, the words "machine" and "computer" can be represented as follows :

if 𝕍machine the vector representing the word "machine " and 𝕍computer representing the word "computer" in a corpus we have:

$$\mathbb{V}\text{machine} \quad = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

$$\mathbb{V}\text{computer} \; = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

Figure 7 - One-hot vectors

Word representations as unique and independent vectors provide better computation and allow the machine to differentiate each word. However, the problem with one-hot vectors is that there is no notion of similarity and relationship between words. Furthermore, with roughly one million words in the English vocabulary, we would require a million-dimensional matrix to represent our entire corpus. This would be too expensive and time-consuming (Manning 2022).

3.1.4 Distributional Semantics (Word Embedding)

The idea of distributional semantics goes back to 1957, when the British linguist **John Rupert Firth** famously said, "*You shall know a word by the company it keeps*" (Firth 1957).The principal idea is that all the words that appear in its context (all the words around it) give a word's meaning. This is the most successful idea in modern NLP, argues (Manning 2022). Since, distributional semantics has given birth to fantastic research such as :

- **Word2Vec**: A model that generates dense word embeddings based on the context of words.(An example of an embedding space in Figure 8)
- **GloVe (Global Vectors for Word Representation)**: An approach that creates word vectors by leveraging global word-word co-occurrence statistics.
- **Term Frequency–Inverse Document Frequency (TF-IDF):** A statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus). It helps identify key terms that distinguish documents within a more extensive set.
- **FastText**: An extension of Word2Vec that includes subword information, enhancing its ability to handle morphologically rich languages.
- **Self-Attention and Attention Mechanism:** An advanced approach that allows models to focus on different parts of a text sequence when encoding it, capturing contextual relationships more effectively. This mechanism is a fundamental component of Transformer architectures, enabling the modelling of dependencies across long distances in a text.

These advancements have revolutionised NLP, enabling a more nuanced and accurate language understanding (Manning 2022).

Word embedding captures more semantic properties by mapping words with similar meanings in the embedding space. All the words in a corpus are represented by dense vectors with similar words positioned closer together.

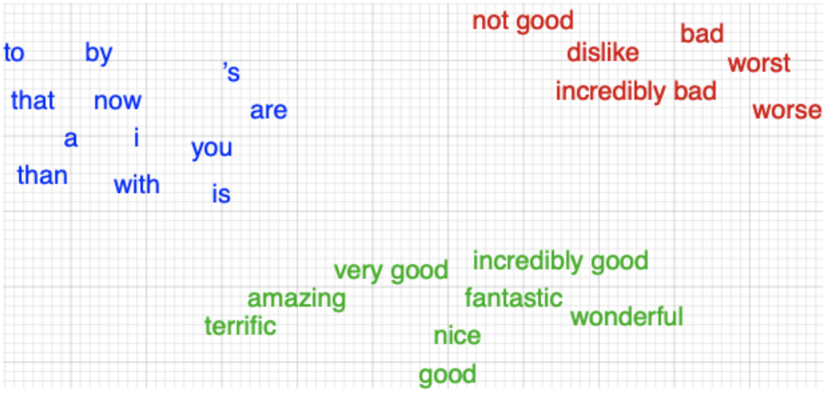


Figure 8 - An embedding space (Jurafsky, Martin 2000)

3.2 Sentiment Analysis

Sentiment analysis is the preferred technique for text classification. With sentiment analysis, we can derive information from customers' reviews by classifying the underlying sentiment as positive or negative. It is a binary classification or multiclassification task, meaning there are only two or three possible outcomes: ***"0"*** can be related to a ***negative sentiment***, ***"1"*** to a ***positive sentiment***, and everything in between or so as ***neutral*** (Tejwani 2014)**.**

Several approaches can be used for sentiment analysis tasks. We can note :

- **Rule-based or Lexicons-based methods:**

The lexicons-based methods proposed by linguists use a list of words and phrases (lexicons) that are linked to different sorts of emotion to classify words and detect sentiment (*Lexicon-based sentiment analysis: What it is & how to conduct one | KNIME*)

For example, words like "happy" and" joy" could be associated with positive sentiment, while "sad" and" bad" might be associated with negative sentiment.

- **Machine learning approach :**

It involves using supervised classification algorithms to recognise text sentiments. Common algorithms include :

- ***Naive Bayes:*** A probabilistic classifier based on Bayes' Theorem with strong (naive) independence assumptions between features.

- ***Support Vector Machines (SVM):*** A linear classifier that finds the optimal hyperplane which maximises the margin between different classes in the feature space.

- ***Logistic regression:*** A statistical model that uses a logistic function to model the probability of a binary outcome.

- ***Decision Trees:*** A Decision tree splits the data based on feature values to make predictions.

(Monika et al. 2022; Boiy, Moens 2009)

- **Deep learning approach:**

Deep learning uses neural networks to model and understand complex patterns and relationships in data. **Neural networks (NN)** are inspired by the brain's computation mechanism, which consists of neurons exchanging signals. Deep learning has recently gained much attention as it has outperformed classical linear models. Deep learning models can capture complex, non-linear relationships in data. Additionally, deep learning models can successfully learn from unlabelled data through unsupervised and semi-supervised learning techniques (Goldberg 2016). Figure 9 displays the architecture of a simple feed-forward neural network.

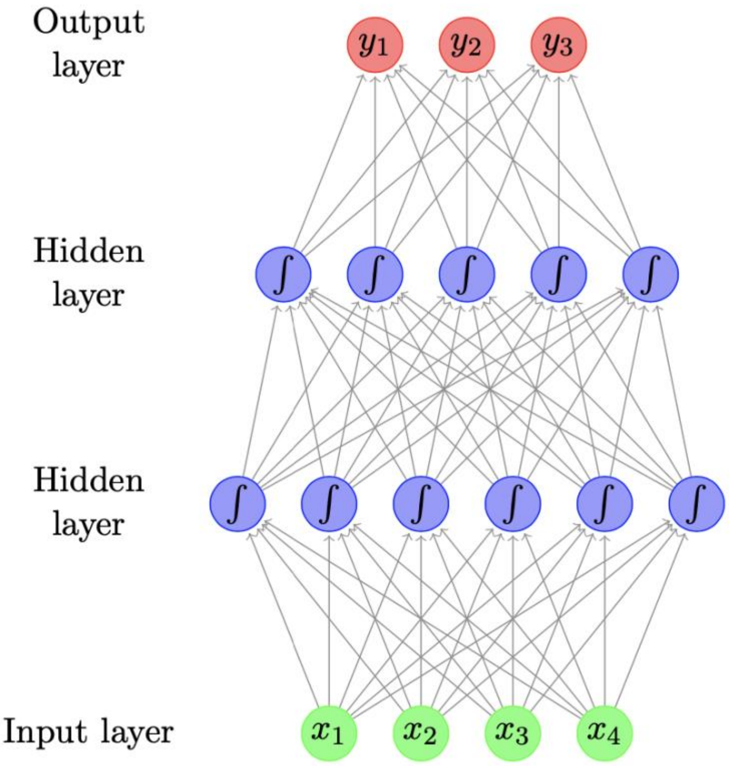


Figure 9 - Feed Forward Neural network architecture (Goldberg 2016)

A simple way to think about a neural network (NN) is as a complex calculus problem where we aim to minimise the loss function with optimisation techniques like **Gradient Descent (GD)** or **Stochastic Gradient Descent (SGD)**. The core mechanism that enables this optimisation in neural networks is **backpropagation** (Nielsen).

Backpropagation involves computing the gradient of the loss function for each weight and bias in the network. These gradients are partial derivatives that indicate adjusting each parameter to decrease the loss. By updating the weights and biases based on these gradients, the network learns to improve its predictions (Nielsen).

The loss function measures how well the neural network's predictions match the outcomes. In the case of a classification task like sentiment analysis, the most commonly used loss function is the **Cross-Entropy Loss** (Nielsen)**.** The Cross-entropy Loss measures the performance of a classification model whose output is a probability value between 0 and 1 given by the **softmax function.** There are :

Binary Cross-Entropy Loss (Log Loss): For binary classification, where the target variable can take on two values (0 or 1):

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i\log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$$

Where:

- N is the number of samples.
- yi is the true label for the i-th sample (0 or 1).
- $\hat{y}$i is the predicted probability of the positive class for the i-th sample.

Cross-Entropy Loss for Multi-Class Classification: For multi-class classification, where the target variable can take on one of *C* different classes, the cross-entropy loss is often called **Categorical Cross-Entropy Loss**.

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c})$$

Where:

- *N* is the number of samples.
- *C* is the number of classes.

- $\mathcal{Y}$i,c is a binary indicator (0 or 1) if class label c is the correct classification for sample i.
- $\hat{\mathcal{Y}}i$,c is the predicted probability of sample i being in class c.

Gradient Descent (GD) and **Stochastic Gradient Descent (SGD)**

- **GD** is an optimisation algorithm that adjusts the network's weights and biases to minimise the loss function. It computes the loss gradient (partial derivatives) for all parameters and updates them in the direction that reduces the loss.
- **SGD** is a variant of GD that updates the parameters using only a subset (mini-batch) of the data at each iteration, making it more efficient for large datasets.

The SGD is more commonly used as the GD is too computationally expensive. In NLP, we often work with billions of words in a corpus; it would take forever to update the entire corpus. Despite the noise it causes, SGD helps update the function (Manning 2022).

The general formula for GD follows:

$$\theta := \theta - \alpha \nabla J(\theta)$$

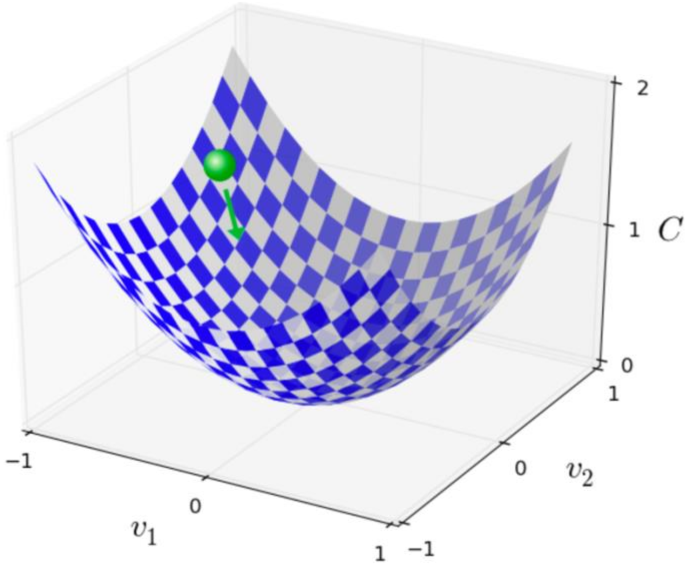


Figure 10 - Gradient Descent (Nielsen)

Standard deep learning models for sentiment analysis include :

- ***Recurrent Neural Network (RNN):*** RNNs are designed to handle sequential data by maintaining a memory of previous inputs. However, they can struggle with long sequences due to vanishing gradients (Zhang, Wang, Liu 2018).

- ***Long Short-Term Memory(LSTM):*** LSTMs are a particular type of RNN that addresses the limitations of standard RNNs, such as the vanishing gradient problem. They use gates to control the flow of information, making them effective at learning long-term dependencies (Zhang, Wang, Liu 2018).

- ***Transformer models:*** Transformer models use self-attention mechanisms to process input sequences in parallel rather than sequentially, making them efficient and effective at capturing long-range dependencies. They have become the foundation for many advanced NLP models, such as BERT and GPT (Pathak et al. 2020).

3.3 Recurrent Neural Networks (RNNs)

RNNs are a class of NN designed to process sequential data, such as time series or text (*What are Recurrent Neural Networks? | IBM* 2021). RNNs are called recurrent because they perform the same task for every sequence element, with the output dependent on the previous computations. Another way to think about RNNs is that they have a "memory" which captures information about what has been calculated so far (Nabi 2019). In a classic NN, all the inputs enter the network simultaneously, whereas RNN processes inputs sequentially, one step at a time. The output from each step is used as part of the input for the next step, allowing the network to maintain a form of memory of previous inputs (*Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville - Google Books*).

A simple representation of an RNN follows:

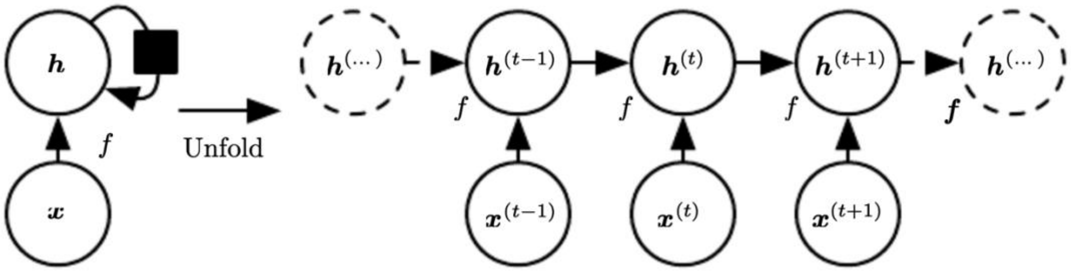


Figure 11 - Simple RNN (*Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville - Google Books*)

There are several RNN architectures. For sentiment analysis, the **Many to One** architecture is commonly used. The RNN take in a sequence of data, such as a sentence, a comment, or a review, then processes the entire sequence to generate a single output value (positive/ negative sentiment) (Sachinsoni 2024)

A Many to One RNN is represented as follows:

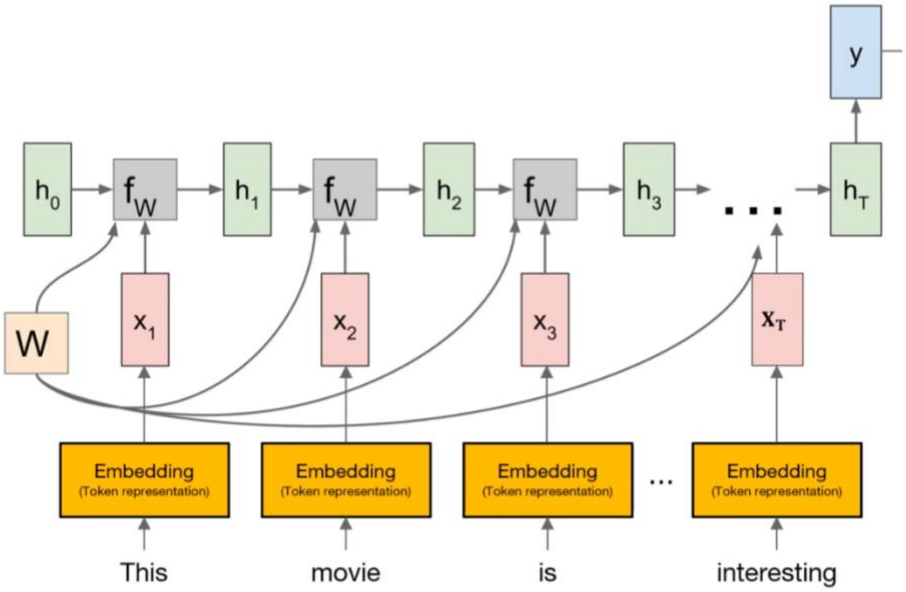

y
h0
fW
h1
fW
h2
fW
h3
. . .
hT
W
x1
x2
x3
xT
Embedding (Token representation)
Embedding (Token representation)
Embedding (Token representation)
...
Embedding (Token representation)
This
movie
is
interesting

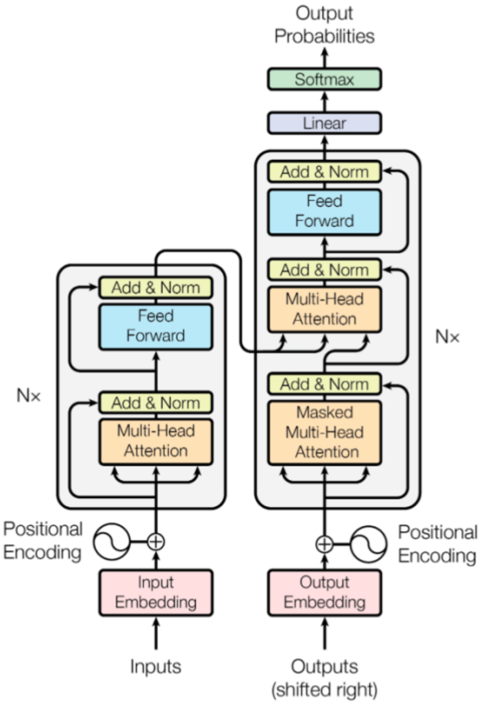
Figure 12 - Many to One RNN (Kang 2020)

One standard issue with RNN is its short-term memory, which causes the network to have difficulty remembering information from the beginning of long sentences. As new inputs are processed, the influence of earlier inputs can diminish, making it challenging for the network to retain important information from the start of the sequence (***Vanishing Gradient Problem***). Advanced RNN variants like Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRUs) and Bidirectional RNNs have been developed to address this issue and many others. These architectures include mechanisms to better retain information over long sequences and handle gradient issues more effectively (*Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville - Google Books*)

3.4 Transformers

Transformers are a type of neural network architecture. Proposed in 2017 by Google engineers and researchers from the University of Toronto, Transformers are considered the most crucial breakthrough in artificial intelligence, especially in deep learning, NLP and Computer vision (*What are Transformers? - Transformers in Artificial Intelligence Explained - AWS*).

The initial transformer architecture was proposed to solve neural machine translation problems and any task that transforms an input sequence into an output sequence (Vaswani et al. 2023).



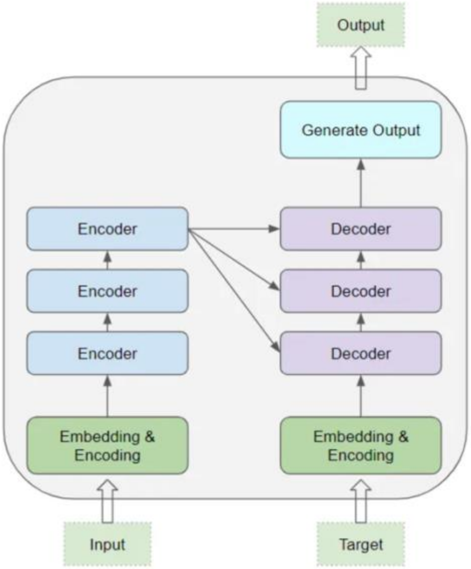


Figure 13 - Initial Transformer Architecture

(Vaswani et al. 2023)

Figure 14 - Also a Transformer

(Doshi 2021)

To give some intuition, this is how the initial transformer will translate a sentence (Doshi 2021):

- **The input embedding:** convert the sentence into vectors
- **The positional encoding:** add the position of each word to the embedding

- **The encoder:** uses the attention mechanism to capture context and pass the result through a feed-forward neural network (FNN) to add additional transformation and non-linearity to the vectors.
- **The decoder:** generates the translation by focusing on the encoded representation of the entire input sentence. Then, it uses the first generated word to help generate the next, again referring to the encoded input and considering the previously generated word.

Compared to RNN, Transformers can deal with long-range dependencies between words in long sentences and are not limited by the time-step computation(one word at a time), which slows the training process. Thanks to **Attention mechanisms** at the core of the process, enabling them to capture context and nuances, Transformers represent a significant quantum leap in text processing and language understanding (Doshi 2021). Since then, numerous projects, including Google's **Bidirectional Encoder Representations from Transformers (BERT)** and OpenAI's **Generative Pre-trained Transformer (GPT)**, have been built on its foundation and have reached unprecedented performance (Gupta 2024).

For a classification task like sentiment analysis, a classic Transformer architecture would look like this:

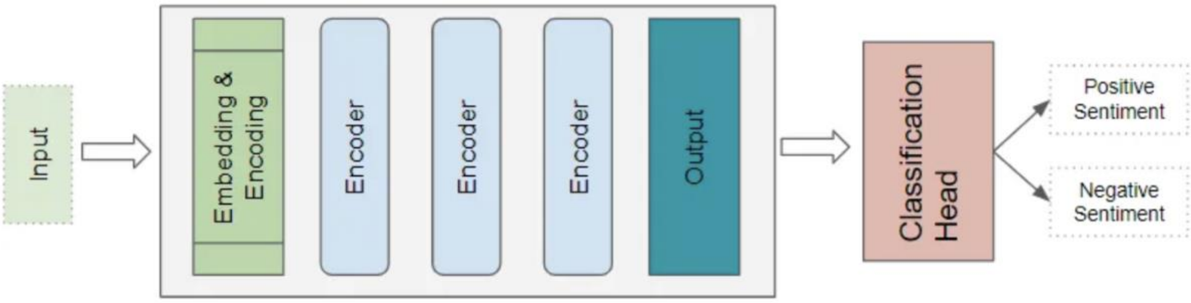


Figure 15 - Transformer for sentiment analysis (Doshi 2021)

3.5 Attention, Self-attention And Multi-Head Attention

Attention Mechanism

In the encoder-decoder architecture, encoders build a representation of the input, which the decoder uses to generate the target text. The final hidden state of the encoder layers acts as a **bottleneck (static context vector with a fixed size)**, representing everything about the meaning of the input. Since all the decoder knows about the input is the information presented by the **bottleneck,** information at the beginning of the input sentence, especially for long sentences, may not be equally represented in the last encoder layer. This is where **attention mechanism** intervenes (Jurafsky, Martin 2000)

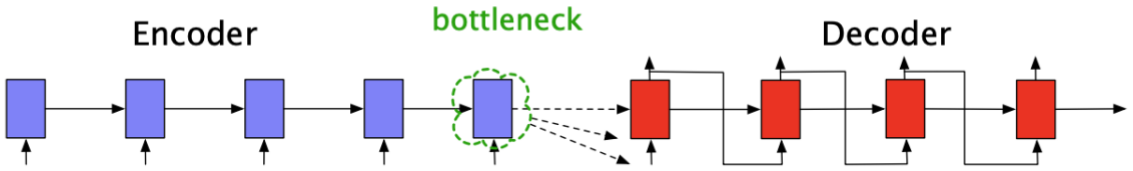


Figure 16 - Encoder-Decoder Bottleneck (Jurafsky, Martin 2000)

The attention mechanism helps the decoder get information from all the hidden states of the encoder layers and not just the last one (Jurafsky, Martin 2000). Introduced by (Bahdanau, Cho, Bengio 2016) to leverage RNNs, the idea suggested that not only should all input words be taken into account in the context vector, but each should also be given relative importance or **Attention**.

The following example helps build intuition: Imagine someone chatting in a crowded, noisy room like a bar. Notwithstanding the noise, the brain will focus on the sound from the exchange while filtering any background distraction. This is what ***attention*** is doing by attributing different levels of importance to each input (Malingan 2023).

This other example also helps build intuition (Bishop, Bishop 2023):

Consider the two sentences :

I swam across the river to get to the other bank.

I walked across the road to get cash from the bank.

When inputting these sequences into the network, the word "*bank*" is represented by the same vector in both sentences after the embedding. However, the appropriate interpretation can only be detected by examining the context. We also see that some words help us better detect the meaning of "*bank*" than others.

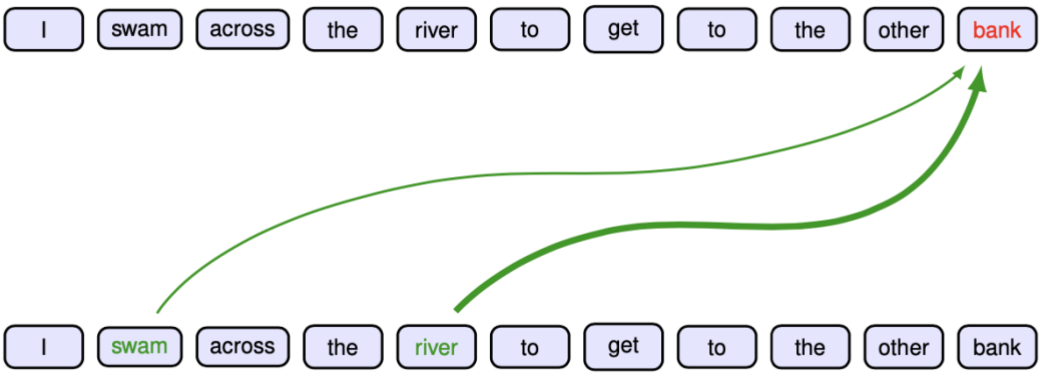


Figure 17 - Schematic illustration of attention in which the interpretation of the word 'bank' is influenced by the words 'river' and 'swam' (Bishop, Bishop 2023)

Therefore, to accurately interpret *"bank,"* the network should ***attend to*** or rely more heavily on specific words.

Appendix 2 - How the encoder-decoder attention mechanism works

Self-Attention Mechanism

The transformer model comprises many layers, each building a representation of the input based on the representation of previous layers. Self-attention, the key innovation in transformer architecture, allows the network to build a richer representation of the input based on the representation of the last layer by capturing more context and relationships between words (Jurafsky, Martin 2000).

Where the encoder-decoder Attention allows the decoder to focus on different parts of a sequence while generating the output, Self-attention focuses only on the relationships and dependencies within a sequence to build a richer representation of the same sequence (*Self attention vs attention in transformers | Medium*).

The following example helps build intuition

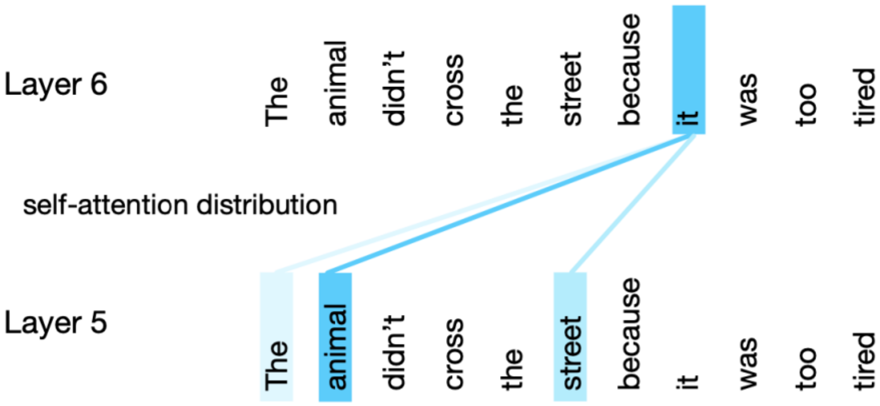


Figure 18 - A Self-attention distribution (Jurafsky, Martin 2000)

In **Figure 18**, the Self-attention layer 6 builds representation for the word *"it"*. In computing its representation, the Self-attention mechanism attends differently to the various words at layer 5, with darker shades indicating higher self-attention values.

Self-attention aims to update the vector of every word in the input sequence. Thus, the embedding better represents the context in which the word is used (Smith 2024). **Figure 19** showcases how the embedding of the word "bank" does not relate to any particular context but shifts as the input " man fishing on a river bank." passes through the self-attention layer.

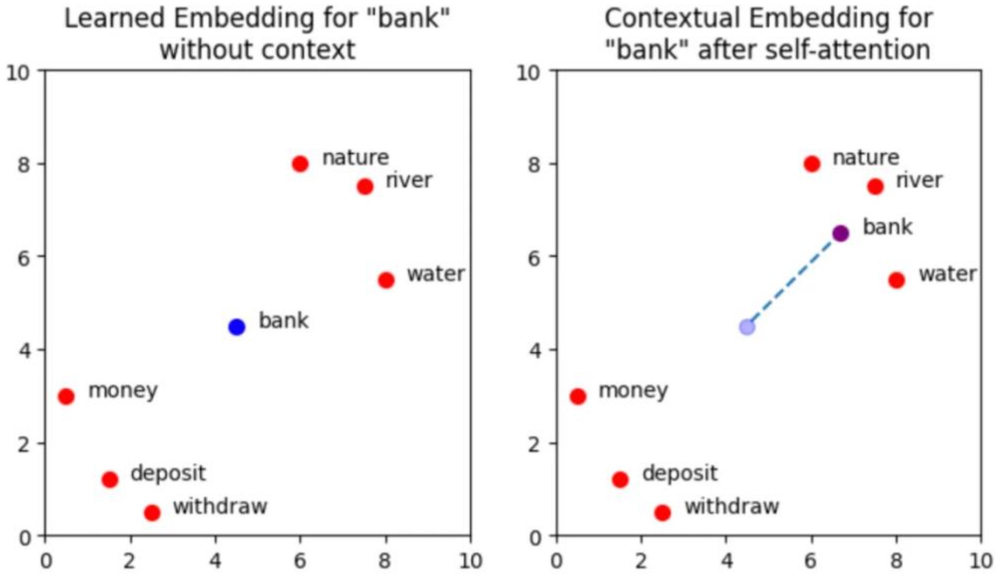


Figure 19 - After Self-attention (Smith 2024)

The self-attention layer can be implemented differently, as shown in F**igure 20**.

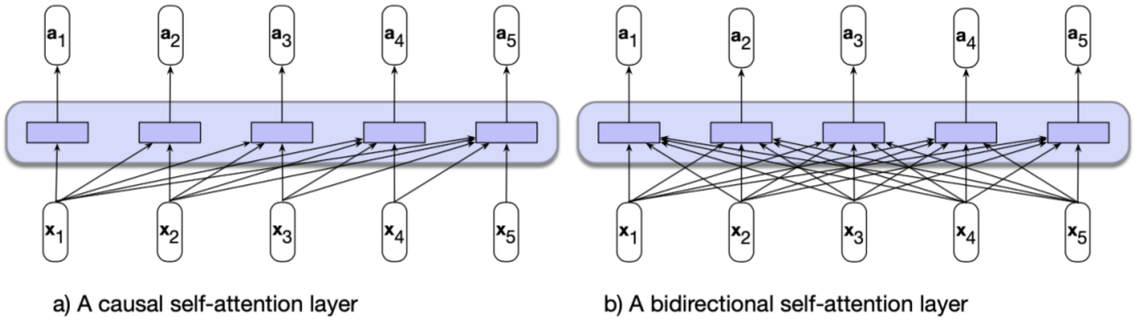

a₁ a₂ a₃ a₄ a₅
x₁ x₂ x₃ x₄ x₅
a) A causal self-attention layer
b) A bidirectional self-attention layer

Figure 20 - Casual self-attention and Bidirectional self-attention (Jurafsky, Martin 2000)

In a casual self-attention layer, each token only attends to the tokens that precede it in the input sequence, thus ignoring any tokens that come after and preventing the model from accessing the future tokens while making the prediction. Casual self-attention is commonly used in **Natural Language Generation** (NLG) and is suitable for tasks like text generation or machine translation. By giving each token of an input sequence the full context of the sequence (preceding or succeeding tokens), the Bidirectional self-attention layer ensures a deep understanding of the input sequence suitable for **Natural Language Understanding (NLU)** tasks such as sentiment analysis or question-answering (Jurafsky, Martin 2000).

When the input embedding gets into the self-attention layer, it passes the following steps:

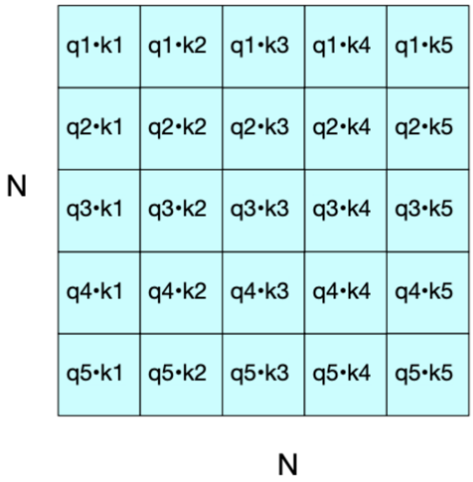
- First, the input embedding is updated by adding information about the position of each sequence token through **positional encoding.**

- The updated embedding input is then used to generate three different matrices through a learned Query, Key, and Value weight matrices :

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{\mathbf{Q}};\ \ \mathbf{K} = \mathbf{X}\mathbf{W}^{\mathbf{K}};\ \ \mathbf{V} = \mathbf{X}\mathbf{W}^{\mathbf{V}}$$

Figure 21 - Query, Key and Value learned matrices (Jurafsky, Martin 2000)

- Hence, attention scores are computed using the dot product of the Key and Query matrices as a notion of similarity between each input vector, including the vector himself. Figure 22 showcases the attention scores of an input sequence of 5 words:



	q1•k1	q1•k2	q1•k3	q1•k4	q1•k5
	q2•k1	q2•k2	q2•k3	q2•k4	q2•k5
N	q3•k1	q3•k2	q3•k3	q3•k4	q3•k5
	q4•k1	q4•k2	q4•k3	q4•k4	q4•k5
	q5•k1	q5•k2	q5•k3	q5•k4	q5•k5

Figure 22 - Attention scores of a 5 inputs sequence (Jurafsky, Martin 2000)

- The dimensionality of the Key matrix then scales the attention scores to prevent the dot product from becoming too large, which can make the gradient unstable.

- The softmax function is then used to normalise the scores so that the sum is 1 across each sequence.

- A weighted sum of the attention weights applied to the Value matrix gives the final attention output, which is called the output of a single attention head.

The following formula summarises the whole process:

$$\text{SelfAttention}(\mathbf{Q},\mathbf{K},\mathbf{V}) \;=\; \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathsf{T}}}{\sqrt{d_k}}\right)\mathbf{V}$$

Figure 23 - Self-attention formula (Vaswani et al. 2023)

Padding Mask

In NLP, input text can vary in length, which poses a challenge for batch processing, which requires all inputs to have the same dimensions. To address this, sequences are often padded to a uniform length, and padding masks are used to ensure that the padding tokens do not influence the model's learning process. Hence, at each batch, each sequence is padded to the length of the longest sequence in the batch (Mongaras 2022).

This is handled by ensuring the padding tokens do not contribute to the attention score computation. To do so, the attention scores for padding tokens are set to negative infinity, which, scaled with the softmax, gives a score of 0, effectively masking them out (Mongaras 2022).

Figure 24 highlights the attention scores matrix with padding masked out.

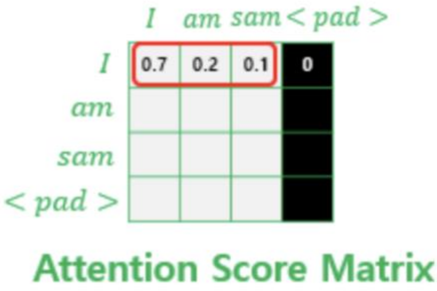
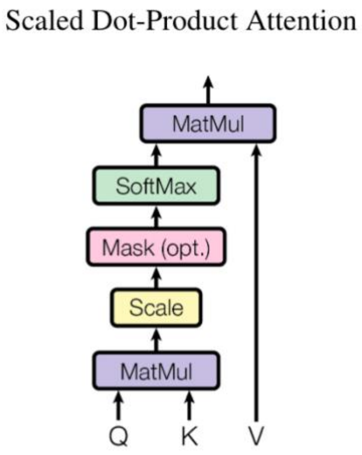


Figure 24 - Padding masked example (*마스킹| 패딩 마스크(Padding Mask), 룩어헤드 마스킹(Look-ahead masking)*)

Multi-Head Attention

The Multi-head attention layer is a set of multiple self-attention heads run in parallel to provide multiple views of the same input sequence to the model. With multiple self-attention heads, the model can attend to different parts of the sequence simultaneously, thus capturing various types of dependencies between each token, conveying a richer and more nuanced representation of the input sequence (Jurafsky, Martin 2000).

To do so, the multi-head attention follows the same process of a self-attention head with different weight inputs for each head. All the outputs are then concatenated and projected with a learnable weight matrix to produce the final matrix (Jurafsky, Martin 2000).



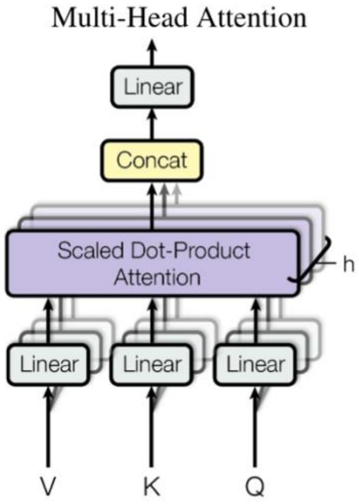


Figure 25 - A self-attention head (Vaswani et al. 2023)

Figure 26 - Multi-head attention (Vaswani et al. 2023)

The Add & Normalise layer

Error grows in a deep neuronal network as it gets very deep. This is because features in the later layers are remotely related to the inputs, leading to a vanishing gradient problem. To address this issue, the inputs bypass one or more layers (**Skip connection**) and are added to the add layer of the add&normalise block. This process is called **Residual connection** (*Demystifying the Add & Norm Block in the Transformer Neural Network Architecture: With Code*)**.** This increases performance in the learning process.

While training a deep neuronal network, the distribution of each input layer changes as the parameters of the previous layers are updated, which can slow the training. This is known as the **Internal covariate shift**, and it causes the gradients to become very large or slow as they propagate through the network. The normalisation layer of the add&normalise block troubleshoots this issue by ensuring that each layer has a more consistent distribution (by ensuring the mean equal to 0 and the standard deviation equal to 1) (*Demystifying the Add & Norm Block in the Transformer Neural Network Architecture: With Code*).

Hence, the Add & Normalise layer is essential for stabilising the training process and improving the performance of deep neural networks. Combining residual connections and layer normalisation helps the model learn more effectively and efficiently.

4. Simulation

In this simulation, we imagine a private investment firm commissioned us to develop a sentiment analysis model. The firm aims to analyse feedback from a selected population to inform their investment decisions regarding the advertisement budget for upcoming movie releases. The goal is to leverage the sentiment of customer reviews to determine which movie advertisements should receive the most investment.

By analysing the sentiment of these reviews, the firm hopes to identify which movies will likely be well-received by the audience. This insight will help them allocate their advertising budget more effectively, focusing on the movies with the highest potential for success. Our task is to create a robust sentiment analysis model that can accurately capture and interpret the sentiment expressed in customer reviews. This model will provide the investment firm with insights, enabling them to make data-driven decisions that maximise their return on investment.

5. Methodology

To implement our sentiment analysis model, we will go by two approaches. The first approach involves creating our model from scratch to understand everything happening in the network. In contrast, the second approach involves using a technique that has become increasingly widespread in modern AI: fine-tuning. In both approaches, we will always go through the following steps :

- Data pre-processing
- Embedding
- Model building
- Training and Validation.

A GitHub repository has been created for this occasion and is available through the following link: https://github.com/TheEternalFool/Attention_is_all_your_business_need

5.1 The Dataset

I have selected the IMDB reviews dataset from Kaggle to train our model. The dataset, comprising 50,000 equally distributed reviews, is well-known for binary classification tasks.

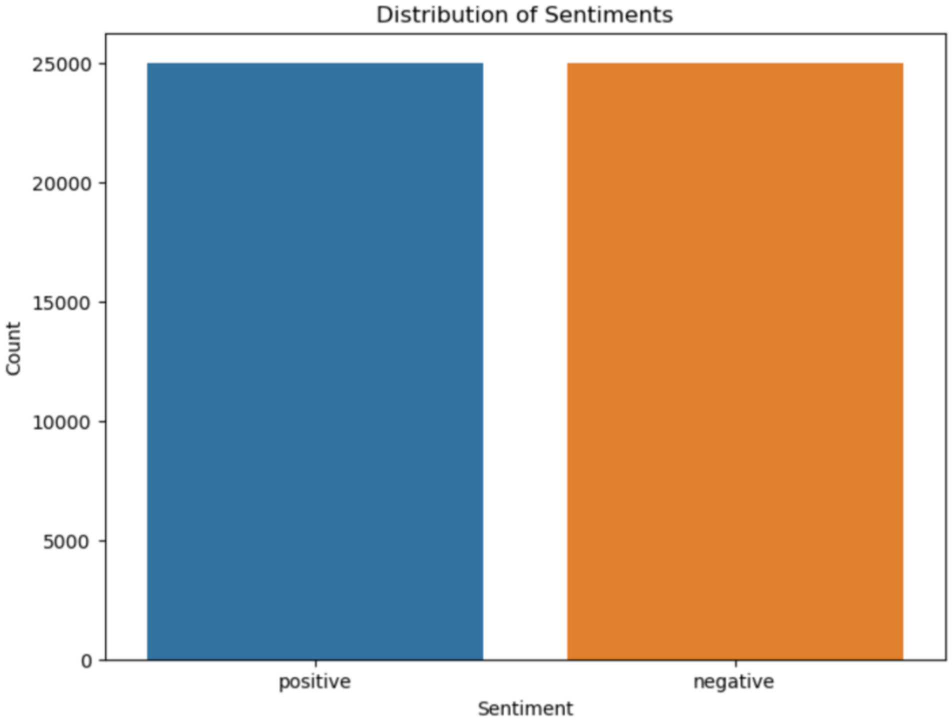


Figure 27 - IMDB reviews Description

5.2 Building The Model From Scratch

To build our model from scratch, I was inspired by a transformer architecture example found online. The architecture follows:

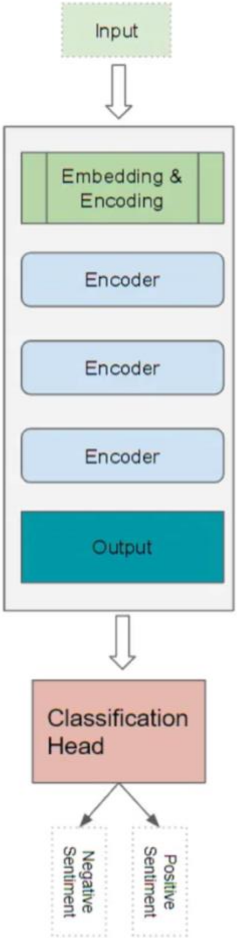


Figure 28 - Model architecture example (Doshi 2021)

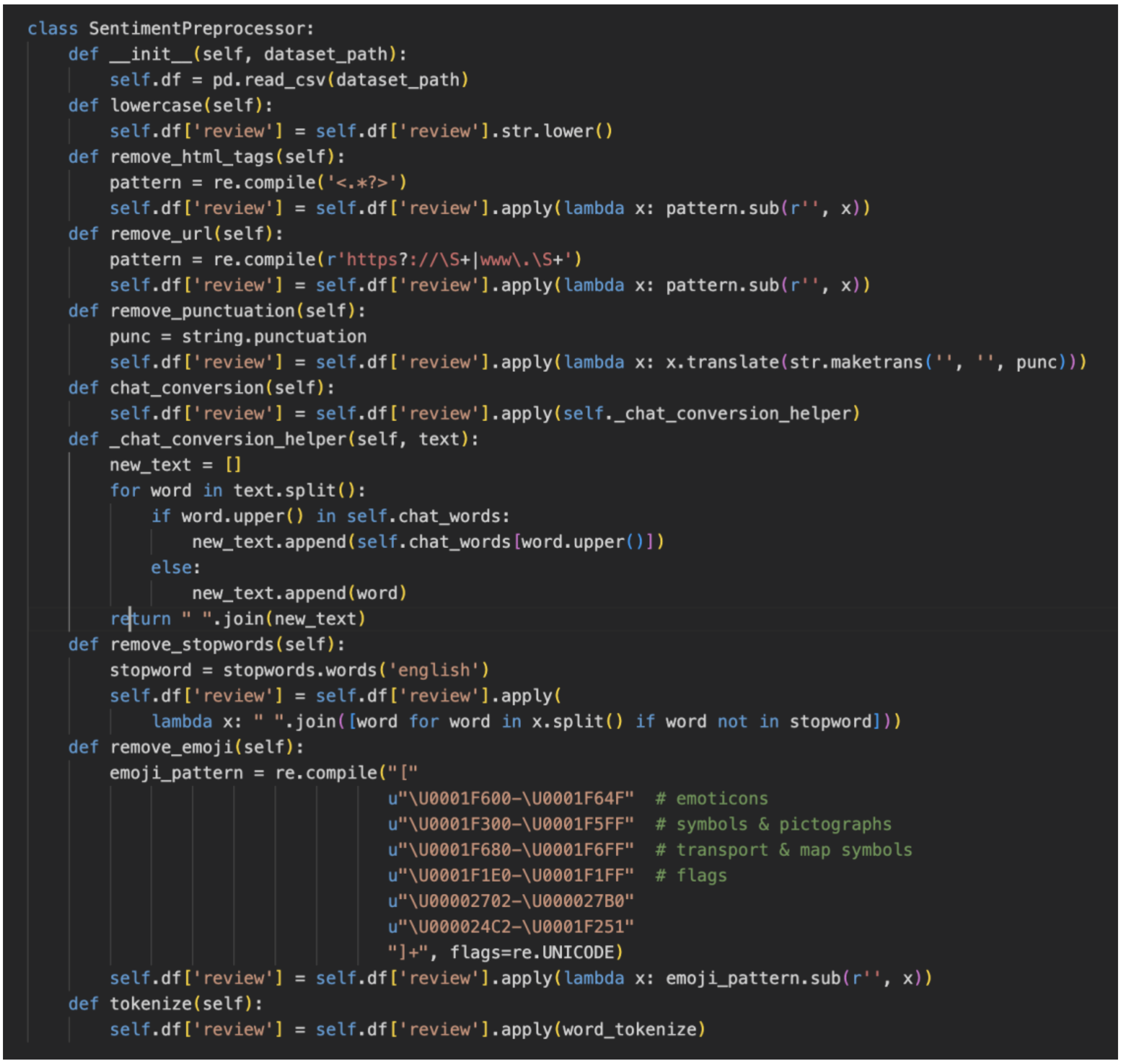
My idea is to implement the precedent architecture by creating each component using the PyTorch library, which I was introduced to in the Stanford NLP courses. By training my model with the configurations used for BERT, I aim to achieve a certain degree of performance. The model will consist of an embedding layer, a positional encoding layer, and several encoder blocks pimped together. Each encode block will comprise a multi-head attention layer, a feed-forward layer, and an add-normalise layer.

5.2.1 Data Preprocessing

Data preprocessing is a crucial step that will significantly influence the performance of our model. It consists of cleaning the raw data before training. To do so, I have created a Python class that takes the raw and goes through the following steps:

- **Text cleaning:** removing punctuation, HTML tags, and URLs and converting the whole corpus into lowercase.
- **Removing stop words:** redundant words like "the", "and", "or" are removed as they do not carry much signification.
- **Tokenisation:** splitting the corpus into unique words to constitute our vocabulary.

The code of my SentimentPreprocessor class follows:



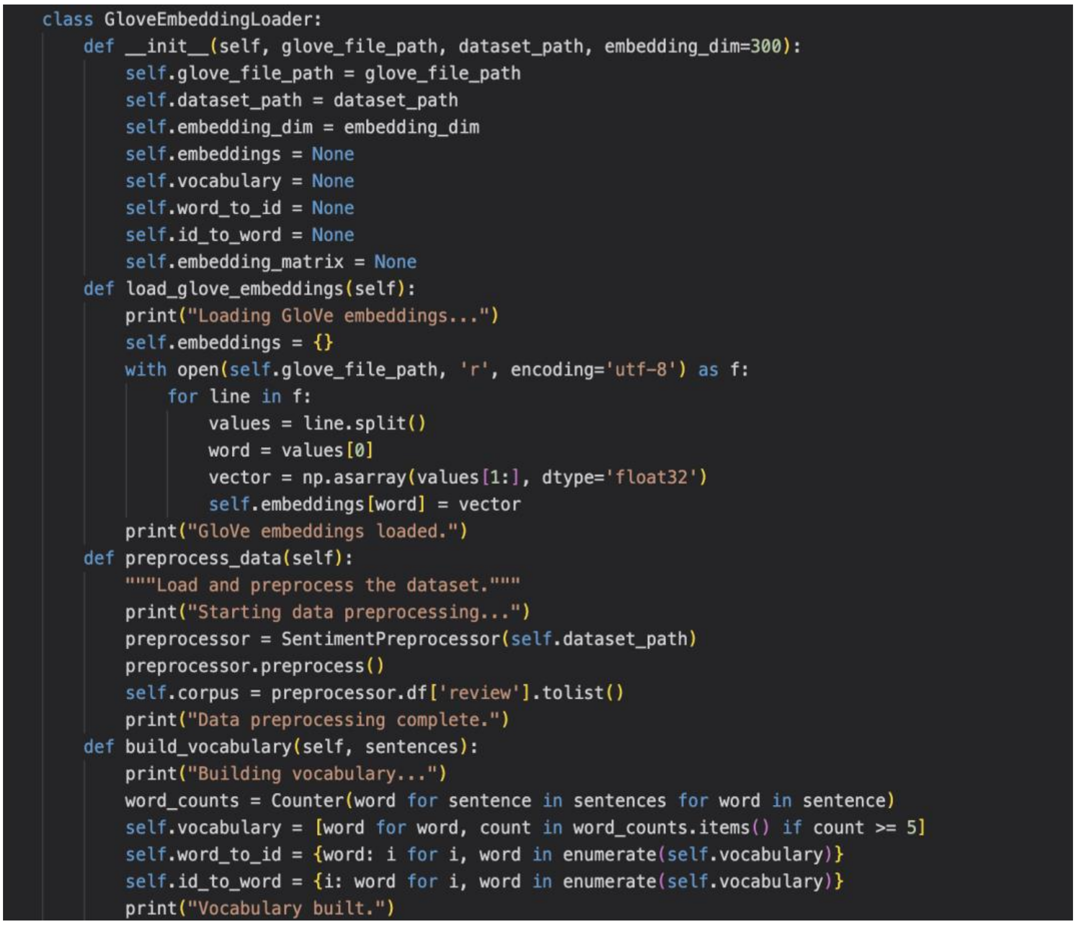
```python
class SentimentPreprocessor:
    def __init__(self, dataset_path):
        self.df = pd.read_csv(dataset_path)
    def lowercase(self):
        self.df['review'] = self.df['review'].str.lower()
    def remove_html_tags(self):
        pattern = re.compile('<.*?>')
        self.df['review'] = self.df['review'].apply(lambda x: pattern.sub(r'', x))
    def remove_url(self):
        pattern = re.compile(r'https?://\S+|www\.\S+')
        self.df['review'] = self.df['review'].apply(lambda x: pattern.sub(r'', x))
    def remove_punctuation(self):
        punc = string.punctuation
        self.df['review'] = self.df['review'].apply(lambda x: x.translate(str.maketrans('', '', punc)))
    def chat_conversion(self):
        self.df['review'] = self.df['review'].apply(self._chat_conversion_helper)
    def _chat_conversion_helper(self, text):
        new_text = []
        for word in text.split():
            if word.upper() in self.chat_words:
                new_text.append(self.chat_words[word.upper()])
            else:
                new_text.append(word)
        return " ".join(new_text)
    def remove_stopwords(self):
        stopword = stopwords.words('english')
        self.df['review'] = self.df['review'].apply(
            lambda x: " ".join([word for word in x.split() if word not in stopword]))
    def remove_emoji(self):
        emoji_pattern = re.compile("["
                                   u"\U0001F600-\U0001F64F"  # emoticons
                                   u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                                   u"\U0001F680-\U0001F6FF"  # transport & map symbols
                                   u"\U0001F1E0-\U0001F1FF"  # flags
                                   u"\U00002702-\U000027B0"
                                   u"\U000024C2-\U0001F251"
                                   "]+", flags=re.UNICODE)
        self.df['review'] = self.df['review'].apply(lambda x: emoji_pattern.sub(r'', x))
    def tokenize(self):
        self.df['review'] = self.df['review'].apply(word_tokenize)
```

Figure 29 - Model data and input pre-processing code

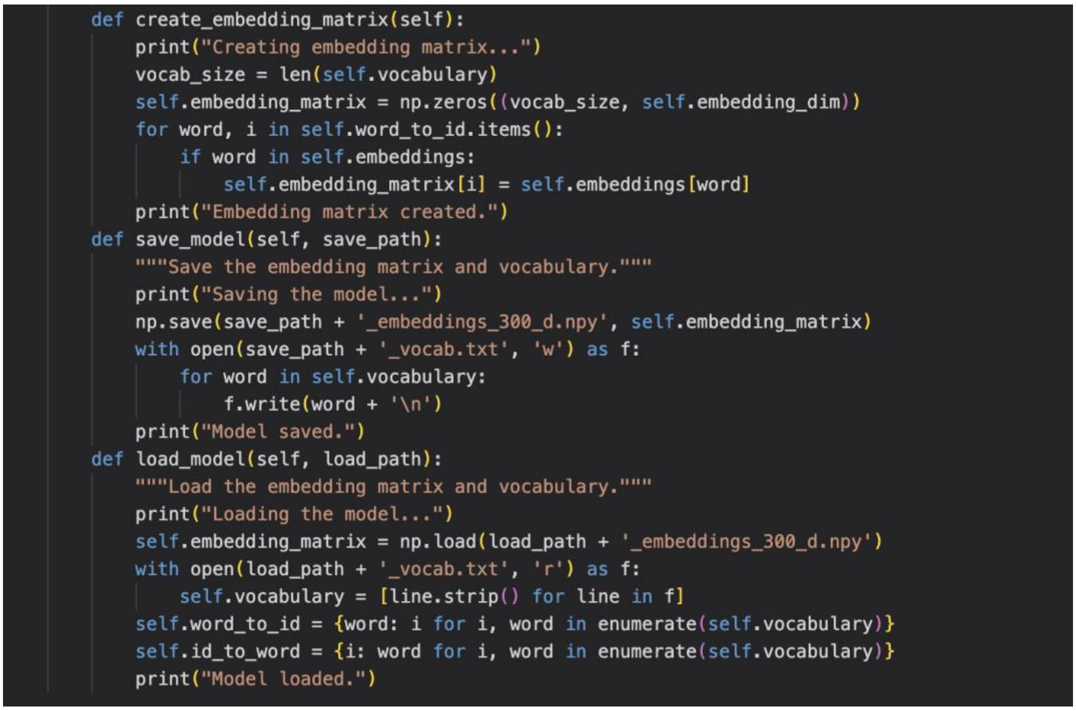
5.2.2 The Embedding

The data set now cleaned, we can proceed to create the embedding for our model. The embedding comprises the representation of all our vocabulary words in a dense vector, with each word(token) representing a specific vector at a specific position in the embedding space.

To build the embedding, I first attempted to train my own embedding model using the glove technique, but with **44351** words in my vocabulary, building a co-occurrence matrix would have required a billion loops. Therefore, I went for a more straightforward approach. The glove website provides a pre-trained model that can be mapped to a vocabulary. Thus, I downloaded the 300-dimensions glove.840B.300d of 2GO, which I have mapped to my vocabulary. The code for the embedding follows :



```
class GloveEmbeddingLoader:
    def __init__(self, glove_file_path, dataset_path, embedding_dim=300):
        self.glove_file_path = glove_file_path
        self.dataset_path = dataset_path
        self.embedding_dim = embedding_dim
        self.embeddings = None
        self.vocabulary = None
        self.word_to_id = None
        self.id_to_word = None
        self.embedding_matrix = None
    def load_glove_embeddings(self):
        print("Loading GloVe embeddings...")
        self.embeddings = {}
        with open(self.glove_file_path, 'r', encoding='utf-8') as f:
            for line in f:
                values = line.split()
                word = values[0]
                vector = np.asarray(values[1:], dtype='float32')
                self.embeddings[word] = vector
        print("GloVe embeddings loaded.")
    def preprocess_data(self):
        """Load and preprocess the dataset."""
        print("Starting data preprocessing...")
        preprocessor = SentimentPreprocessor(self.dataset_path)
        preprocessor.preprocess()
        self.corpus = preprocessor.df['review'].tolist()
        print("Data preprocessing complete.")
    def build_vocabulary(self, sentences):
        print("Building vocabulary...")
        word_counts = Counter(word for sentence in sentences for word in sentence)
        self.vocabulary = [word for word, count in word_counts.items() if count >= 5]
        self.word_to_id = {word: i for i, word in enumerate(self.vocabulary)}
        self.id_to_word = {i: word for i, word in enumerate(self.vocabulary)}
        print("Vocabulary built.")
```



```
    def create_embedding_matrix(self):
        print("Creating embedding matrix...")
        vocab_size = len(self.vocabulary)
        self.embedding_matrix = np.zeros((vocab_size, self.embedding_dim))
        for word, i in self.word_to_id.items():
            if word in self.embeddings:
                self.embedding_matrix[i] = self.embeddings[word]
        print("Embedding matrix created.")
    def save_model(self, save_path):
        """Save the embedding matrix and vocabulary."""
        print("Saving the model...")
        np.save(save_path + '_embeddings_300_d.npy', self.embedding_matrix)
        with open(save_path + '_vocab.txt', 'w') as f:
            for word in self.vocabulary:
                f.write(word + '\n')
        print("Model saved.")
    def load_model(self, load_path):
        """Load the embedding matrix and vocabulary."""
        print("Loading the model...")
        self.embedding_matrix = np.load(load_path + '_embeddings_300_d.npy')
        with open(load_path + '_vocab.txt', 'r') as f:
            self.vocabulary = [line.strip() for line in f]
        self.word_to_id = {word: i for i, word in enumerate(self.vocabulary)}
        self.id_to_word = {i: word for i, word in enumerate(self.vocabulary)}
        print("Model loaded.")
```

Figure 30 - Embedding code

The fascinating feature of embeddings is how similar words are represented in the embedding space. The visualisation of similar words with regard to a particular word follows :

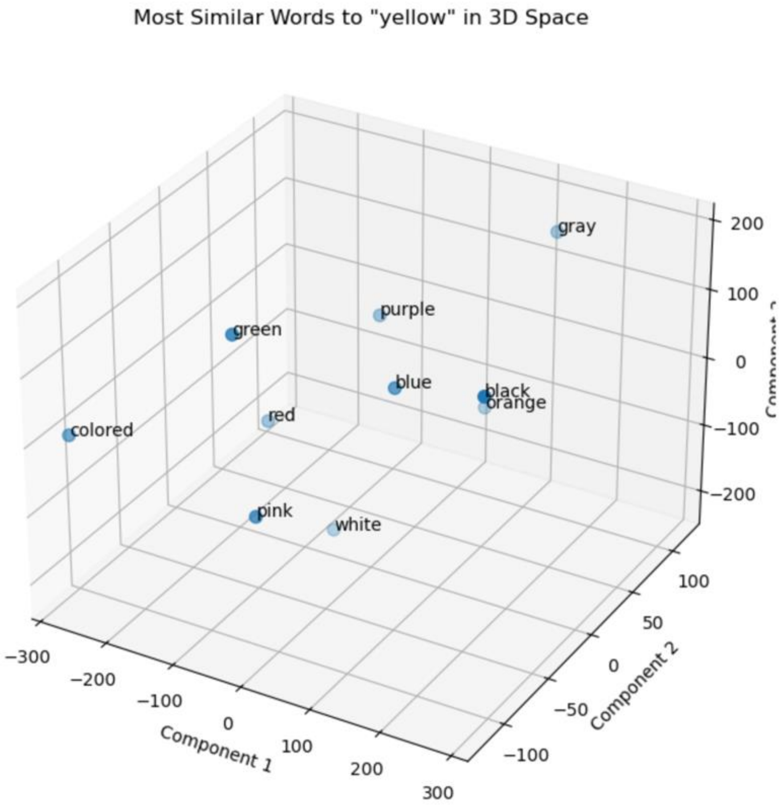
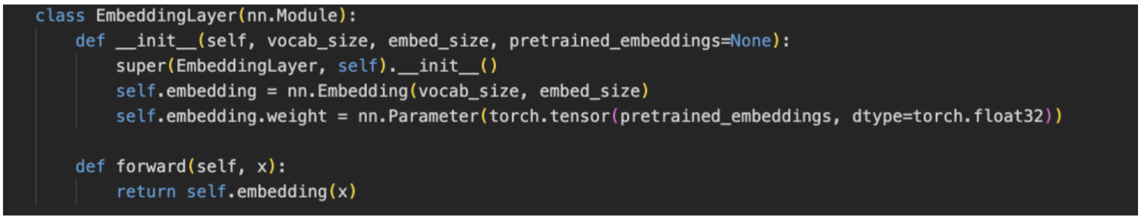


Figure 31 - Embedding visualisation

5.2.3 Building The Model

Now that we have an embedding, we can start assembling the model. To do so, we must define all the layers composing our model. As mentioned earlier, our model will comprise an embedding layer, a positional encoding layer, and the encoding blocks.

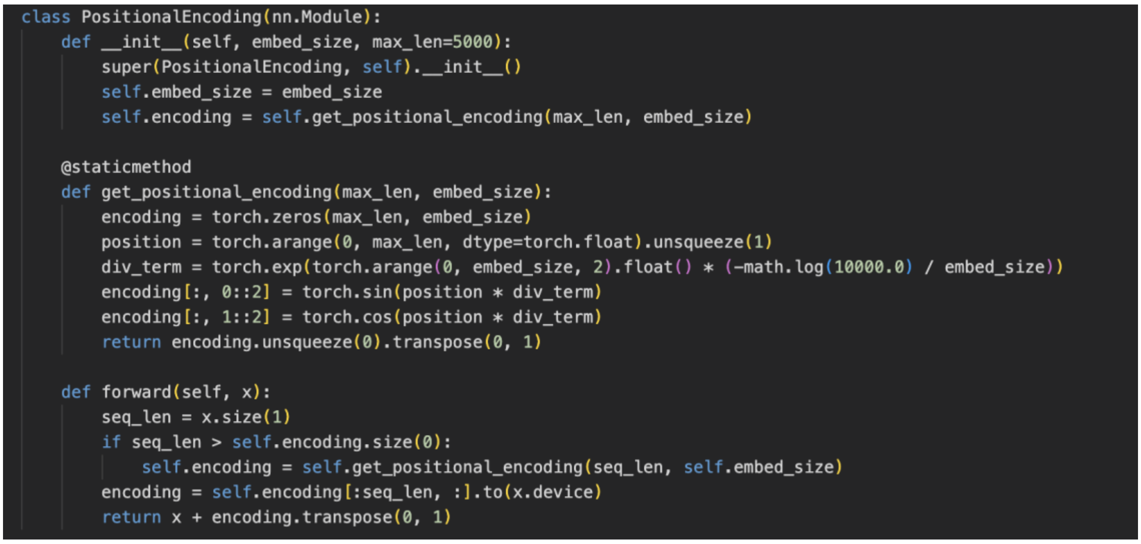
- **The embedding layer** receives our glove-embedding, maps each word of the vocabulary to its correspondent vector, and forwards the embedding to the next layer.



```python
class EmbeddingLayer(nn.Module):
    def __init__(self, vocab_size, embed_size, pretrained_embeddings=None):
        super(EmbeddingLayer, self).__init__()
        self.embedding = nn.Embedding(vocab_size, embed_size)
        self.embedding.weight = nn.Parameter(torch.tensor(pretrained_embeddings, dtype=torch.float32))

    def forward(self, x):
        return self.embedding(x)
```

Figure 32 - Model embedding layer code

- **The positional encoding layer** adds information about the position of each word in a sequence, as transformer-based architectures don't process the input sequentially.

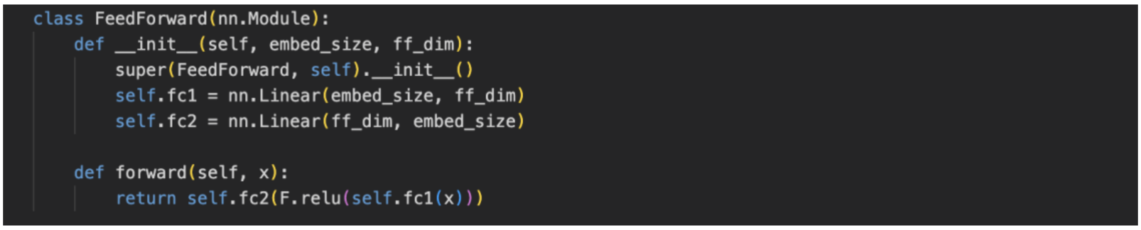


```python
class PositionalEncoding(nn.Module):
    def __init__(self, embed_size, max_len=5000):
        super(PositionalEncoding, self).__init__()
        self.embed_size = embed_size
        self.encoding = self.get_positional_encoding(max_len, embed_size)

    @staticmethod
    def get_positional_encoding(max_len, embed_size):
        encoding = torch.zeros(max_len, embed_size)
        position = torch.arange(0, max_len, dtype=torch.float).unsqueeze(1)
        div_term = torch.exp(torch.arange(0, embed_size, 2).float() * (-math.log(10000.0) / embed_size))
        encoding[:, 0::2] = torch.sin(position * div_term)
        encoding[:, 1::2] = torch.cos(position * div_term)
        return encoding.unsqueeze(0).transpose(0, 1)

    def forward(self, x):
        seq_len = x.size(1)
        if seq_len > self.encoding.size(0):
            self.encoding = self.get_positional_encoding(seq_len, self.embed_size)
        encoding = self.encoding[:seq_len, :].to(x.device)
        return x + encoding.transpose(0, 1)
```

Figure 33 - Model positional encoding layer code

One encoding block comprises the following components:

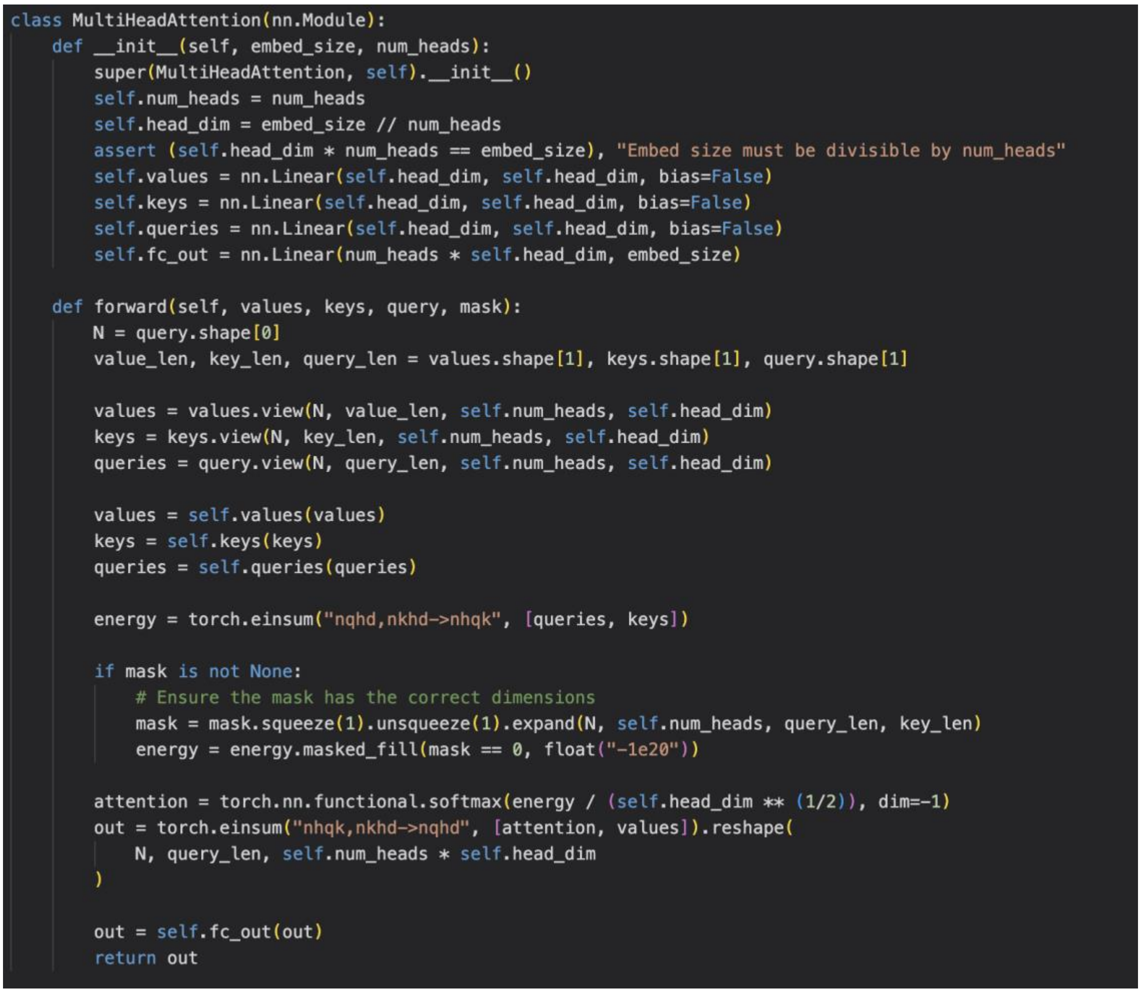
- **The feed-forward network** consists of two dense layers with a ReLU activation function in between. This network processes each position separately and identically, applying a non-linear transformation to the input embeddings.



```python
class FeedForward(nn.Module):
    def __init__(self, embed_size, ff_dim):
        super(FeedForward, self).__init__()
        self.fc1 = nn.Linear(embed_size, ff_dim)
        self.fc2 = nn.Linear(ff_dim, embed_size)

    def forward(self, x):
        return self.fc2(F.relu(self.fc1(x)))
```

Figure 34 - Model FNN code

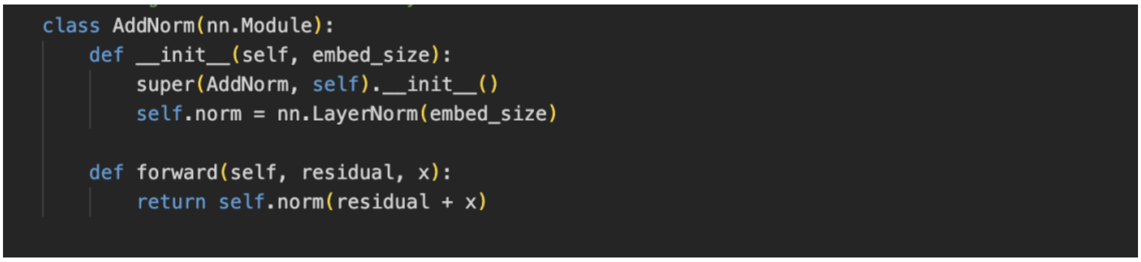
- **The multi-head attention** layer allows the model to focus on different positions of the input sequence simultaneously. It splits the input into multiple heads, applies self-attention to each head, and concatenates the outputs.



```python
class MultiHeadAttention(nn.Module):
    def __init__(self, embed_size, num_heads):
        super(MultiHeadAttention, self).__init__()
        self.num_heads = num_heads
        self.head_dim = embed_size // num_heads
        assert (self.head_dim * num_heads == embed_size), "Embed size must be divisible by num_heads"
        self.values = nn.Linear(self.head_dim, self.head_dim, bias=False)
        self.keys = nn.Linear(self.head_dim, self.head_dim, bias=False)
        self.queries = nn.Linear(self.head_dim, self.head_dim, bias=False)
        self.fc_out = nn.Linear(num_heads * self.head_dim, embed_size)

    def forward(self, values, keys, query, mask):
        N = query.shape[0]
        value_len, key_len, query_len = values.shape[1], keys.shape[1], query.shape[1]

        values = values.view(N, value_len, self.num_heads, self.head_dim)
        keys = keys.view(N, key_len, self.num_heads, self.head_dim)
        queries = query.view(N, query_len, self.num_heads, self.head_dim)

        values = self.values(values)
        keys = self.keys(keys)
        queries = self.queries(queries)

        energy = torch.einsum("nqhd,nkhd->nhqk", [queries, keys])

        if mask is not None:
            # Ensure the mask has the correct dimensions
            mask = mask.squeeze(1).unsqueeze(1).expand(N, self.num_heads, query_len, key_len)
            energy = energy.masked_fill(mask == 0, float("-1e20"))

        attention = torch.nn.functional.softmax(energy / (self.head_dim ** (1/2)), dim=-1)
        out = torch.einsum("nhqk,nkhd->nqhd", [attention, values]).reshape(
            N, query_len, self.num_heads * self.head_dim
        )

        out = self.fc_out(out)
        return out
```

Figure 35 - Model multi-head attention code

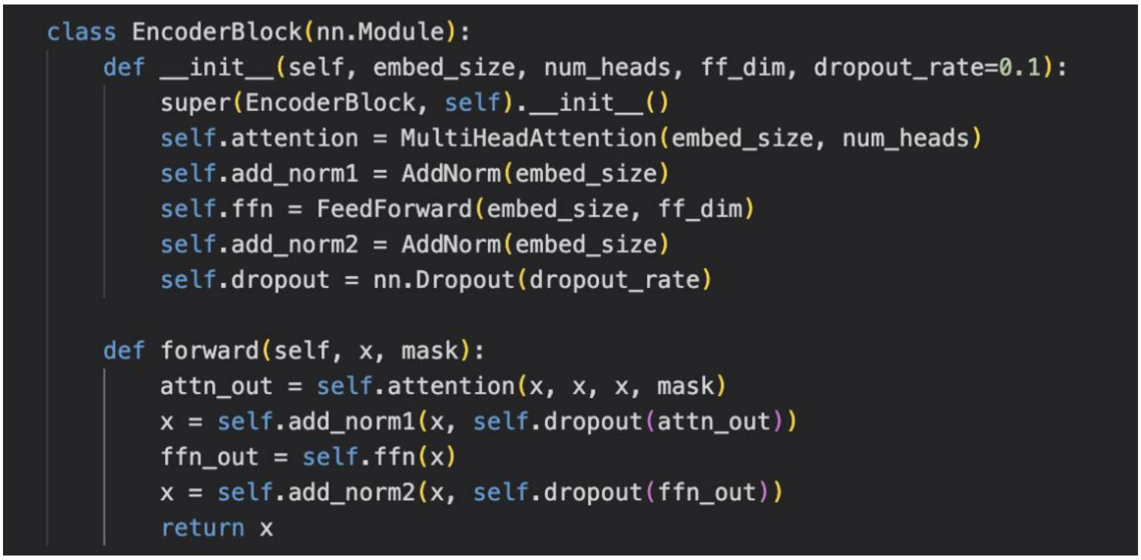
- **The add and normalise** layer adds the original input embedding to the output of the multi-head attention and then normalises the values to prevent them from becoming too large or too low.



```python
class AddNorm(nn.Module):
    def __init__(self, embed_size):
        super(AddNorm, self).__init__()
        self.norm = nn.LayerNorm(embed_size)

    def forward(self, residual, x):
        return self.norm(residual + x)
```

Figure 36 - Model add&normalise layer code

The code of one encoder block with all of the pieces put together follows :

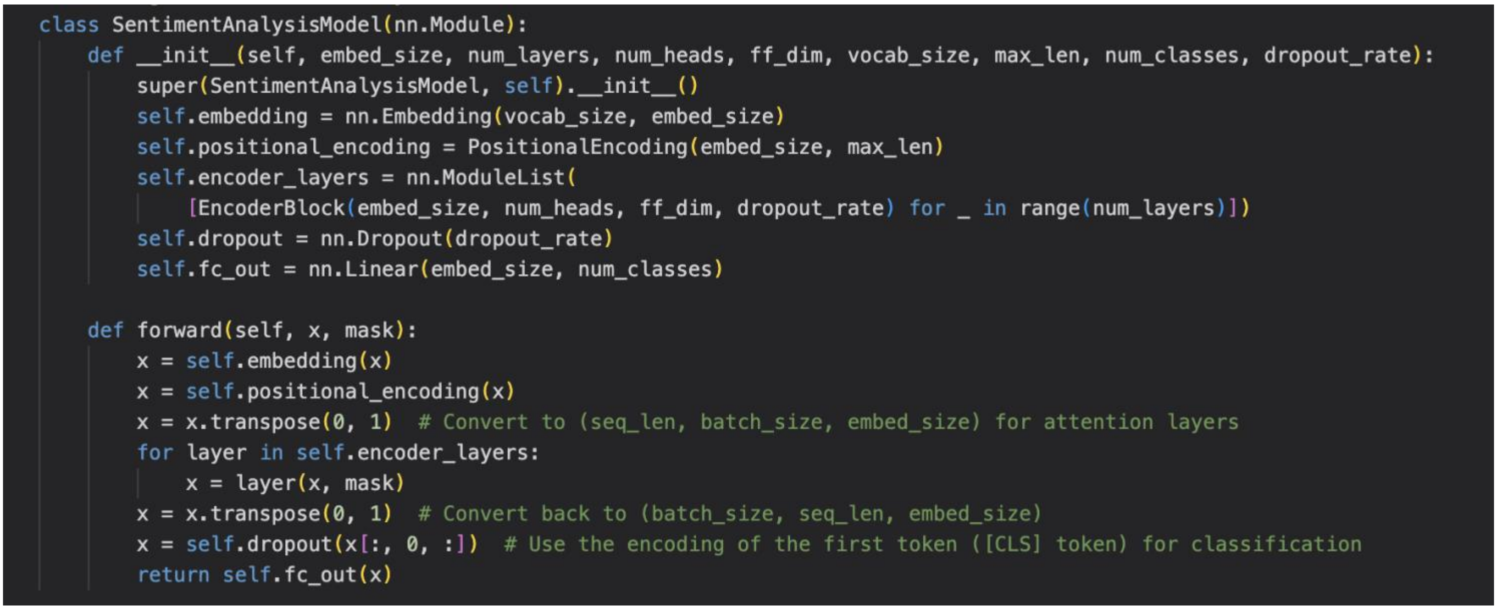


```python
class EncoderBlock(nn.Module):
    def __init__(self, embed_size, num_heads, ff_dim, dropout_rate=0.1):
        super(EncoderBlock, self).__init__()
        self.attention = MultiHeadAttention(embed_size, num_heads)
        self.add_norm1 = AddNorm(embed_size)
        self.ffn = FeedForward(embed_size, ff_dim)
        self.add_norm2 = AddNorm(embed_size)
        self.dropout = nn.Dropout(dropout_rate)

    def forward(self, x, mask):
        attn_out = self.attention(x, x, x, mask)
        x = self.add_norm1(x, self.dropout(attn_out))
        ffn_out = self.ffn(x)
        x = self.add_norm2(x, self.dropout(ffn_out))
        return x
```

Figure 37 - Model encoding block code

I added a dropout layer to the encoding block to prevent overfitting by randomly turning off some input units during training. This makes the model learn stronger, more general features.

With all the pieces ready, we can now put the model together:



```
class SentimentAnalysisModel(nn.Module):
    def __init__(self, embed_size, num_layers, num_heads, ff_dim, vocab_size, max_len, num_classes, dropout_rate):
        super(SentimentAnalysisModel, self).__init__()
        self.embedding = nn.Embedding(vocab_size, embed_size)
        self.positional_encoding = PositionalEncoding(embed_size, max_len)
        self.encoder_layers = nn.ModuleList(
            [EncoderBlock(embed_size, num_heads, ff_dim, dropout_rate) for _ in range(num_layers)])
        self.dropout = nn.Dropout(dropout_rate)
        self.fc_out = nn.Linear(embed_size, num_classes)

    def forward(self, x, mask):
        x = self.embedding(x)
        x = self.positional_encoding(x)
        x = x.transpose(0, 1)  # Convert to (seq_len, batch_size, embed_size) for attention layers
        for layer in self.encoder_layers:
            x = layer(x, mask)
        x = x.transpose(0, 1)  # Convert back to (batch_size, seq_len, embed_size)
        x = self.dropout(x[:, 0, :])  # Use the encoding of the first token ([CLS] token) for classification
        return self.fc_out(x)
```

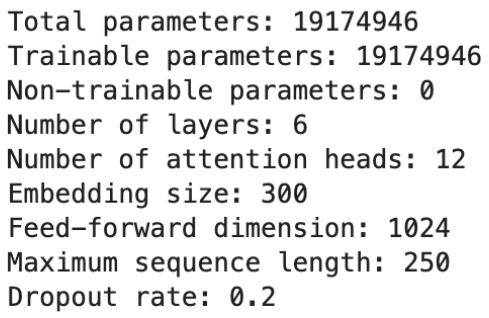
Figure 38 - Model by author code

The model receives as parameters the embedding size, the number of encoder layers, the number of attention heads, the feed-forward layer dimensions, the vocabulary size, the maximum length of the input sequence, the number of output classes, and the dropout rate. During the forward pass, input data is embedded and encoded positionally, then sequentially passed through each encoder layer. Dropout is then applied before the final output is produced through a fully connected layer, resulting in class probabilities or scores for sentiment analysis.

5.2.4 Training And Validation

The training has been the most challenging part of this journey. After more than three days of training under Google Collab A100 GPU and tweaking parameters to achieve better results, I finally understood that the main problem was the method I was using for the training. Deep neural networks require a titanic amount of data to perform well; thus, using 50,000 reviews for training and testing was clearly not enough. The output of my first models confirmed this. They did poorly and roughly attained 50% accuracy. To troubleshoot this issue, techniques like data augmentation could have helped expand the set and provided more examples for the model to learn from, but time was clearly not in my favour to get into it.

The model, in itself, seems quite robust, as showcased in the model card :



```
Total parameters: 19174946
Trainable parameters: 19174946
Non-trainable parameters: 0
Number of layers: 6
Number of attention heads: 12
Embedding size: 300
Feed-forward dimension: 1024
Maximum sequence length: 250
Dropout rate: 0.2
```

Figure 39 - Model From-Scratch card

With more than 19 million parameters, the model can take up to 250 characters as input sequences and predict the sentiment. With its complex transformer-based architecture for sentiment analysis, this model should be capable of capturing complex patterns in data. Initially, I implemented a padding mask to focus the model on relevant data. However, this led to inconsistent results, likely due to the complexities of the transformer model. After researching and finding similar challenges reported by others, I decided to remove the padding mask. This simplification resulted in more stable training and better overall performance, proving that sometimes less is more in model optimisation.

The results of the training and validation of my best model are as follows :

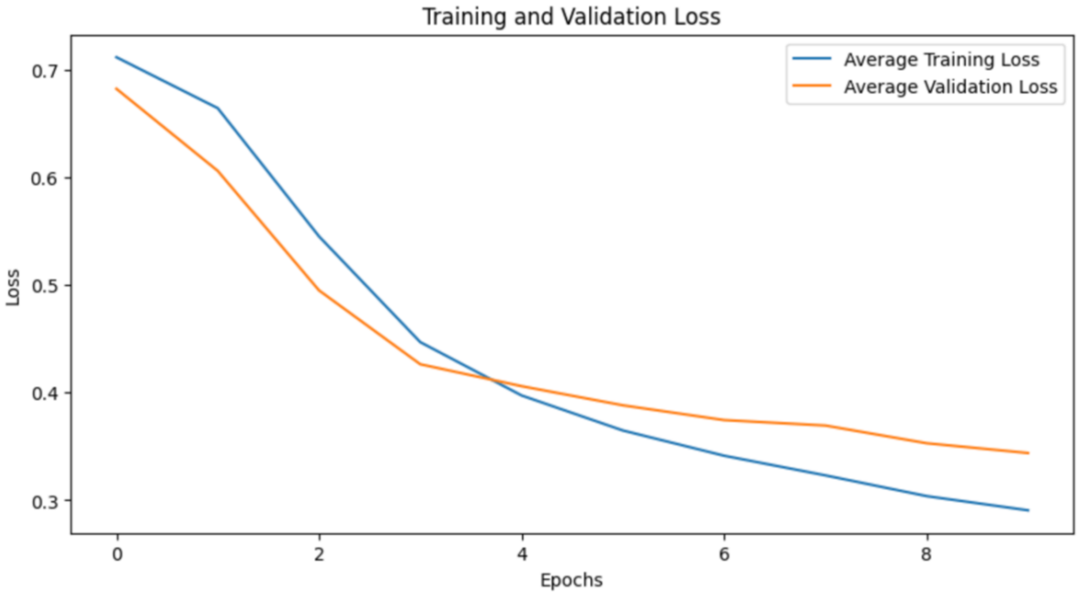


Figure 40 - Model From-Scratch Training and Validation Loss

The training and validation loss has steadily decreased throughout the epochs, which indicates that the model has learned effectively and generalised well to unseen data.

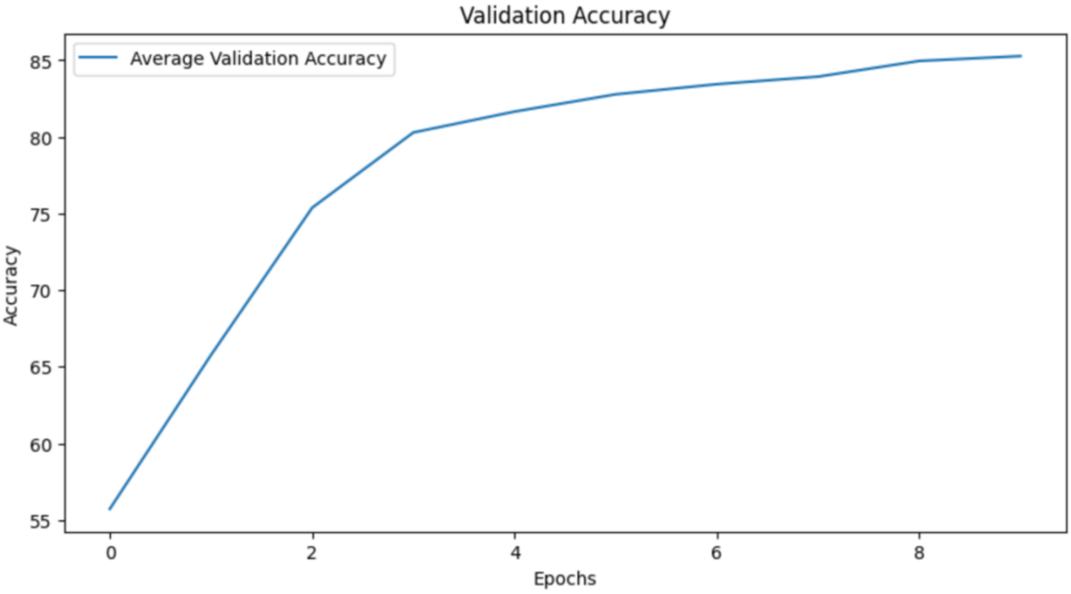


Figure 41 - Model From-Scratch Validation Accuracy

The validation accuracy showcases how the model has improved during the training. The accuracy consistently increased, reaching 85% by the end of the training. This demonstrates a staggering improvement compared to my previous models, which didn't get over 50% accuracy.

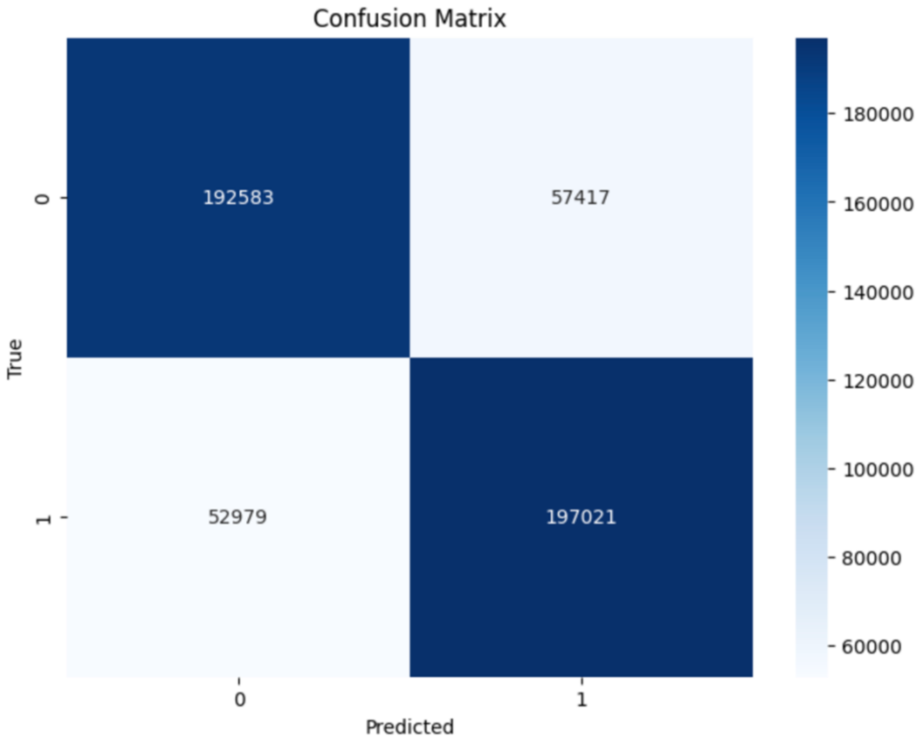
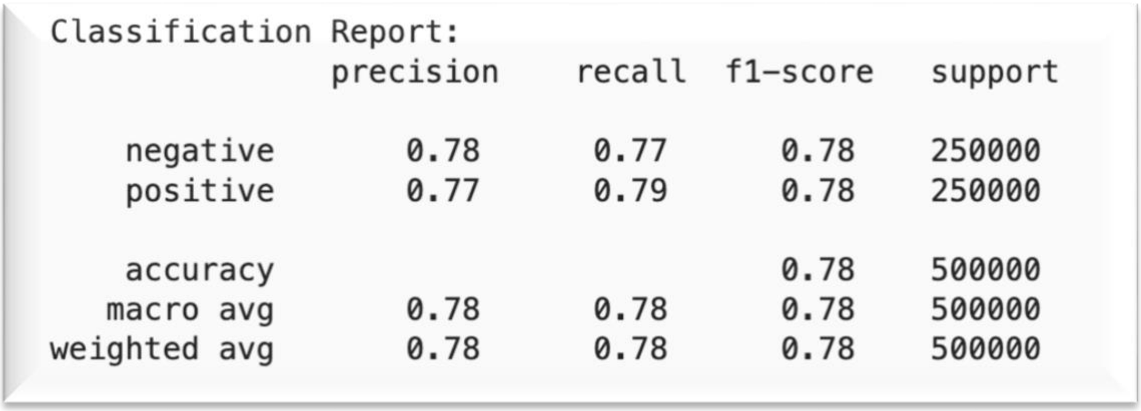


Figure 42 - Model From-Scratch Confusion Matrix (All folds)

This confusion matrix combines the prediction of all folds on 10 epochs. Out of 500,000 samples, 389,604 were well predicted, whereas 1103,96 were wrongly predicted. This is a fair result for one training, knowing the model could be retrained to improve its performance.



```
Classification Report:
              precision    recall  f1-score   support

    negative       0.78      0.77      0.78    250000
    positive       0.77      0.79      0.78    250000

    accuracy                           0.78    500000
   macro avg       0.78      0.78      0.78    500000
weighted avg       0.78      0.78      0.78    500000
```

Figure 43 - Classification Report Model From-Scratch

- **The precision** metric measures how many correct predictions were actually correct. The result indicates that **78%** of the instances predicted as negative were actually negative, whereas **77%** of the instances predicted as positive were actually positive.

- **The recall** measures the model's ability to predict all the correct instances. The results highlight that the model better predicts negative reviews **(79% recall)** than positive reviews **(77% recall)**.
- **The F1 Score** strikes a balance between precision and recall, giving the model an overall accuracy of **78%.** The model shows that it can effectively distinguish between positive and negative reviews with some minor errors in the prediction.

5.3 Fine-tuning Approach

Fine-tuning is making minor adjustments to a pre-trained machine-learning model to adapt it to a specific task. Pre-trained models are often built with millions or billions of parameters and trained on enormous amounts of data. Such models have acquired robust and broad learning, making them very powerful. Fine-tuning, hence, makes it easier and cheaper to have access to powerful and custom models serving a specific purpose (*What is Fine-Tuning? | IBM* 2024).

Fine-tuning has become the chosen approach in real-world applications because it requires less data to train a model and reduces the need for extensive computational resources and time. Coupled with the overall exceptional performance of pre-trained models, this trend is further supported by the growing number of publicly available pre-trained models and robust frameworks and libraries (*What is Fine-Tuning? | IBM* 2024).

For our sentiment analysis task, we will fine-tune the **"bert-base-uncased,"** a well-known pre-trained model for natural language understanding with 110 million parameters trained on 3TO of data. The model is available for free on **HuggingFace (a GitHub for AI)** and can be accessed through the Transformers library.

The results of the fine-tuned model after the training and validation follow:

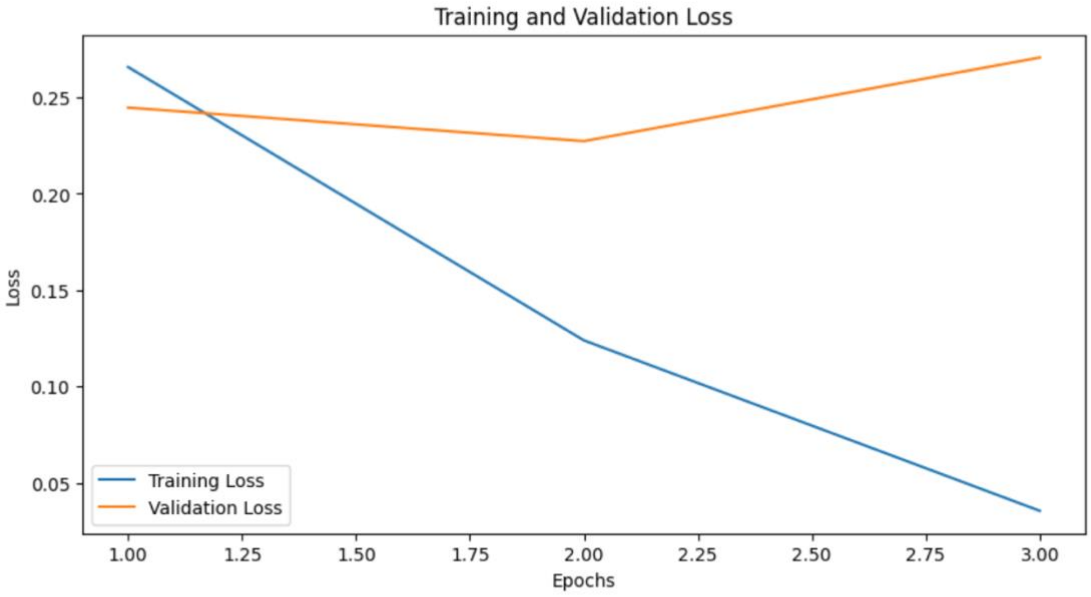


Figure 44 - Fine-Tuned Model Training and Validation Loss

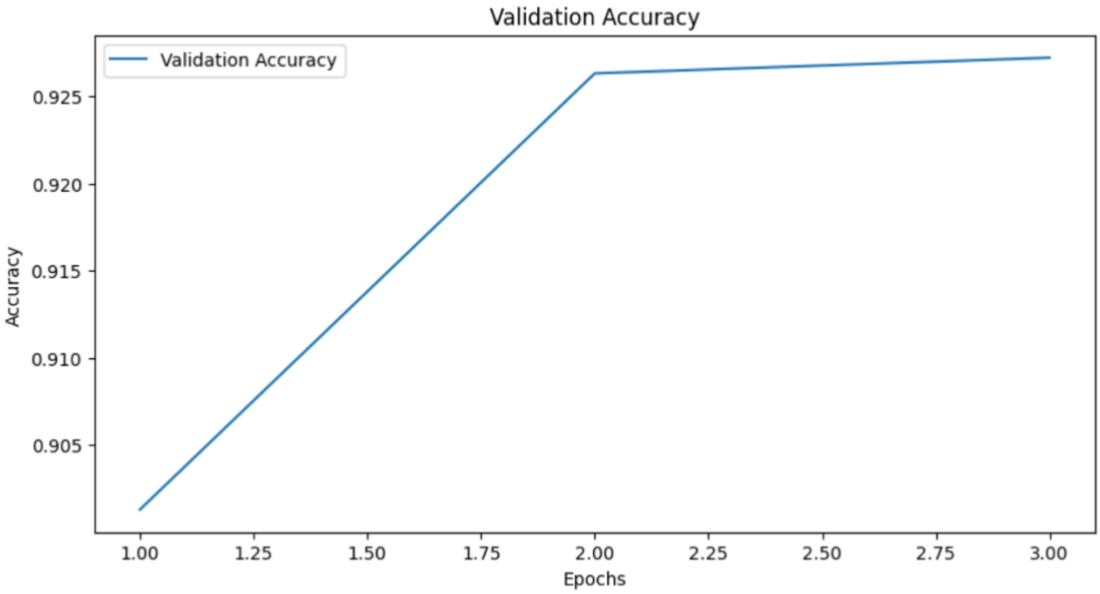


Figure 45 - Fine-Tuned Model Validation Accuracy

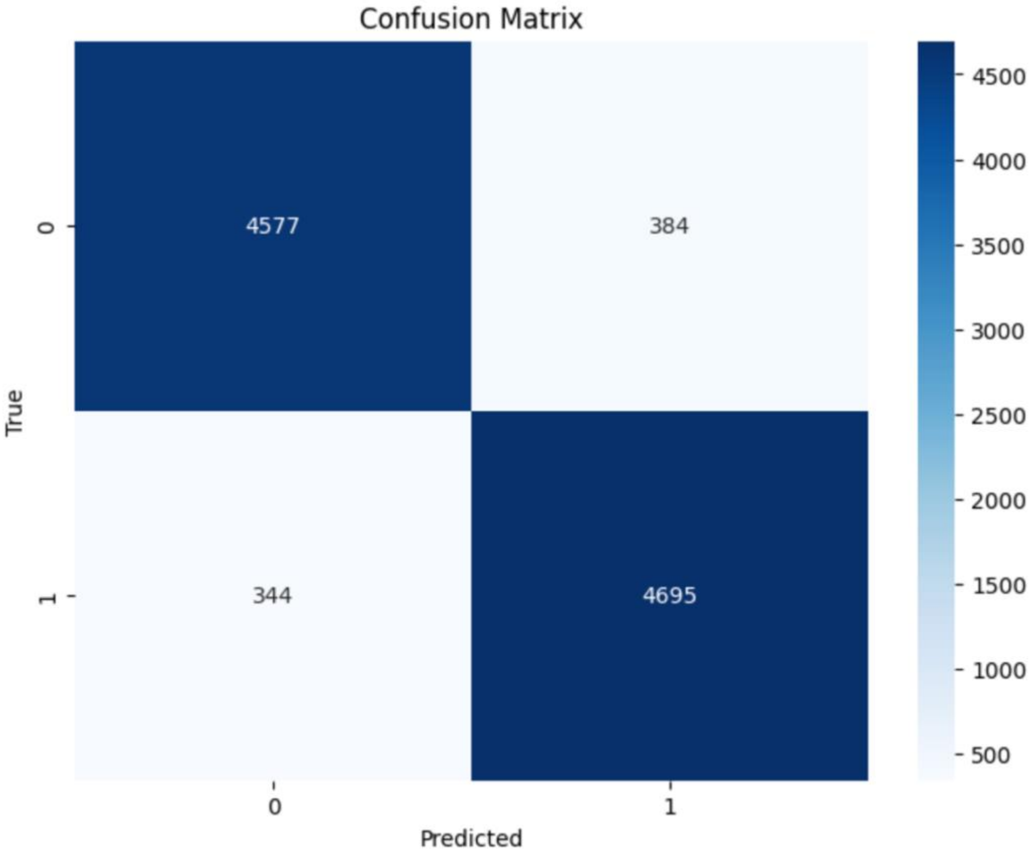
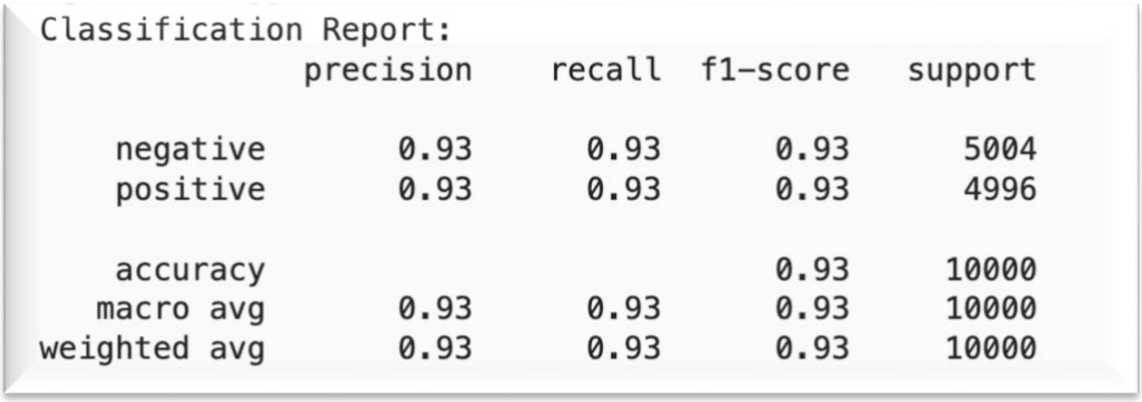


Figure 46 - Fine-Tuned Model Confusion Matrix



```
Classification Report:
              precision    recall  f1-score   support

    negative       0.93      0.93      0.93      5004
    positive       0.93      0.93      0.93      4996

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000
```

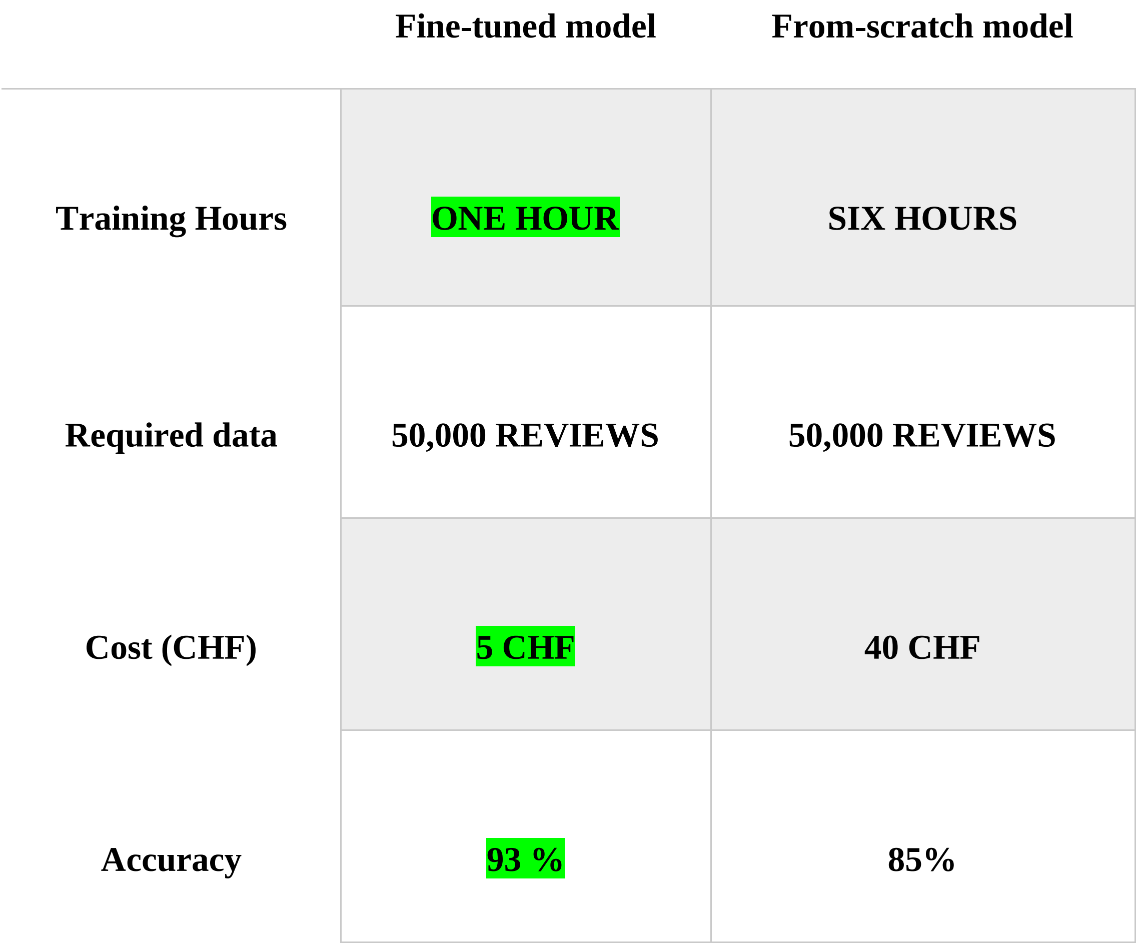
Figure 47 - Fine-Tuned Model Classification Report

Overall, the fine-tuned model has lived up to its hype. Despite an increase in the validation loss after the second epochs, which likely suggests some overfitting, the model performs better than the model built from scratch (unsurprisingly and thankfully), with an overall accuracy of **93%** reached after approximately an hour of training.

6. Model Comparison

A comparison is necessary to draw conclusions about the model that corresponds better to our simulation needs. Firstly, I have chosen some criteria that determine the best model regarding computational resources and time. Then, we will test the models on the new sample and see how they behave.

6.1 Resources And Time Comparison



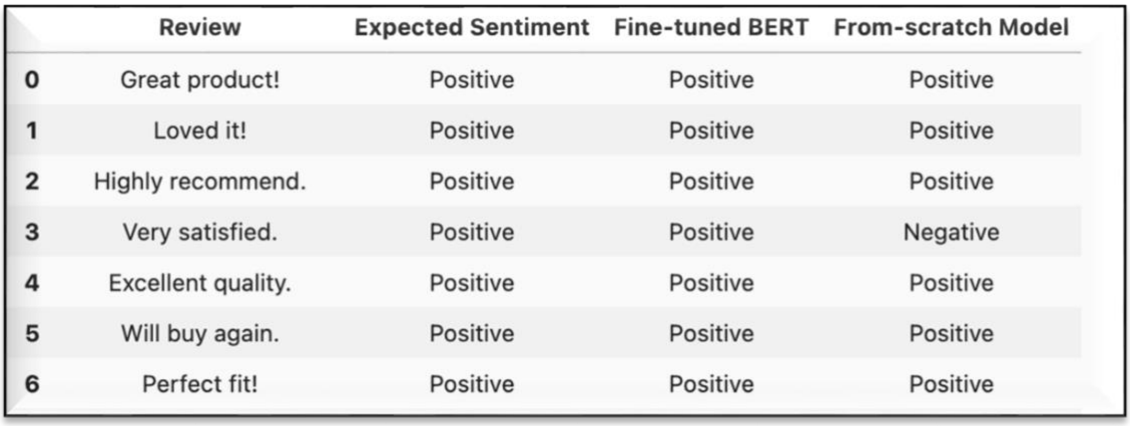
	Fine-tuned model	From-scratch model
Training Hours	**ONE HOUR**	**SIX HOURS**
Required data	**50,000 REVIEWS**	**50,000 REVIEWS**
Cost (CHF)	**5 CHF**	**40 CHF**
Accuracy	**93 %**	**85%**

Figure 48 - Model computational resources and time comparison

Unsurprisingly, the fine-tuned model outperforms the from-scratch model by requiring less time to train and achieving better accuracy while costing less.

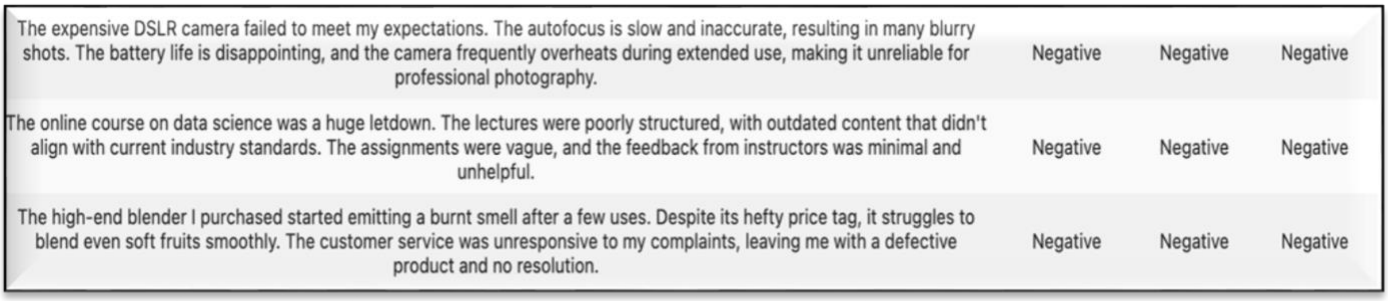
6.2 Testing On New Samples

Testing a model on new samples is a great way to assert its effectiveness, identify if it has overfitted, and simulate some real-world usage. Therefore, I have created a list of positive and negative reviews to observe how each model behaves on each list. The quite surprising results follow :



	Review	Expected Sentiment	Fine-tuned BERT	From-scratch Model
0	Great product!	Positive	Positive	Positive
1	Loved it!	Positive	Positive	Positive
2	Highly recommend.	Positive	Positive	Positive
3	Very satisfied.	Positive	Positive	Negative
4	Excellent quality.	Positive	Positive	Positive
5	Will buy again.	Positive	Positive	Positive
6	Perfect fit!	Positive	Positive	Positive

Figure 49 - Easy Positives Testing



The expensive DSLR camera failed to meet my expectations. The autofocus is slow and inaccurate, resulting in many blurry shots. The battery life is disappointing, and the camera frequently overheats during extended use, making it unreliable for professional photography.	Negative	Negative	Negative
The online course on data science was a huge letdown. The lectures were poorly structured, with outdated content that didn't align with current industry standards. The assignments were vague, and the feedback from instructors was minimal and unhelpful.	Negative	Negative	Negative
The high-end blender I purchased started emitting a burnt smell after a few uses. Despite its hefty price tag, it struggles to blend even soft fruits smoothly. The customer service was unresponsive to my complaints, leaving me with a defective product and no resolution.	Negative	Negative	Negative

Figure 50 - Complex Negatives Testing

Despite some minor inaccuracy, the From-scratch model seems capable of contending with the Fine-tuned model on even complex reviews. This test confirms the recall result: the From-scratch model distinguishes negative from positive reviews better. I am stunned that a ten times less sophisticated model can challenge a model pre-trained on 3 billion text data. This showcases the power of deep neural networks and Transformers, which have been vital to findings even outside of NLP.

7. Attention Is All Your Business Needs

Our simulation has given birth to a sentiment analysis model that can accurately capture and interpret the sentiment expressed in customer reviews. This model can provide the investment firm with insights, enabling them to make **data-driven decisions** that maximise their return on investment. This simulation illustrates how businesses can harness the power of sentiment analysis to enhance their strategic decision-making. In competitive fields like finance and investment, data-driven insights can be crucial, helping companies gain an edge and make more informed choices.

Netflix's approach to the entertainment industry pertains to this idea. To understand viewer preferences (themes, actors, seasons...), identify new trends and make informed investments in originals, data collected from users' ratings and monitored opinions on social media platforms have given the brand invaluable insights into what content to produce or to acquire, contents that should continue or put aside, and how to refine their recommendation algorithm for their viewers. Data-driven decision making strategies help Netflix maintain its competitiveness in the entertainment industry (Walsh 2019).

In June 2023, British Airways revealed its first travel trends report, resulting from various sentiment analysis efforts to understand customer feedback and improve its services. The company has examined the behaviours and attitudes shaping its travellers by scrapping reviews from popular airline review sites. This has led to the introduction of all-inclusive travel offers and holiday packages, especially for millennials. This is an example of **customer-centric product development** that showcases how the company refined its services by listening to its customers' needs (Airways). Despite the ongoing challenges in the aviation industry since COVID-19, British Airways generated substantial profit, highlighting the effectiveness of its customer-oriented strategies informed by sentiment analysis efforts (*British Airways made £50 per second profit during first nine months of 2023* 2023)

SAMSUNG was caught in a bombshell in 2016 due to the faulty battery issue of its Galaxy Note 7. This has considerably damaged the company's reputation and resulted in substantial losses. During the crisis, one of their key risk management strategies was to track newspapers and online articles, social media (Twitter, Facebook, Youtube), experts' analysis and opinions, consumer forums, and discussion platforms. This approach allowed the brand to gather and analyse real-time feedback, understand public sentiment and adjust their communication strategy accordingly. While sentiment analysis alone doesn't directly resolve the issue, it offers critical insights that can significantly shape a company's communication strategy during **crisis management**. By understanding public sentiment, businesses can tailor their messages more effectively, addressing the concerns and emotions of their audience. This approach helps manage the crisis more smoothly and is critical in rebuilding trust and stabilising the brand in the aftermath (Abbas 2023).

By accurately capturing and interpreting customer sentiments, businesses can make informed decisions that drive success, as seen in the examples of Netflix, British Airways, and Samsung. Whether it's optimising content production, refining services, or managing crises, sentiment analysis provides invaluable insights that help companies stay competitive, address customer needs effectively, and rebuild trust when faced with challenges. The data-driven approach is, I believe, essential for thriving in today’s competitive market.

8. Conclusion

In this thesis, "**Attention is All Your Business Needs: A Sentiment Analysis of Customer Reviews**," we have explored the application of attention mechanisms to enhance the accuracy and effectiveness of sentiment analysis. Integrating attention mechanisms within NLP models has proven to be a significant advancement, allowing for a more nuanced understanding of customer feedback and sentiments.

In this memoir, we have demonstrated the potential of attention-based models to analyse text and predict sentiments, thereby aiding businesses in making informed decisions. The simulated project for an investment firm showcased the practical applications of these models in real-world settings, highlighting their ability to transform customer reviews into actionable insights. Furthermore, examining how major tech companies leverage these techniques underscores the competitive advantage that sophisticated AI tools can provide. By staying at the forefront of NLP and deep learning advancements, businesses can gain valuable insights that drive strategic decisions and enhance customer satisfaction.

While this research has explored advances in sentiment analysis, several avenues for future exploration remain. Expanding models to support multilingual sentiment analysis would increase their global applicability while developing real-time analysis capabilities could offer businesses immediate insights for quicker decision-making. Finally, addressing ethical considerations, such as bias and transparency, will be essential as AI continues to play a significant role in business strategies.

This research enhances our understanding of how learning machine interprets human language, offering valuable insights for businesses to refine their strategies based on consumer sentiment. As AI evolves, I am convinced that its integration into business processes is not just an option but a crucial step for all industries. While a discussion is necessary to determine which processes would benefit most from AI, the reality is that firms that choose not to embrace AI will likely fall behind in an increasingly competitive market. The future belongs to those who adapt, and integrating advanced AI techniques will be essential for staying ahead.

Bibliography

3 steps Amazon is taking to stop fake reviews, [online]. Retrieved from : https://www.aboutamazon.com/news/policy-news-views/how-amazon-is-working-to-stop-fake-reviews [accessed 18 July 2024].

2023 Ratings & Reviews Report: Amazon Edition, 2023.

2023 Tripadvisor Review Transparency Report, [online]. Retrieved from : https://tripadvisor.shorthandstories.com/2023-tripadvisor-review-transparency-report/ [accessed 6 July 2024].

ABBAS, Tahir, 2023. Remarkable Recovery: Samsung Crisis Management Case Study. *CMI* [online]. 6 June 2023. Retrieved from : https://changemanagementinsight.com/samsung-crisis-management-case-study/ [accessed 10 August 2024].

AIRWAYS, British. British Airways holidays reveals travel trends: all-inclusive on the rise and holiday packages in vogue with millennials. [online]. Retrieved from : https://mediacentre.britishairways.com/pressrelease/details/16548 [accessed 10 August 2024].

Amazon improves the customer reviews experience with AI, [online]. Retrieved from : https://www.aboutamazon.com/news/amazon-ai/amazon-improves-customer-reviews-with-generative-ai [accessed 18 July 2024].

Amazon.com, [online]. Retrieved from : https://www.amazon.com/ [accessed 6 July 2024].

AZENCOTT, Chloé-Agathe, 2022. *Introduction au Machine Learning - 2e éd.* Dunod. ISBN 978-2-10-084143-1. Google-Books-ID: kfxZEAAAQBAJ

BAHDANAU, Dzmitry, CHO, Kyunghyun and BENGIO, Yoshua, 2016. *Neural Machine Translation by Jointly Learning to Align and Translate* [online]. arXiv:1409.0473. arXiv. arXiv:1409.0473. Retrieved from : http://arxiv.org/abs/1409.0473 [accessed 24 July 2024]. arXiv:1409.0473 [cs, stat]

BERGER, Jonah, 2014. Word of mouth and interpersonal communication: A review and directions for future research. *Journal of Consumer Psychology*. Vol. 24, no. 4, pp. 586–607. DOI 10.1016/j.jcps.2014.05.002.

BERNOFF, Josh and SCHADLER, Ted, 2010. *Empowered: Unleash Your Employees, Energize Your Customers, and Transform Your Business*. Harvard Business Press. ISBN 978-1-4221-6233-0. Google-Books-ID: DYVHdCXwHEgC

BHASIN, Hitesh, 2023. Customer Reviews: Definition, Importance and Strategies. *Marketing91* [online]. 21 July 2023. Retrieved from : https://www.marketing91.com/customer-reviews/ [accessed 6 July 2024].

BISHOP, Christopher M. and BISHOP, Hugh, 2023. *Deep Learning: Foundations and Concepts*. Springer Nature. ISBN 978-3-031-45468-4. Google-Books-ID: 0uTgEAAAQBAJ

BOIY, Erik and MOENS, Marie-Francine, 2009. A Machine Learning Approach to Sentiment Analysis in Multilingual Web Texts. *Inf. Retr.* Vol. 12, pp. 526–558. DOI 10.1007/s10791-008-9070-z.

British Airways made £50 per second profit during first nine months of 2023, 2023*The Independent* [online]. Retrieved from : https://www.independent.co.uk/travel/news-and-advice/british-airways-profit-2023-iag-b2436997.html [accessed 10 August 2024].

CAPLAN, Frank, 2002. What Is a Customer? *Quality Engineering*. Vol. 14, no. 2, pp. ix–x. DOI 10.1081/QEN-100108666.

Customer review, 2024*Wikipedia* [online]. Retrieved from : https://en.wikipedia.org/w/index.php?title=Customer_review&oldid=1233572635 [accessed 18 July 2024]. Page Version ID: 1233572635

DE BRUYN, Arnaud. Will They Listen Anyway? Viral Marketing and the Effectiveness of Online Word-of-Mouth Referrals. .

Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville - Google Books, [online]. Retrieved from : https://books.google.ch/books?hl=en&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=ian+goodfellow+deep+learning&ots=MOO-gsnGOW&sig=M7qTlp1-QBQI740y_f78GiLPlxE&redir_esc=y#v=onepage&q=ian%20goodfellow%20deep%20learning&f=false [accessed 22 July 2024].

Demystifying the Add & Norm Block in the Transformer Neural Network Architecture: With Code, [online]. Retrieved from : https://www.linkedin.com/pulse/demystifying-add-norm-block-transformer-neural-network-ajay-taneja [accessed 27 July 2024].

DOSHI, Ketan, 2021. Transformers Explained Visually (Part 1): Overview of Functionality. *Medium* [online]. 3 June 2021. Retrieved from : https://towardsdatascience.com/transformers-explained-visually-part-1-overview-of-functionality-95a6dd460452 [accessed 23 July 2024].

Effects of Restaurant Satisfaction and Knowledge Sharing Motivation on eWOM Intentions, . DOI 10.1177/1096348013515918.

Electronic word-of-mouth via consumer-opinion platforms: What motivates consumers to articulate themselves on the Internet?, . DOI 10.1002/dir.10073.

FIRTH, J. R., 1957. Applications of General Linguistics. *Transactions of the Philological Society*. Vol. 56, no. 1, pp. 1–14. DOI 10.1111/j.1467-968X.1957.tb00568.x.

FISKE, Susan T., 2018. *Social Beings: Core Motives in Social Psychology*. John Wiley & Sons. ISBN 978-1-119-49273-3. Google-Books-ID: zE6MDwAAQBAJ

GALLAGER, Robert G. (ed.), 2008. Introduction to digital communication. In : , *Principles of Digital Communication*, pp. 1–15. Cambridge : Cambridge University Press. ISBN 978-0-511-81349-8. DOI 10.1017/CBO9780511813498.002.

GOLDBERG, Yoav, 2016. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*. Vol. 57, pp. 345–420. DOI 10.1613/jair.4992.

Google Reviews: The Complete Guide for Businesses, *ReviewTrackers* [online]. Retrieved from : https://www.reviewtrackers.com/guides/google-reviews/ [accessed 18 July 2024].

GUPTA, Prashant, 2024. Transformer Models: A breakthrough in Artificial Intelligence. *Medium* [online]. 18 March 2024. Retrieved from : https://medium.com/@prashantgupta17/transformer-models-a-breakthrough-in-artificial-intelligence-e3de92d37f8f [accessed 23 July 2024].

How Amazon is using AI to spot fake reviews, [online]. Retrieved from : https://www.aboutamazon.com/news/policy-news-views/how-ai-spots-fake-reviews-amazon [accessed 19 July 2024].

HOYER, Wayne D., 1984. An Examination of Consumer Decision Making for a Common Repeat Purchase Product. *Journal of Consumer Research*. Vol. 11, no. 3, pp. 822–829. DOI 10.1086/209017.

HUETE-ALCOCER, Nuria, 2017. A Literature Review of Word of Mouth and Electronic Word of Mouth: Implications for Consumer Behavior. *Frontiers in Psychology*. Vol. 8. DOI 10.3389/fpsyg.2017.01256.

JURAFSKY, Dan and MARTIN, James H., 2000. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, N.J : Prentice Hall. Prentice Hall series in artificial intelligence. ISBN 978-0-13-095069-7.

KANG, Chanseok, 2020. RNN - Many-to-one. *Chan`s Jupyter* [online]. 6 December 2020. Retrieved from : https://goodboychan.github.io/python/deep_learning/tensorflow-keras/2020/12/06/01-RNN-Many-to-one.html [accessed 23 July 2024].

KATOLE, Hemant J., 2022. A research article on Importance of Online Customer Reviews on Customer Purchase. *The Business and Management Review*. Vol. 13, no. 03. DOI 10.24052/BMR/V13NU03/ART-04.

Lexicon-based sentiment analysis: What it is & how to conduct one | KNIME, [online]. Retrieved from : https://www.knime.com/blog/lexicon-based-sentiment-analysis [accessed 21 July 2024].

MALINGAN, Navaneeth, 2023. Attention Mechanism in Deep Learning. *Scaler Topics* [online]. 17 March 2023. Retrieved from : https://www.scaler.com/topics/deep-learning/attention-mechanism-deep-learning/ [accessed 24 July 2024].

MANNING, Christopher D., 2022. Human Language Understanding & Reasoning. *Daedalus*. Vol. 151, no. 2, pp. 127–138. DOI 10.1162/daed_a_01905.

McAuley-Lab/Amazon-Reviews-2023 · Datasets at Hugging Face, 2024 [online]. Retrieved from : https://huggingface.co/datasets/McAuley-Lab/Amazon-Reviews-2023 [accessed 18 July 2024].

MERLO, Omar, EISINGERICH, Andreas B. and AUH, Seigyoung, 2013. Why Customer Participation Matters. *MIT Sloan Management Review* [online]. Retrieved from : https://sloanreview.mit.edu/article/why-customer-participation-matters/ [accessed 22 April 2024].

MONGARAS, Gabriel, 2022. How Do Self-Attention Masks Work? *Medium* [online]. 27 October 2022. Retrieved from : https://gmongaras.medium.com/how-do-self-attention-masks-work-72ed9382510f [accessed 26 July 2024].

MONIKA, P. et al., 2022. Machine learning approaches for sentiment analysis: A survey. *International journal of health sciences*. Vol. 6, no. S4, pp. 1286–1300. DOI 10.53730/ijhs.v6nS4.6119.

MUMUNI, Alhassan G. et al., 2020. Online Product Review Impact: The Relative Effects of Review Credibility and Review Relevance. *Journal of Internet Commerce* [online]. Retrieved from : https://www.tandfonline.com/doi/abs/10.1080/15332861.2019.1700740 [accessed 5 May 2024].

NABI, Javaid, 2019. Recurrent Neural Networks (RNNs). *Medium* [online]. 21 July 2019. Retrieved from : https://towardsdatascience.com/recurrent-neural-networks-rnns-3f06d7653a85 [accessed 22 July 2024].

NEGNEVITSKY, Michael, 20. *Artificial intelligence: a guide to intelligent systems*. 2. ed., [Nachdr.]. New York Munich : Addison-Wesley. ISBN 978-0-321-20466-0.

NIELSEN, Michael. Neural Networks and Deep Learning. .

PATHAK, Ajeet Ram et al., 2020. Application of Deep Learning Approaches for Sentiment Analysis. In : AGARWAL, Basant et al. (eds.), *Deep Learning-Based Approaches for Sentiment Analysis*, pp. 1–31. Singapore : Springer. ISBN 9789811512162. DOI 10.1007/978-981-15-1216-2_1.

PETERSON, Jordan B., 2021. *Beyond Order: 12 More Rules for Life*. Penguin UK. ISBN 978-0-241-40765-3. Google-Books-ID: br8KEAAAQBAJ

PUSH-PULL, 2022. A Deep Dive into Amazon Consumer Review Statistics. *Push-Pull* [online]. 2 May 2022. Retrieved from : https://pushpullagency.com/blog/a-deep-dive-into-amazon-consumer-review-statistics/ [accessed 18 July 2024].

QUAN, Nguyen, 2023. Top 8 Applications of Natural Language Processing (NLP). *Eastgate Software* [online]. 8 December 2023. Retrieved from : https://eastgate-software.com/top-8-applications-of-natural-language-processing-nlp/ [accessed 21 July 2024].

RANJAN, Nihar et al., 2016. A Survey on Techniques in NLP. *International Journal of Computer Applications*. Vol. 134, no. 8, pp. 6–9. DOI 10.5120/ijca2016907355.

Recommendations on the Responsible Use of Ratings and Reviews on Digital Platforms, .

RUSSELL, Stuart J. and NORVIG, Peter, 2016. *Artificial intelligence : a modern approach* [online]. Pearson. ISBN 978-1-292-15396-4. Retrieved from : https://thuvienso.hoasen.edu.vn/handle/123456789/8967 [accessed 20 July 2024].

SACHINSONI, 2024. Recurrent Neural Networks (RNN) from Basic to Advanced. *Medium* [online]. 25 March 2024. Retrieved from : https://medium.com/@sachinsoni600517/recurrent-neural-networks-rnn-from-basic-to-advanced-1da22aafa009 [accessed 22 July 2024].

Self attention vs attention in transformers | Medium, [online]. Retrieved from : https://medium.com/@wwydmanski/whats-the-difference-between-self-attention-and-attention-in-transformer-architecture-3780404382f3 [accessed 25 July 2024].

SIMONS, Robert, 2014. Choosing the Right Customer. *Harvard Business Review* [online]. Retrieved from : https://hbr.org/2014/03/choosing-the-right-customer [accessed 17 July 2024].

SMITH, Bradney, 2024. Self-Attention Explained with Code. *Medium* [online]. 21 July 2024. Retrieved from : https://towardsdatascience.com/contextual-transformer-embeddings-using-self-attention-explained-with-diagrams-and-python-code-d7a9f0f4d94e [accessed 25 July 2024].

TEJWANI, Rahul, 2014. Sentiment Analysis: A Survey. *ArXiv* [online]. Retrieved from : https://www.semanticscholar.org/paper/3a01306cbd74d82b134bd961a38d5cf2a8cfa1d9 [accessed 5 May 2024].

TURING, Alan M., 2009. Computing Machinery and Intelligence. In : EPSTEIN, Robert, ROBERTS, Gary and BEBER, Grace (eds.), *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, pp. 23–65. Dordrecht : Springer Netherlands. ISBN 978-1-4020-6710-5. DOI 10.1007/978-1-4020-6710-5_3.

VASWANI, Ashish et al., 2023. *Attention Is All You Need*. arXiv:1706.03762. arXiv. arXiv:1706.03762. DOI 10.48550/arXiv.1706.03762. arXiv:1706.03762 [cs]

VISHESH, 2020. Customer decision-making process and the effect of marketing on the final purchase decision. *Journal of Management Research and Analysis*. Vol. 5, no. 3, pp. 304–311. DOI 10.18231/2394-2770.2018.0048.

WALSH, Morgan, 2019. Sentiment Analysis of Netflix Originals vs Licensed Content. *Synthesio* [online]. 8 August 2019. Retrieved from : https://www.synthesio.com/blog/sentiment-analysis-netflix-originals/ [accessed 10 August 2024].

What are Recurrent Neural Networks? | IBM, 2021 [online]. Retrieved from : https://www.ibm.com/topics/recurrent-neural-networks [accessed 22 July 2024].

What are Transformers? - Transformers in Artificial Intelligence Explained - AWS, *Amazon Web Services, Inc.* [online]. Retrieved from : https://aws.amazon.com/what-is/transformers-in-artificial-intelligence/ [accessed 23 July 2024].

What is Fine-Tuning? | IBM, 2024 [online]. Retrieved from : https://www.ibm.com/topics/fine-tuning [accessed 7 August 2024].

ZHANG, Lei, WANG, Shuai and LIU, Bing, 2018. Deep learning for sentiment analysis: A survey. *WIREs Data Mining and Knowledge Discovery*. Vol. 8, no. 4, p. e1253. DOI 10.1002/widm.1253.

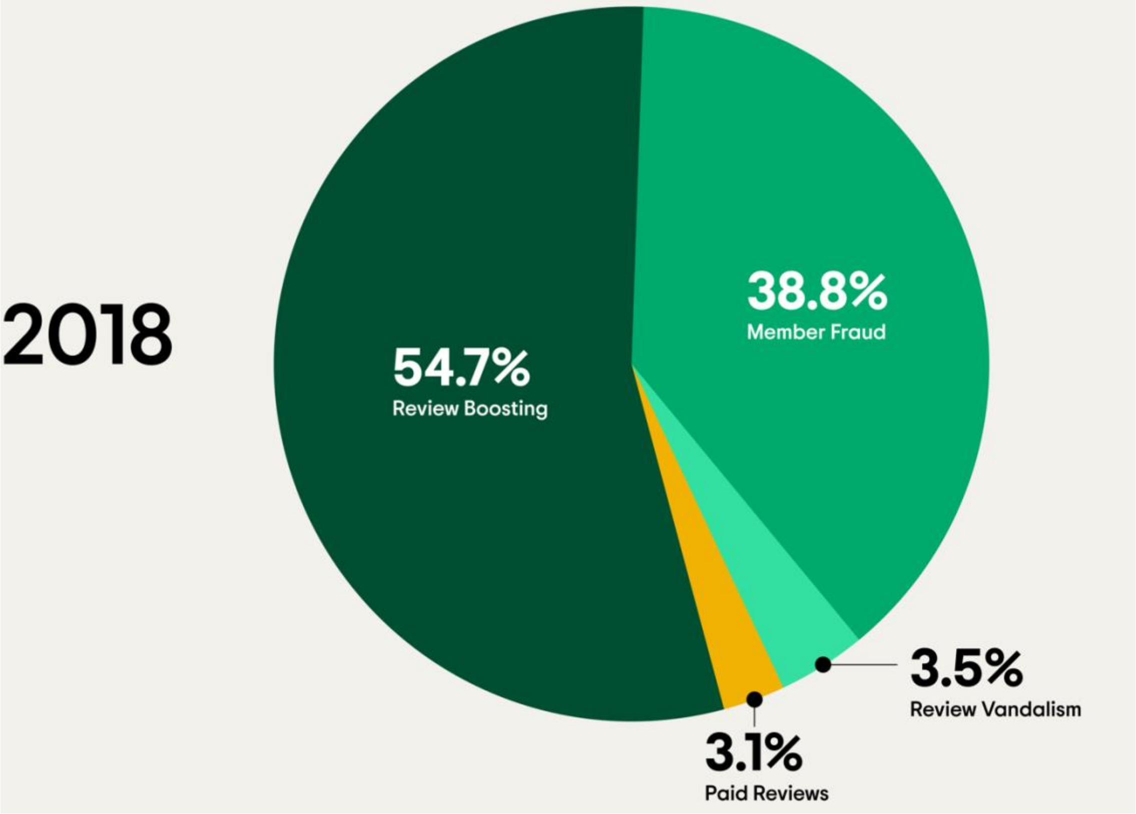
마스킹| 패딩 마스크(Padding Mask), 룩 어헤드 마스킹(Look-ahead masking), [online]. Retrieved from : https://velog.io/@cha-suyeon/마스킹-패딩-마스크Padding-Mask-룩-어헤드-마스킹Look-ahead-masking [accessed 26 July 2024].

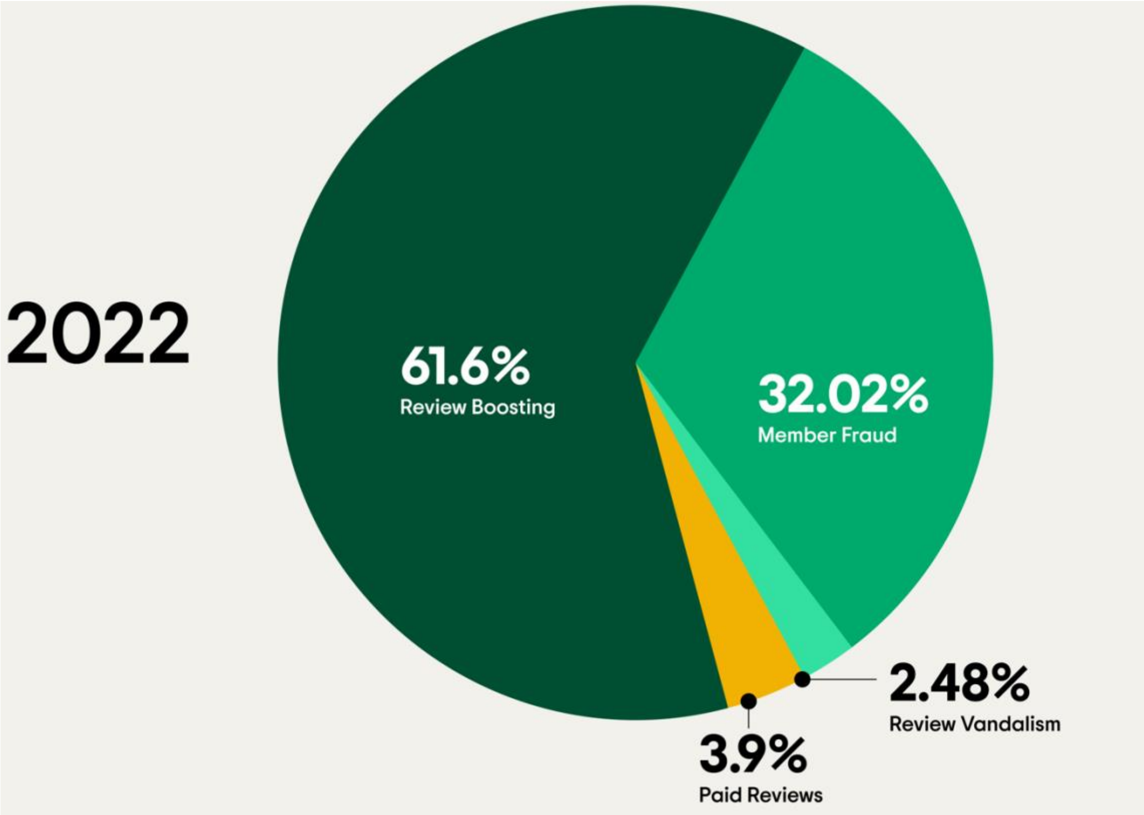
Appendix 1 - Fake Reviews

(*2023 Tripadvisor Review Transparency Report*)

A fake review has been submitted by someone biased in some way and/or did not have personal experience with the business they reviewed. This includes, but is not limited to:

- **Review boosting:** occurs when someone connected to the business, such as an owner, employee, or family member, posts a positive review.
- **Review vandalism**: when someone connected to a competing business submits a deliberately malicious review about a business to unfairly lower its ranking position or discredit it in some way.
- **Paid reviews**: when a business employs the services of an individual or a firm to boost its ranking position on Tripadvisor with positive reviews.
- **Member fraud:** when a user knowingly and/or maliciously submits a review violating our guidelines, potentially with the intent of affecting a property's ranking, but does so independently and not influenced by a property listed on Tripadvisor.





(*How Amazon is using AI to spot fake reviews*)

“Fake reviews aim to mislead customers by providing biased and unauthentic information,” said Josh Meek, senior data science manager on Amazon’s Fraud Abuse and Prevention team. Many customers rely on real reviews when deciding what to buy, and brands need Amazon to effectively find and remove fake reviews. Amazon works hard to monitor and enforce its policies to ensure reviews reflect the opinions of actual customers and to protect honest sellers.

To combat fake reviews, Amazon uses advanced AI technologies to stop millions of suspected fraudulent reviews, manipulated ratings, and fake accounts before customers can see them. They analyze a lot of data, such as whether sellers have paid for ads (which can lead to more reviews), customer reports of abuse, and unusual behaviour patterns. They also use natural language processing to look for signs that a review might be fake or incentivized with rewards like gift cards. Deep learning techniques help identify complex patterns and suspicious activity.

“The difference between real and fake reviews can be hard to see for people outside of Amazon,” Meek said. “For example, a product might get many reviews quickly if a seller is advertising or if it’s a good product at a good price. Alternatively, a customer might think a review is fake just because it has poor grammar.” By using advanced technology and unique data, Amazon can better identify fake reviews, looking deeper than just obvious signs of fraud to understand the connections between bad actors.

Appendix 2 - Encoder-decoder Attention Mechanism

(Malingan 2023)

To understand the attention mechanism in detail, it is essential to grasp Sequence-to-Sequence models like LSTMs and GRUs.The first paper that introduced the idea of the attention mechanism proposed an encoder-decoder model with an additive attention mechanism.

Consider a machine translation example where (x) denotes the source sentence with a length of (n) and (y) denotes the target sequence with a length of (m).

$$[x = [x_1, x_2, \dots, x_n]]$$

$$[y = [y_1, y_2, \dots, y_m]]$$

A bidirectional sequence model used for this task will have two hidden states: the forward and the backward hidden states. In the original paper, a simple concatenation of these two hidden states represents the encoder state. This way, both preceding and following words can be used to compute the attention of any word in the input.

In a standard encoder-decoder model, only the last hidden state of the encoder represents the encoder state. The decoder network's hidden state is:

$$[s_t = f(s_{t-1}, y_{t-1}, c_t)]$$

where (t) denotes the length of the sequence and (c_t) is the context vector (for each output (y_t)), which is the sum of the hidden states of the input sequence (h_i), weighted by alignment scores.

$$[c_t = \sum_{i=1}^{n} \alpha_{t,i} h_i]$$

How is this alignment score, which acts as the weight, calculated? The alignment score is parameterised by a single feed-forward neural network that is trained along with the other parts of the model.

The alignment model gives a score $(\alpha_{t,i})$ for each pair of input (i) and output at position (t), $((y_t, x_i))$based on the relevance. The set of $(\{\alpha_{t,i}\})$ the weights indicate how much each source hidden state should attend to each output state.

$$[\alpha_{t,i}=\mathrm{align}(y_t,x_i)=\frac{\exp\left(\mathrm{score}(s_{t\text{-}1},h_i)\right)}{\sum_{i'=1}^{n} \exp\left(\mathrm{score}(s_{t\text{-}1},h_{i'})\right)}]$$

The feed-forward neural network learns this relevance/alignment score, and its hidden state is passed through a softmax function to obtain the probabilities.