Foundation models in affective computing

Mostafa Mahmoud Amin

Angaben zur Veröffentlichung / Publication details:

Amin, Mostafa Mahmoud. 2025. “Foundation models in affective computing.” Augsburg: Universität Augsburg.

Foundation Models in Affective Computing

DISSERTATION

for the attainment of the degree of
Doctor of Engineering (Doktor-Ingenieur)
at the
Faculty of Applied Computer Science
of the
University of Augsburg

by

Mostafa Mahmoud Amin

Earlier known as: Mostafa Mahmoud Amin Ismail Mohamed

2025

Advisor:	Prof. Dr.-Ing. habil. Björn W. Schuller
Second Examiner:	Prof. Dr. Annemarie Friedrich
Date of submission:	14.08.2024
Date of oral exam:	24.01.2025

Acknowledgments

Throughout my doctoral research journey, I was very fortunate to work on many interesting problems with very wide scopes of interest, and meet many talented researchers.

First of all, I would like to express my deepest gratitude to my supervisor, Prof. Dr. Björn W. Schuller, for his continuous mentoring and guidance throughout my doctoral research journey, always providing new research insights and avenues, providing support and encouragement during difficult times, and insisting on the highest standards of academic excellence.

I would like to sincerely thank my friend Mina A. Nessiem for his support as a friend, business partner, co-author, and companion along the doctoral journey, as well as for his initiative, which enabled the beginning of this journey, as industrial doctoral candidates.

I want to deeply acknowledge the support and funding of SYNCPILOT, by allowing me to embark on this entire doctoral research journey during my employment there; without this support, this research journey would not have been possible.

Special thanks go to all my co-authors, especially Prof. Dr. Erik Cambria and Dr. Rui Mao, who provided me with very productive research collaboration opportunities that helped in progressing my research. Also, special thanks go to Dr. Anton Batliner, who mentored me immensely in my first journal article, Dr. Alice Baird, Dr. Lukas Stappen, Dr. Shahin Amiriparian, Manuel Milling, who supported me in getting started and throughout my doctoral research journey.

Last but not least, I would like to express my gratitude to my wife, Emese, who supported and encouraged me immensely with all the hardships, stress, and frustration I endured during my doctoral journey, while also sharing in the joy during the good times. I want to express my heartfelt gratitude to my family for their unwavering love, faith, encouragement, and the effort they invested in helping me become a successful person. I also express my gratitude to my friends for their support, especially Zaza, Omar Kassem, and Younis.

Mostafa M. Amin

Abstract

Foundation models, particularly Large Language Models (LLMs) like ChatGPT and its underlying Generative Pre-trained Transformer (GPT) models, have transformed the landscape of Natural Language Processing (NLP) and offer new avenues for affective computing. This dissertation explores the integration of the GPT foundation models with affective computing tasks to enhance the bridging between human emotions and advanced Artificial Intelligence (AI) technologies. This thesis contributes to the field by introducing the "prompting paradigm", a novel computational framework that leverages foundation models to solve affective computing problems; this paradigm provides the skeleton for the main studies underlying this work and the basis for future foundation models approaches to affective computing.

Key findings of this work indicate that LLMs exhibit emergent affective computing capabilities, offering nuanced emotional insights and improved task performance. A follow-up study is conducted to explore a wide range of affective computing tasks, grouped into three main categories: sentiment-based problems, psychology-based problems, and problems with implicit signals. GPT models, including GPT-3.5 and GPT-4, show the most superior performance on psychology-based problems, compared to specialised models. Furthermore, GPT models have shown strong performance in sentiment-based problems; however, specialised NLP methods still exhibit the most superior performance in many cases. Finally, GPT models perform poorly on problems with implicit signals. The GPT models are compared against NLP baselines, featuring different generations of NLP methods, namely using Bag-of-Words, Word2Vec, and RoBERTa features. The sentiment-based problems are sentiment analysis, opinion extraction, aspect extraction, aspect target polarity classification, subjectivity detection, sentiment intensity ranking, and emotions intensity ranking. The psychology-based problems are toxicity detection, suicide-tendency detection, and well-being and stress detection. The implicit signals problems are sarcasm detection, personality assessment, and engagement measurement.

The nuances of the paradigm are also explored in other studies. One of the studies conducts a comprehensive analysis of prompt sensitivity and sampling strategies, highlighting the importance of precise prompt engineering to optimise model outputs. Key prompting insights are using con-

servative sampling, and instructing the model to behave as an expert or to conduct step-by-step problem solving before giving a final answer, known as Chain-of-Thoughts (CoT). Furthermore, novel fusion strategies that combine outputs from LLMs with traditional NLP methods demonstrate performance improvements. A key idea to employ this is using verbose responses from LLMs, then further analysing them with NLP methods instead of simple text parsing. This assists in acquiring scores that can be fused easily with traditional NLP methods.

Furthermore, ethical considerations and compliance with regulatory frameworks are addressed to ensure the responsible deployment of AI technologies in affective computing applications, in alignment with the European Union AI Act.

By exploring the intersection of foundation models and affective computing, this research lays the groundwork for future studies, focusing on refining methodologies for the use of foundation models within affective computing. This assists in maximising the impact of affective computing on real-world applications while considering societal impacts.

Contents

List of Publications

Google Scholar profile: scholar.google.com/citations?user=WqUuu9oAAAAJ
ORCID profile: orcid.org/0000-0002-3175-2817

Journal Articles:

- **M. M. Amin**, E. Cambria, and B. W. Schuller, "Will Affective Computing Emerge from Foundation Models and General AI? A First Evaluation on ChatGPT," *IEEE Intelligent Systems*, pp. 15–23, 2023.

  Impact factor: 6.4.

- **M. M. Amin**, R. Mao, E. Cambria, and B. W. Schuller, "A Wide Evaluation of ChatGPT on Affective Computing Tasks," *IEEE Transactions on Affective Computing*, pp. 1–9, 2024.

  Impact factor: 11.2.

- **M. M. Amin**, E. Cambria, and B. W. Schuller, "Can ChatGPT's Responses Boost Traditional Natural Language Processing?," *IEEE Intelligent Systems*, pp. 5–11, 2023.

  Impact factor: 6.4.

- **M. M. Mohamed**, M. A. Nessiem, A. Batliner, C. Bergler, S. Hantke, M. Schmitt, A. Baird, A. Mallol-Ragolta, V. Karas, S. Amiriparian, and B. W. Schuller, "Face mask recognition from audio: The MASC database and an overview on the mask challenge,". *Pattern Recognition*, pp. 108361, 2022.

  Impact factor: 8.

Conference Proceedings:

- **M. M. Amin**, and B. W. Schuller, "On Prompt Sensitivity of ChatGPT in Affective Computing," in *Proceedings of Affective Computing & Intelligent Interaction (ACII)*, (Glasgow, Scotland), pp. 203–209, IEEE, 2024.

- **Best Student Paper Award.** M. A. Nessiem, **M. M. Mohamed**, H. Coppock, A. Gaskell, and B. W. Schuller, "Detecting COVID-19 from Breathing and Coughing Sounds using Deep Neural Networks," in *International Symposium on Computer-Based Medical Systems (CBMS)*, (Aveiro, Portugal), pp. 183–188, IEEE, 2021.
- M. A. Nessiem, **M. M. Amin**, and B. W. Schuller, "SMILENets: Audio Representation Learning via Neural Knowledge Distillation of Traditional Audio-Feature Extractors," in *International Conference on Frontiers of Signal Processing (ICFSP)*, (Corfu, Greece), IEEE, 2023.

Book Contributions:

- M. A. Nessiem, H. Coppock, **M. M. Mohamed**, and B. W. Schuller, "Artificial Intelligence in COVID-19," in *Omics approaches and technologies in COVID-19*, chapter 16, pp. 255–273, Academic Press, 2023.

Preprints:

- B. W. Schuller, A. Mallol-Ragolta, A. P. Almansa, I. Tsangko, **M. M. Amin**, A. Semertzidou, K. Christ, and S. Amiriparian, "Affective Computing Has Changed: The Foundation Model Disruption," *arXiv preprint arXiv:2409.08907*, 2024.
- **M. M. Amin** and B. W. Schuller, "Normalise for Fairness: A Simple Normalisation Technique for Fairness in Regression Machine Learning Problems," *arXiv preprint arXiv:2202.00993*, 2022.
- **M. M. Mohamed**, M. A. Nessiem, and B. W. Schuller, "On Deep Speech Packet Loss Concealment: A Mini-Survey," *arXiv preprint arXiv:2005.07794*, 2020.
- **M. M. Mohamed** and B. W. Schuller, "ConcealNet: An End-to-end Neural Network for Packet Loss Concealment in Deep Speech Emotion Recognition," *arXiv preprint arXiv:2005.07777*, 2020.
- **M. M. Mohamed** and B. W. Schuller, ""I have vxxx bxx connexxxn!": Facing Packet Loss in Deep Speech Emotion Recognition," *arXiv preprint arXiv:2005.07757*, 2020.

Part I

Introduction

Chapter 1

Introduction

Large Language Models (LLMs) have recently experienced a very rapid growth spanning an extremely wide range of applications [5]. The applications include Neural Machine Translation (NMT) [6], Named Entity Recognition (NER) [7], and Sentiment Analysis [8]. The superiority of LLMs created a new paradigm for predictive modelling; the paradigm utilises general-purpose foundation models [9] by way of prompting instead of training specialised machine learning models for each problem. Questions regarding the practicality of such a paradigm arose, questioning the effectiveness of this paradigm in various domains. The development of Artificial Intelligence (AI) and affective computing always went hand-in-hand, where typically AI was leading the development. The discovery of new AI techniques often led to better learning techniques, which were leveraged by affective computing for better methods. On the other hand, affective computing often presented problems that necessitated the further development of AI techniques.

1.1 Motivation

1.1.1 History of Foundation Models

Language Models (LMs) model a language by predicting how probable a given string to be within a language [10]. This is typically achieved in an auto-regressive fashion, on a token-by-token basis, namely, by predicting the conditional probability of a token given all its previous tokens. A token, in this context, refers to an individual unit of text that cannot be broken down further, which is usually one of three possibilities: characters, words, and subwords, where subwords are just parts of words. The conditional probabilities predicted by LMs can be utilised to generate text on a token-by-token basis by sampling tokens in a mechanism that corresponds to their predicted probabilities [11]. With the advancement of these predictions, it turns out that LMs are capable

of generating coherent and insightful texts. Another related problem is the Sequence-to-Sequence modelling [12], utilising an encoder and a decoder, which learns how to map a string of tokens to another string; a typical use-case of this is NMT. This is similar to employing an LM, except that the generated tokens are conditioned on some input tokens.

LMs first appeared in a very primitive format based on n-gram models, by utilising n-grams frequencies for making predictions [10]. However, this had very limited capabilities to actually model languages for complex applications. Such modellings still proved useful in providing features in some applications but not for text generation. The popularisation of Recurrent Neural Networks (RNNs) later on, mainly based on Long Short-Term Memory (LSTM) [13] and Gated Recurrent Units (GRUs) [14], sparked a revolution in LMs. RNNs proved to be effective in capturing deeper time dependencies between the tokens and also having generalisation abilities regarding the tokens; in other words, it did not necessarily need to learn about every possible sequence of tokens, but it could reason and generalise about sequences. Such RNNs could model the text on a syntactical basis, meaning that they typically produced grammatically correct sentences but without exhibiting proper reasoning. An idea that showed great promise to this paradigm was attention, which enables the decoder of Sequence-to-Sequence models to learn a function that decides which tokens are important to pay attention to while generating each token [15]. This enabled models to process longer sequences and avoid a bottleneck of information compression. Afterwards, the discovery of the Transformer architecture [16] has revolutionised the language models and Sequence-to-Sequence modelling. The architecture employs crucial components that resolve many bottlenecks in earlier architectures based on RNNs. Some of these key components are: (i) Multi-head Attention (MHA) enabling attention for multiple concepts at a time, instead of only one; (ii) Self-Attention enabling a form of key-value retrievals. (iii) The flat sequential architecture enables direct connections between each pair of tokens. (iv) Flat sequential architecture, enabling parallel backpropagation steps for different time steps of a long sequence, replacing the linear time dependencies by RNNs.

The Transformer architecture sets the initial point for having reasoning capabilities due to the refined attention mechanism and direct reasoning between any pair of tokens while enabling scaling in terms of model size and sequence lengths. This allowed training of language models at scale, the early examples of this are the Bidirectional Encoder Representations from Transformers (BERT) [17] and Generative Pre-trained Transformer (GPT) models. With the launch of the GPT-3 emerged the few-shot learning capabilities of LLMs [18]. These can be observed when an LLM takes a text including several examples of solving a problem and is then asked to complete the answer part for a given new example. This created a new paradigm of using Machine Learning (ML) models by using foundation models with proper prompting to solve problems. The promising results of this paradigm increased with the scaling of LLMs, which was demonstrated to cause

LLMs to gain emerging skills [19].

The last cornerstone of foundation models is training LLMs to generate text resembling a dialogue between a user and an assistant, where the assistant is following the instructions of the user. The main two techniques to achieve this goal are Supervised Fine-Tuning (SFT) [20] and Reinforcement Learning with Human Feedback (RLHF) [21]. The SFT is executed by giving the ground truth forms of such dialogues, hence it learns to follow some instructions. However, SFT cannot be very effective once the model generates unintended trajectories of text. RLHF fixes this issue by allowing the model to explore a wider space of answers and learning from their estimated reward, where the reward is estimated by a model that learns from human annotations. This also allows aligning the LLMs with several criteria.

1.1.2 History of Affective Computing

Affective computing is an interdisciplinary field at the intersection of computer science, psychology, and cognitive science, focused on enabling computers to recognise, interpret, and simulate human emotions [22]; [22] also introduced the term 'affective computing'. The evolution of the field reflects an understanding of the crucial role emotions play in human-human relationships, learning processes, and decision-making. Emotions are often considered more influential than IQ in various scenarios, contributing significantly to effective communication, collaboration, and rational learning in humans. This recognition has propelled the development of affective computing and sentiment analysis as key components in the advancement of artificial intelligence.

The primary objective of affective computing is to bridge the gap between human emotional experiences and machine understanding. By incorporating emotional intelligence into computing systems, affective computing aims to enhance human-computer interaction, making it more intuitive and responsive. This field encompasses a wide range of applications, including business analytics, politics, healthcare, education, and social media, where understanding and analyzing human emotions and opinions are increasingly valuable.

Historical Context and Evolution

The history of affective computing can be traced back to the late 1990s when researchers began exploring the possibility of endowing machines with the ability to understand human emotions. One of the foundational works in this area was the introduction of methods for predicting the semantic orientation of adjectives, which laid the groundwork for sentiment analysis [23]. Following this, the early 2000s saw significant advancements with the application of ML techniques to classify sentiment in text, particularly in domains such as movie reviews and customer feedback [24, 25].

The mid-2000s marked a period of growth for affective computing, with researchers developing methods for extracting opinion propositions and their holders from text [26]. This era also saw the emergence of benchmark datasets and shared tasks, such as the SemEval Aspect-Based Sentiment Analysis (ABSA) tasks, which provided standardized datasets and evaluation metrics, significantly advancing the field [27].

The late 2010s and early 2020s witnessed a paradigm shift in affective computing due to two core avenues (i) progress in ML methods, namely the introduction of Word2Vec, then Deep Learning (DL), and Transformer models, notably BERT, which improved the accuracy and context-awareness of sentiment analysis systems [28, 29], and (ii) deeper exploration of more nuanced affective computing tasks, like suicide tendency detection, well-being assessment, personality assessment, engagement measure and others. These progressions have enabled a more nuanced understanding and interpretation of human emotions, allowing for more granular insights into text by focusing on specific aspects or features of entities.

Core Components and Methodologies

Affective computing encompasses several core components and methodologies, which can be grouped into three main categories: sentiment-based signals, psychology-based signals, and implicit/latent signals.

Sentiment-Based problems These involve tasks that directly focus on analyzing sentiments and emotions within textual data:

- **Sentiment Analysis:** Also known as opinion mining, this task determines the sentiment polarity (positive, negative, or neutral) expressed in text. Sentiment analysis can be approached at various levels, including document-level, sentence-level, and aspect-level analysis [30, 31].

- **Aspect-Based Sentiment Analysis:** This task provides more granular insights by focusing on specific aspects or features of entities, determining the sentiment polarity associated with each feature [25, 28].

- **Opinion Extraction:** Involves identifying subjective expressions and associated opinions within text. This task includes subjectivity detection, opinion holder identification, and opinion target extraction. Foundational work by [32] introduced a gold-standard dataset for subjectivity classification, laying the groundwork for future research in this area.

- **Subjectivity Detection:** This focuses on identifying whether a piece of text is subjective (expressing opinions, feelings, or beliefs) or objective (stating facts). Subjective expressions are phrases that convey emotions, opinions, or speculations [33, 34].
- **Sentiment Intensity and Emotion Intensity Ranking:** These tasks aim to determine the degree or intensity of sentiment and specific emotions expressed in text. Techniques involve assigning numerical scores representing the intensity of sentiment or emotions, contributing to a more fine-grained analysis of human emotional expressions [35, 36].

Psychology-Based Problems These tasks focus on detecting psychological states and issues from textual data, such as mental health and abusive behaviour:

- **Toxicity Detection:** This task involves identifying the harmful or abusive language in the text. Techniques for detecting hate speech, offensive language, and other forms of toxicity have been developed, leveraging large-scale datasets and ML models [37–39].
- **Suicide-Tendency Detection:** Focuses on identifying signs of suicidal ideation in textual data. Early work explored Natural Language Processing (NLP) techniques for analyzing sentiment in suicide notes, and later research expanded to include social media data [40–42].
- **Well-being and Stress Assessment:** Involves identifying signs of well-being or stress in textual data, often utilizing linguistic styles and markers to assess psychological states [43, 44].

Problems with Latent Signals These tasks focus on detecting implicit traits and behaviours from text:

- **Sarcasm Detection:** Focuses on identifying sarcastic expressions in text. Techniques involve analyzing linguistic and contextual signals indicative of sarcasm, with applications in various domains such as social media and dialogue systems [45–47].
- **Big-Five Personality Assessment:** This involves identifying the Big-Five personality traits from text. Techniques include trait detection and scoring based on linguistic and contextual signals indicative of personality dimensions [43, 48, 49].
- **Engagement Measurement:** Involves predicting the level of engagement (e. g. , likes, shares, comments) that a piece of content will receive. Techniques include engagement prediction and virality assessment based on content characteristics and user behaviour [50, 51].

Affective computing continues to grow as a vital field of research, offering valuable insights and technologies that enhance human-computer interaction by understanding and responding to human emotions and behaviours. By integrating sophisticated machine learning models and interdisciplinary approaches, affective computing aims to make computers more empathetic and adaptive, ultimately improving the quality of interaction between humans and machines across various domains.

1.2 Research Questions

This work is primarily concerned with exploring the emerging capabilities of foundation models and how they impact a wide range of affective computing problems. With the rise of foundation models and their effectiveness, a new paradigm of AI has emerged, namely making predictions by way of prompting using generalist foundation models; this is introduced as the "prompting paradigm" in this work. This paradigm already shows great potential and is expected to grow bigger in the future. As a result, this work is concerned with laying the fundamentals for the interplay between foundation models and affective computing. Furthermore, some nuances of this paradigm are investigated as well, like fusion with traditional NLP models, and the impact of prompting on the effectiveness of foundation models and LLMs.

The research questions investigated in this work are:

- **RQ-1:** Do foundation models encompass emergent abilities for solving affective computing problems? This is investigated in Chapter 6 [1].
- **RQ-2:** What are the properties of the affective computing problems, where foundation models excel or fall behind? This is examined Chapter 7 [2].
- **RQ-3:** How sensitive are foundation models to the prompt templates used, given the major impact of prompts? This is analysed in Chapter 8 [3].
- **RQ-4:** Can both the prompting paradigm and the traditional paradigm of specialised models benefit from each other? This is explored in Chapter 9 [4].

1.3 Contributions

The contributions of this work are given as follows:

- Introduction of the "prompting paradigm", in Chapter 5, which is a computing paradigm based on foundation models and prompting for utilisation of foundation models to make

predictions. In this work, this is primarily used in affective computing problems; however, it can be applied to other scopes.

- Conducting initial studies to explore the emerging capabilities of foundation models in affective computing problems. The initial study, in Chapter 6, compared the performance on three affective computing problems, with various traditional NLP techniques, namely, Word2Vec, Bag-of-Words (BoW), and utilising feature from RoBERTa LLM.
- Extending the initial results with a bigger scale and more systematic process on a wide range of 13 affective computing problems, using GPT-3.5 and the State-of-the-Art (SOTA) model GPT-4, to show the effectiveness of the new paradigm, in Chapter 7. The affective computing problems are sentiment analysis, opinion extraction, aspect extraction, aspect polarity classification, subjectivity detection, sentiment intensity ranking, emotions intensity ranking, toxicity detection, suicide tendency detection, well-being assessment, sarcasm detection, personality assessment, and engagement measurement.
- Introducing a method for evaluating prompting and the sensitivity due to prompting and sampling. This is crucial since it is the main factor that decides how predictions are made, given an underlying foundation model; this is introduced in Chapter 8.
- Conducting an analysis of the input prompt templates utilised to generate the predictions, using the method in Chapter 8.
- Conducting a sensitivity analysis on the effects of sampling parameters, shown in Chapter 8.
- Investigating the added benefits of merging the new prompting paradigm with older existing NLP methods and how to do this effectively. This is in Chapter 9.
- Conducting a discussion regarding the ethical and societal impacts of foundation models and affective computing, and mentioning highlights of the EU AI Act, which regulates AI systems within the EU, in Chapter 10.

1.4 Outline

The outline of the thesis is given as follows:

Chapter 2 explains the background of ML, the different paradigms and models that are commonly utilised in the pre-foundation models era. Furthermore, the Transformer architecture is also explained, which is the basis behind LLMs, and how this morphed into the modern foundation models.

Chapter 3 explores the background of the different affective computing problems and the corresponding problem statements.

Chapter 4 explores the datasets used in all the studies.

Chapter 5 presents the new paradigm of predictive learning, namely the *prompting paradigm*, which outlines a general framework for using foundation models as models to predict affective signals. The studies generally align with the presented framework.

Chapters 6 to 9 constitute the main four studies corresponding to the aforementioned four research questions, introducing the prompting paradigm and how foundation models can be effectively used for solving affective computing problems.

Chapter 10 presents an ethical discussion about the impact of foundation models and affective computing, and the EU AI Act and its role in regulating general-purpose AI systems.

Chapter 11 concludes this work.

Part II

Background

Chapter 2

Machine Learning

ML is a branch of AI that focuses on building systems capable of learning from data, identifying patterns, and making decisions with minimal human intervention. There are four major paradigms for applying ML: Supervised Learning, Unsupervised Learning, Self-Supervised Learning, and Reinforcement Learning. ML, specifically DL, is applied in many domains, including but not limited to Computer Vision (CV), NLP and Speech Processing. Many classes of problems and domains are solved by ML in many different modellings.

While powerful, these models have limitations in handling complex, high-dimensional data, leading to more advanced techniques like Neural Networks (NNs) and DL.

2.1 Types of Machine Learning

Each type of machine learning has its own challenges and applications, and the choice among them depends on the specific problem and the nature of the data available.

2.1.1 Supervised Learning

Supervised Learning is the most prevalent form of ML. In Supervised Learning, an ML model is trained on a labelled dataset, which means the output or the answer is already known. The model learns a function $f : X \rightarrow Y$ that maps inputs X to corresponding outputs Y. Regression problems in Supervised Learning are where Y is a continuous variable, whereas classification problems are where Y consists of a discrete set of labels. Examples of regression problems include predicting house prices based on various features, whereas examples of classification problems are classifying emails into spam and non-spam.

There are various classical ML models employed in this type of ML [52], namely: Linear Regression (LinR), Logistic Regression (LogR), Decision Tree (DT), Random Forest (RF), Gradient Boosting (GB), and Support Vector Machines (SVM). These models are often transparent and interpretable, making them suitable for problems where understanding the decision-making process of the model is crucial.

2.1.2 Unsupervised Learning

In Unsupervised Learning, the training data is unlabelled, meaning there is only input data X, and the goal is to find underlying patterns or structures from the data without guidance. Unsupervised learning can be considered a technique to extract meaningful representations from unlabelled data.

Common Unsupervised Learning tasks include:

- Clustering of similar data points together, e. g. , using k-means or Gaussian Mixture Model (GMM) [52].
- Dimensionality reduction of the number of variables in data or feature spaces, e. g. , Principal Component Analysis (PCA) [52].
- Anomaly detection of data points that raise suspicious activity.

Unlike supervised learning, the main advantage of unsupervised learning is that it can be used on large amounts of data because it does not need an annotation process to extract ground truth labels to train on.

2.1.3 Self-Supervised Learning

Self-Supervised Learning (SSL) is a paradigm in ML, part of unsupervised learning, where the system generates its own labels from the input data. This approach is instrumental in scenarios, when labelled data is scarce or expensive to obtain while unlabelled data is very abundant. SSL circumvents the limitation of scarce labelled data by exploiting the inherent structure of the unlabelled data to learn meaningful representations.

In SSL, a pretext task is created where a portion of the data is hidden or transformed, and the model is trained to predict this missing or altered information. For instance, a common pretext task in image processing is to remove a part of an image and train the model to predict the removed portion [53]. In NLP, a typical SSL task might involve predicting a neighbouring word [54] or predicting the next word in a sentence [17], which is the typical setup in training LMs. After that,

SSL models can employ transfer learning through fine-tuning on a much smaller labelled dataset to achieve better results than training on the labelled dataset from scratch.

The key benefit of SSL is that it can leverage large volumes of unlabelled data, allowing the model to learn rich feature representations that can be fine-tuned for specific tasks with minimal labelled data. This approach has shown significant success in various domains.

2.1.4 Reinforcement Learning

Reinforcement Learning (RL) is one of the major types of ML. It is concerned with how agents ought to take actions to achieve goals in a complex environment, by maximising a notion of cumulative reward. The RL framework is compelling in settings where the solution is not immediately apparent and must be discovered through trial and error interactions with the environment. RL modellings generally contain four essential components, namely:

- $\mathcal{S}$ is a set of states,
- $\mathcal{A}$ is a set of actions,
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability that defines the probability of transitioning from one state to another given an action.

The main aim of RL is to establish a policy $\pi : \mathcal{S} \to \mathcal{A}$, which specifies the action the agent should take at each state to maximise the total expected cumulative rewards.

RL is about making sequences of decisions by learning to achieve a goal in a potentially complex and uncertain environment. The work in [55] demonstrates how Deep Reinforcement Learning (DRL) can be utilised in mastering very complex games like Dota 2. Another example is AlphaGo Zero [56], which demonstrates how an agent based purely on self-play can master the game of Go. Self-play was achieved by training a policy that tries to win against an opponent using the same policy to play. RL has also profound applications in NLP as demonstrated by instruction aligning in InstructGPT [21].

2.2 Support Vector Machines

SVM constitute a set of supervised learning methods for classification and regression [52, 57]. It works well on various datasets, particularly in cases where the dimensionality of the data is high.

Its ability to use different kernel functions makes it versatile and capable of handling non-linear relationships between features. Efficient implementations for linear SVMs are introduced by [58], based on gradient descent algorithms.

2.2.1 SVM for Classification

The primary objective of SVM classification is to find a hyperplane in an n-dimensional space (where n is the number of features) that distinctly classifies the data points. Many possible hyperplanes could be chosen to separate the two classes of data points. The optimal hyperplane is the one that represents the largest separation, or margin, between the two classes.

The hyperplane can be represented mathematically as:

$$\mathbf{w} \cdot \mathbf{x} + b = 0, \tag{2.1}$$

where $\mathbf{w}$ is the weight vector and b is the bias.

The objective is to maximise the margin between the hyperplane and the nearest data points from each class, referred to as support vectors. This can be formulated as:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \, \forall i, \tag{2.2}$$

where y_i are the labels of the training examples $\mathbf{x}_i$, taking values $+1$ or -1.

In cases where the data is not linearly separable, SVM uses a kernel function to map the input space into a higher dimensional space where a hyperplane can be used to separate the classes. Commonly used kernels include the linear, polynomial, Radial Basis Function (RBF), and sigmoid.

2.2.2 SVM for Regression

SVM can be used in regression tasks, known as Support Vector Regression (SVR). While using SVR, the goal is to find a function which deviates from the actual observed responses y_i by a value no greater than ϵ for each training point $\mathbf{x}_i$, and at the same time is as flat as possible.

Mathematically, SVR attempts to fit the error within a certain threshold:

$$|\mathbf{w} \cdot \mathbf{x}_i + b - y_i| \leq \epsilon. \tag{2.3}$$

To accommodate more flexible models, slack variables ξ_i and ξ_i^* are introduced to measure the

degree of margin violations:

$$\min_{\mathbf{w},b,\xi,\xi^*} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \tag{2.4}$$

subject to

$$y_i - \mathbf{w}\cdot\mathbf{x}_i - b \leq \epsilon + \xi_i, \tag{2.5}$$

$$\mathbf{w}\cdot\mathbf{x}_i + b - y_i \leq \epsilon + \xi_i^*, \tag{2.6}$$

$$\xi_i, \xi_i^* \geq 0 \,\forall i. \tag{2.7}$$

Here, C is a regularisation parameter that balances the trade-off between the flatness of f and the amount up to which deviations larger than ϵ are tolerated.

2.3 Neural Networks

NNs are a foundational element in the field of AI, particularly in ML. Inspired by the biological neural networks that constitute animal brains, these computational models are designed to simulate how humans think and learn. At their core, neural networks consist of layers of interconnected nodes, or *neurons*, each responsible for performing specific, simplistic computations. The *neurons* are typically grouped into subsequent layers. The input layer receives raw data, which is then processed through one or more hidden layers where the actual computation takes place, and finally, the output layer delivers the result. This structure enables the NN to recognise hierarchical patterns and features, hence making decisions based on higher-level information. The power of NNs lies in their ability to learn the parameters from data by adjusting them iteratively to maximise the performance over time by minimising a loss function $\mathcal{L}$. This process is known as the *training* process, generally based on the backpropagation algorithm [59].

2.3.1 Multi-Layer Perceptrons

Multi-Layer Perceptron (MLP) is one of the most basic forms of NNs [52]. Historically, it formulated the first form of NNs and assisted in formulating the backpropagation algorithm [59].

A demonstration of an MLP – the simplest form of a NN – with two hidden layers is demonstrated

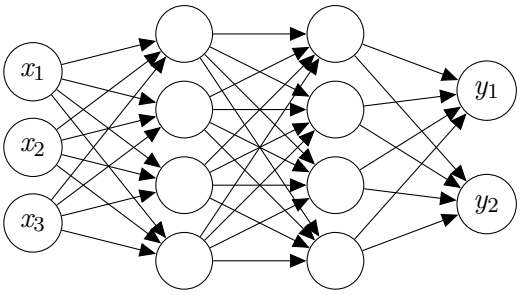


Figure 2.1: Demonstration of a simple NN with two hidden layers.

in Figure 2.1. Accordingly, the output $h_i^{(l)}$ of the i^{th} node at the l^{th} hidden layer is given by:

$$z_i^{(1)} = \sum_k x_k w_{ki}^{(1)} + b_i^{(1)}, \tag{2.8}$$

$$h_i^{(1)} = f^{(1)}(z_i^{(1)}), \tag{2.9}$$

$$z_i^{(2)} = \sum_k h_k^{(1)} w_{ki}^{(2)} + b_i^{(2)}, \tag{2.10}$$

$$h_i^{(2)} = f^{(2)}(z_i^{(2)}), \tag{2.11}$$

$$z_i^{(3)} = \sum_k h_k^{(2)} w_{ki}^{(3)} + b_i^{(3)}, \tag{2.12}$$

$$\hat{y}_i = f^{(3)}(z_i^{(3)}), \tag{2.13}$$

where all $w_{ij}^{(l)}, b_i^{(l)}$ are the parameters of the NN in this example, these can be thought of as the connections shown in Figure 2.1, $f^{(l)}$ are non-linear functions (called activation functions), and $\hat{y}_i$ is the predicted output corresponding to the label y_i.

A total error value $\varepsilon = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(y_i, \hat{y}_i)$ is calculated, where $\mathcal{L}$ is a *loss* function comparing both the inputs and outputs, then the gradients with respect to all parameters are calculated and used for updates.

2.3.2 Generalised Definition of Neural Networks

In many resources, NNs are defined very similarly to MLPs, mainly due to historical reasons, given that NNs started as MLPs. However, this definition is very limiting to understanding what NNs have become today, performing extremely complex operations like scanning images and modelling language. Thus, a generalised definition of NNs is given here, which is used implicitly by modern-day systems that build large NNs for practical purposes.

In order to give a generalised definition of NNs, two main building blocks are defined as follows:

- A tensor is an n-dimensional array of numbers, typically floating-point (decimal) numbers. That is a tensor T is a member $T \in \mathcal{T}$, where $\mathcal{T} = \cup_{m \in \mathbb{N}} \mathbb{R}^m$.
- A computational operation is a differentiable function $f : \mathcal{T}^n \rightarrow \mathcal{T}^m$ that transforms an ordered list of tensors into another set of tensors.

Tensors within the scope of an NN can be categorised into four main categories:

- placeholders $\mathbf{X}$ for the input to the NN,
- parameters $\mathbf{\Theta}$ of the given NN,
- *Intermediate* tensors that are output from the computational operations (used on any of the available tensors),
- A subset $\mathbf{Y}$ of the intermediate tensors, which is the set of output tensors of the NN.

Given the earlier definitions, NNs can be defined as a directed acyclic computational graph consisting of tensors as nodes, where different tensors are connected by assigning pairs of such sets to computational operations. Subsequently, data can flow within an NN for two possible purposes: training and inference. Furthermore, data can flow in forward or backward directions, also known as feed-forward and backpropagation, respectively.

The feed-forward pass is used in both training and inference, mainly by assigning the placeholders $\mathbf{X}$ some values, then running the tensors through the graph by computing the functions of the corresponding computational operations of the network along the way, until the values of the output tensors $\mathbf{Y}$ are obtained. These values are the predictions at inference time.

The backpropagation step is used in training and must be used after a feed-forward pass. After reaching the output tensors $\mathbf{Y}$, a computational operation with a loss function $\mathcal{L}$ is computed using $\mathbf{Y}$ while recording all the computational paths that reach $\mathbf{Y}$ on a tape. Eventually, according to the values of $\mathcal{L}$ and the computational paths recorded on the tape, all the parameters $\mathbf{\Theta}$ are updated accordingly. The updates are based on the gradients of $\nabla_{\mathbf{\Theta}}\mathcal{L}$ of a loss function $\mathcal{L}$ (computed on $\mathbf{Y}$) w. r. t. the parameters $\mathbf{\Theta}$.

An essential component of the evolution of NNs is primarily focused on engineering the architectures of the computational graphs of NNs to better process information depending on a given problem, without bottlenecks in information processing that hinder the capabilities of inferring deeper features.

An NN can be considered a sophisticated factory production line, where a conveyor belt transports items (data) through various stages, each representing a different layer in the network. These

stages are akin to computational operations in NNs, such as convolutional layers in Convolutional Neural Networks (CNNs) or attention layers in Transformers. As each item (a tensor representing multidimensional data like an image or text snippet) moves along the conveyor belt, it undergoes transformations at each stage. These transformations modify the attributes of the items - similar to how data is processed and altered in each layer of an NN. For example, in an image recognition task, an image tensor is progressively altered to emphasise certain features like edges or textures, analogous to the item being reassembled at each stage of the factory line.

The entire journey of the item on the conveyor belt parallels the forward pass in NNs, where data flows from the input to the output layer, being transformed along the way. Upon reaching the end of the line, the final product undergoes a quality check, analogous to the output evaluation by a loss function. If the product does not meet standards, feedback is sent back along the conveyor belt, instructing each station to make specific adjustments. This mirrors the backward pass in NNs, where gradients are calculated and used to update the parameters of the network $\boldsymbol{\Theta}$ to improve future outputs. Over time, this process of continuous feedback and adjustment enhances the efficiency of the factory, much like NNs learning and refining its accuracy through training.

2.3.3 Recurrent Neural Networks

RNNs are a class of neural networks adept at processing sequential data. They achieve this by maintaining a hidden state that captures information from previously seen elements in the sequence. A general common property is satisfying the constraint that $[y_t, h_t] = f(x_t, y_{t-1}, h_{t-1})$, processes a sequence by $x_{1 \dots T}$ by using the RNN sequentially, using the given property for all $t \in [1, T]$. RNNs often struggle with long-range dependencies due to the vanishing and exploding gradient problems during training. However, two main architectures overcome this limitation, namely LSTM [13] and GRU [14].

Long Short-Term Memory (LSTM) LSTM [13] units are a special kind of RNN architecture designed to combat the vanishing gradient problem in Vanilla RNNs. An LSTM unit comprises a cell, an input gate, an output gate, and a forget gate. The cell remembers values over arbitrary time intervals, and the three gates regulate the flow of information into and out of the cell.

The equations governing the LSTM unit are:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.14}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.15}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.16}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.17}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.18}$$

$$h_t = o_t * \tanh(C_t), \tag{2.19}$$

where σ is the sigmoid function, W and b are weights and biases, f_t, i_t, and o_t are the forget, input, and output gates, respectively, C_t is the cell state, and h_t is the hidden state.

2.3.4 Transformers

The Transformer architecture [16] revolutionised sequence processing, particularly in NLP. Transformers were initially introduced in the context of NMT [16]. The Transformer architecture deploys an encoder-decoder architecture, typical in sequence-to-sequence problems.

Transformers solve conditional auto-regressive text generation, meaning generating text in an auto-regressive fashion that is conditioned on some input. In the context of NMT, this generates a translated text sequence in a target language, conditioned on the input text in a source language. However, in the context of LMs, the conditional generation part is removed, and the problem becomes unconditional auto-regressive text generation. Unconditional generation entails just language generation without any conditions, except for the input text to start generating with, which will be the alternative way to condition text generation as discussed Chapter 8.

The encoder and decoder in the Transformer architecture consist of N identical blocks and dimensionality d, that contain several components, which are highlighted in the following sections.

Byte-Pair Encodings

Byte-Pair Encoding (BPE) is a tokenisation method primarily used to preprocess text data in NLP. Initially developed for data compression, BPE has found significant utility in the field of NLP, especially in the context of Transformer-based models.

BPE works by iteratively merging the most frequent pair of consecutive tokens (or characters in the beginning) in the dataset. This process continues for a predetermined number of merge operations. The result is a set of high-frequency tokens and a vocabulary of subword units, which

can represent common morphemes or word segments effectively. These subword tokens help manage vocabulary size, allowing the model to handle rare words or out-of-vocabulary terms by breaking them down into smaller known units. This technique has been pivotal in improving the performance of models like GPT and BERT, as it allows for more efficient and expressive representations of text data.

Attention Mechanisms

The attention mechanism is central to the Transformer model. It is defined as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V, \tag{2.20}$$

where Q, K, V are the query, key, and value matrices, respectively. The matrices K, V originate from one sequence, while Q can originate from another sequence.

The $Q \in \mathbb{R}^{T_q \times d}$ is a sequence of vectors originating from a sequence X of length T_X, where the vectors represent information queries to ask for at each time point. On the other hand, $K, V \in \mathbb{R}^{T_E \times d}$ are sequences of vectors originating from a sequence E of length T_E. K represents an identifier of the kind of information in the sequence at each time point, whereas V represents the actual corresponding information content (or value) at each time point.

Putting all of this together, the inner product $\frac{QK^T}{\sqrt{d}} \in \mathbb{R}^{T_X \times T_E}$ is a similarity matrix between all combinations of time points of the two sequences X, E, showing the similarity between information being asked for by the sequence X by queries Q and the kind of information they match in the sequence E identified by K. Applying softmax normalises these to probability vectors over the sequence E, constituting the attention weights matrix. Finally, multiplying the attention weights by V performs a weighted average over the information content averaged by their corresponding attention weights. Conclusively, the output of Equation (2.20) is a sequence corresponding to the sequence X containing different information from E while attentively matching time points from X with all the times points from E.

Equation (2.20) is typically applied in one of three ways, hence forming three types of attention:

- Cross-attention is when X and E are different, hence Equation (2.20) is making predictions for the sequence X, while conditioning the prediction on information in the sequence E.
- Self-attention is when X and E are the same sequence. Equation (2.20) would compute relations between different elements of the given input sequence.
- Causal Self-attention, which is self-attention done causally by modifying the attention weights

to be zeros at the points where Q can access future elements of the sequence corresponding to K, V, this is achieved using a lower-triangular matrix. In that case, the masking is achieved by masking $\frac{QK^T}{\sqrt{d}}$ to be $-\infty$, hence achieving a weight of 0 after applying the softmax function while ensuring the sum of weights remains 1.

Multi-head Attention

The MHA mechanism is described by Equation (2.20). It extends it by linearly projecting Q, K, V multiple times with different learned linear projections and then performing the attention function on each using Equation (2.20) independently (in parallel), followed by concatenation:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\, W^O \tag{2.21}$$

$$\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.22}$$

In these equations, W_i^Q, W_i^K, W_i^V, and W^O are parameter matrices.

The intuition behind MHA is simply applying multiple instances (H instances) of the attention mechanism described earlier, so that different heads can focus on specific concepts, then share this information eventually by the projection at the end in Equation (2.21).

Position-wise Feed-Forward Networks

Each layer in the Transformer contains a simple, fully connected feed-forward network, which is applied to each position separately and identically:

$$\text{FFN}(x) = f(xW_1 + b_1)\, W_2 + b_2, \tag{2.23}$$

where W_1, W_2 are weight matrices, b_1, b_2 are bias vectors, and f is a non-linear activation function. In the original Transformers [16], ReLU function was used as f, however, in GPT variants GELU is used [60].

Positional Encoding

Given that the attention mechanism by design does not pay attention to the positional nature of sequences. It simply calculates attention weights between all the time combinations of the given sequences. As a result, if a sequence is to have the same element but repeated at different positions, then the attention mechanism as-is would not distinguish between the different repetitions. This

motivates the need to employ additional information encapsulating information about the locations of the elements in the sequence.

To account for the sequence order, additional positional encodings PE corresponding to an input sequence of dimensionality $T \times d$ are used, given by:

$$PE_{t,2i} = \sin\left(\frac{t}{10000^{2i/d}}\right), \tag{2.24}$$

$$PE_{t,2i+1} = \cos\left(\frac{t}{10000^{2i/d}}\right), \tag{2.25}$$

where t is the position and i corresponds to the dimension. The values of t, i are set up to reach the same dimensionality as the input sequence.

The positional encoding can then be combined with a sequence of token embeddings using point-wise addition.

Layer Normalisation and Residual Connections

Given the deep nature of NNs, it can be hard to optimise such deep NNs due to the effects of vanishing gradients with high depth. To mitigate the side effects of this depth, two techniques are employed. The first is using residual connections [61] of adding the input of the sublayer to the output of the sublayer, where the sublayer here refers to the feed-forward network or the MHA. The second is layer normalisation [62], which normalises each dimension from the d dimensions of a given input to have 0 mean and 1 standard deviation. This is outlined by:

$$\text{LayerNorm}(x + \text{Sublayer}(x)). \tag{2.26}$$

The aforementioned formulation is the formulation in [16]. However, other formulations were adopted later on. One of them is the formulation adopted in GPT-2 [63]:

$$x + \text{Sublayer}(\text{LayerNorm}(x)). \tag{2.27}$$

Encoder and Decoder Stacks

The Transformer uses stacks of N encoders and N decoders blocks.

Encoder: Each layer in the encoder contains two sub-layers: a multi-head self-attention mecha-

nism and a feed-forward network.

$$a^{(i)} = \text{LayerNorm}(\text{MHA}(e^{(i-1)}, e^{(i-1)}) + e^{(i-1)}), \tag{2.28}$$
$$e^{(i)} = \text{LayerNorm}(\text{FF}(a^{(i)}) + a^{(i)}). \tag{2.29}$$

The original input x is projected with an embedding, combined with the positional encoding, and then used as $e^{(0)}$ in the above equations. The output of the encoder is $e^{(N)}$, which is a rich representation of the given sequence x. The sole aim of this representation is to condition the decoder by using cross-attention with $e^{(N)}$.

Decoder: Each layer in the decoder contains three sub-layers: a causal multi-head self-attention, then a cross-attention with the output of the encoder, then a feed-forward network.

$$a^{(i)} = \text{LayerNorm}(\text{MHA}_{\text{causal}}(d^{(i-1)}) + d^{(i-1)}), \tag{2.30}$$
$$b^{(i)} = \text{LayerNorm}(\text{MHA}(a^{(i)}, e^{(N)}) + a^{(i)}), \tag{2.31}$$
$$d^{(i)} = \text{LayerNorm}(\text{FF}(b^{(i)}) + b^{(i)}). \tag{2.32}$$

The decoder is fed a sequence after using a projection combined with a positional encoding on it as $d^{(0)}$, similar to the encoder.

GPTs and Decoder-only Transformers

For decoder-only Transformers like GPTs and other LLMs, the decoding is not conditioned on an encoder, hence the cross-attention is removed. Given this using the normalisation from Equation (2.27), the equations for the decoder become:

$$a^{(i)} = \text{MHA}_{\text{causal}}(\text{LayerNorm}(d^{(i-1)})) + d^{(i-1)}, \tag{2.33}$$
$$d^{(i)} = \text{FF}(\text{LayerNorm}(a^{(i)})) + a^{(i)}. \tag{2.34}$$

In this setup, the decoder is fed the ground truth output shifted such that the prediction of the decoder at time step t has only seen elements at time step t or earlier, due to the causal masking. Accordingly, the prediction of the decoder at time step t corresponds to the ground truth sequence at time step $t + 1$. For the training scenario, this is equivalent to training the decoder to simultane-

ously predict the next element of a given sequence for all the sequence prefixes. For the prediction scenario, the decoder block is then used to generate a sequence of length T by starting with the sequence with a single element y_0, then generating the T elements one-by-one by predicting the output $y_1, y_2, \cdots, y_T$ in an auto-regressive fashion.

BERT and RoBERTa, and Encoder-only Transformers

BERT is a significant advancement in applying Transformer models for natural language understanding [17]. BERT is an encoder-only Transformer. It stands out due to its bidirectional context understanding, achieved through a novel training approach known as Masked Language Modelling (MLM). In MLM, some percentage of the input tokens are masked randomly, and the model is trained to predict these masked tokens based on their context. This training allows BERT to develop a deep understanding of language context and word relationships.

Another distinctive feature of BERT is its ability to be fine-tuned for a wide range of NLP tasks with minimal task-specific adjustments. This versatility and powerful language understanding capabilities have led to its widespread adoption and numerous adaptations.

Robustly optimised BERT Approach (RoBERTa) is an optimisation and refinement of BERT [64]. RoBERTa modifies key hyperparameters in BERT and removes the Next Sentence Prediction (NSP) task from the training process, focusing solely on MLM. It also trains on a larger dataset for a longer duration with larger mini-batches. These changes result in significant performance improvements over BERT on various NLP tasks. There are two variants of RoBERTa of different sizes, namely *RoBERTa-base* and *RoBERTa-large*.

There are two ways of utilising transfer learning with RoBERTa models, either by fine-tuning or by using them as feature extractors without fine-tuning. In [1, 2, 4] only RoBERTa features rather than fine-tuning RoBERTa is used because the complexity of fine-tuning is significantly higher compared to using the features. The experimental designs in these works intentionally avoid increasing the complexity of specialised models (trained specifically for the tasks) to emphasise their accuracy and to objectively assess the limitations of the inspected LLMs; otherwise, it would be expected that generalist LLMs would fall behind. On the other hand, the feature extraction approach was shown to be effective in many setups [17, 65], especially when combined with LSTMs [66].

Both BERT and RoBERTa serve as foundational models for numerous subsequent developments in the field of NLP, highlighting the effectiveness of encoder-only Transformer architectures in understanding and processing natural language.

	this	**is**	**a**	**happy**	**story**	**very**	**sad**	**I**	**feel**	**today**
Documents with TF										
This story is a happy story	1	1	1	1	2	0	0	0	0	0
This is a very sad story	1	1	1	0	1	1	1	0	0	0
I feel sad today	0	0	0	0	0	0	1	1	1	1
Document count	2	2	2	1	2	1	2	1	1	1
Documents with TF-IDF										
This story is a happy story	0.5	0.5	0.5	1	1	0	0	0	0	0
This is a very sad story	0.5	0.5	0.5	0	0.5	1	0.5	0	0	0
I feel sad today	0	0	0	0	0	0	0.5	1	1	1

Figure 2.2: TF and TF-IDF approaches to produce BoW representations. The first three rows show the count of each word in the corresponding document, which is the TF representation. The row after this shows the number of documents where a word exists. The last three rows show an approach for TF-IDF, namely dividing the TF representation by the corresponding word counts. The columns for the stop words ‘is’ and ‘a’ are commonly dropped in practice.

2.4 Pre-foundation Models Natural Language Processing

2.4.1 Bag-of-Words Approaches

The BoW is a widely used technique in NLP and information retrieval. At its core, BoW is a method for simplifying text representation by disregarding grammar and even word order but keeping multiplicity. This model converts text into a numerical representation, commonly a vector, where each unique word corresponds to a value in the vector. The value at each index reflects the frequency or presence of the corresponding word in a document. This vector can be used within any of the typical ML models, like SVM. A demonstration of two BoW variants, namely Term-Frequency (TF) and Term-Frequency Inverse Document Frequency (TF-IDF), are depicted in Figure 2.2.

Due to the possible large vocabulary sizes, the vocabulary (e. g. , the header row in Figure 2.2) is restricted to the most common N words, where N is a tunable hyperparameter. The steps further that are usually taken include removing stop-words, namely words like ‘a’ and ‘is’, because they are too frequent in the language and hence do not carry a very unique meaning. Another step is normalising the frequencies by the number of words in the corresponding sentence. Finally, there are more advanced variants of BoW, namely TF-IDF, which normalises the score of each word by the number of documents it appears in (or the $\log$ thereof).

The BoW model has its limitations. It treats all words equally important, ignoring their context and grammatical relationships. For instance, “This is bad and not good” and “This is good and not bad” will have the same BoW representation despite having totally opposite meanings. Despite

these drawbacks, the simplicity of the model and its ease of implementation make it a popular starting point in text analysis.

2.4.2 Word2Vec Approaches

Word2Vec is a groundbreaking approach in the field of NLP that revolutionised how machines understand human language [54]. Unlike traditional models that encode words as isolated symbols, Word2Vec captures the context and semantic meaning of words by analysing their co-occurrence in large text corpora. The model uses a shallow NN to learn word associations from a large text corpus. Once trained, the model generates vectors for each word in the corpus, with the position of a word vector within a multidimensional space reflecting its relationship to other words. Words with similar contexts in the corpus are positioned closer in this vector space, effectively capturing their semantic similarity. This allows Word2Vec to facilitate tasks like finding synonyms and analogies and even predicting words in a sentence.

One of the critical strengths of Word2Vec is its ability to capture the meaning of a word based on its usage in different contexts. Two primary architectures are used in Word2Vec: Continuous Bag-of-Words (CBoW) and Skip-Gram. CBOW predicts a word based on its context, effectively using surrounding words to predict a target word. In contrast, Skip-Gram does the opposite; it uses a word to predict its surrounding context. This makes Skip-Gram particularly effective with smaller datasets and better at representing less frequent words. The ability of Word2Vec to transform words into vectors not only aids in understanding the semantic and syntactic relationships between words but also enables the application of mathematical operations to words. For example, by manipulating the vectors, one can find the word most similar to the result of the equation ‘king - man + woman’, which should closely match the vector for ‘queen’. Despite its simplicity, the impact of Word2Vec on NLP and ML has been profound, setting the stage for more advanced language models and embedding techniques.

2.5 Large Language Models

A very general definition of LMs is: LMs are models that can predict the probability of a string to occur within a language. However, with the popularisation of LLMs, the definition of LMs has shifted focus on auto-regressive LMs, because they can be used to generate text. Therefore, despite the general definition, LMs within this work refer to auto-regressive LMs. On the other hand, LLMs are just large-scale LMs based on the Transformer architecture; large-scale is not strictly defined, but it usually refers to LMs with more than one billion parameters.

2.5.1 Auto-regressive Language Models

LMs are any models that can model language by predicting how probable a string s will occur in the given language. This is typically modelled in auto-regressive fashion by predicting the following probability for all possible tokens τ in a vocabulary:

$$\forall \tau \cdot \mathbb{P}(x_t = \tau | x_1, \cdots, x_{t-1}), \tag{2.35}$$

hence, predicting the probability of a string consisting of a sequence of N tokens $\tau_1, \cdots, \tau_N$ by using the following equation:

$$\prod_{t=1}^{N} \mathbb{P}(x_t = \tau_t | x_1 = \tau_1, \cdots, x_{t-1} = \tau_{t-1}), \tag{2.36}$$

or equivalently by calculating what is known as the *perplexity* of the sequence:

$$\exp(-\frac{1}{N} \sum_{t=1}^{N} \log \mathbb{P}(x_t = \tau_t | x_1 = \tau_1, \cdots, x_{t-1} = \tau_{t-1})), \tag{2.37}$$

or shortly,

$$\exp(-\frac{1}{N} \sum_{t=1}^{N} \log \mathbb{P}(\tau_t | \tau_1, \cdots, \tau_{t-1})). \tag{2.38}$$

2.5.2 Generating Text using Language Models

Given an LM, it can be utilised to complete a given text, consisting initially of n tokens: $\tau_1, \tau_2, \cdots, \tau_n$. In order to generate tokens $\tau_{n+1}, \tau_{n+2}, \cdots, \tau_N$, a tree search can be performed to explore the search space of all possible completions. A string is expanded by one token at a time, checking all the possible tokens in the vocabulary of the given LM [11]. This continues until a maximum number of tokens is generated or a special token is encountered, a token denoting 'end-of-text' or 'end-of-sentence'. In the end, the branch with the highest probability is selected as the generated text, alternatively, the path with the lowest accumulated negative log-likelihood. An example of this expansion is depicted in Figure 2.3.

Executing this mechanism is obviously not feasible due to the exponential size of this search space. Hence, different pruning mechanisms are typically utilised to restrict the search space. LLMs typically outputs a list of values, known as logits, $l_1, l_2, \cdots, l_V$, where V is the vocabulary size. The softmax function is applied to these values to obtain the probabilities $p_1, p_2, \cdots, p_V$,

which is used as the probabilities in Equation (2.35), where:

$$p_k = \frac{\exp(l_k)}{\sum_{i=1}^{V} \exp(l_i)} \tag{2.39}$$

There are some techniques to prune such search trees in a reasonable way by manipulating the logits, obtaining modified probabilities, and then sampling a token with the modified probabilities. These techniques are investigated in [67], they are given by:

- **Greedy Search**: Always choose the highest probable next token. For example, in Figure 2.3, the expansion would always take the leftmost branch.

- **Beam Search**: Given a beam size B, the algorithm selects at each level the B branches with the highest cumulative probability up to that level, then expands only these branches [68]. In each of the subsequent levels, the best B are selected. In practice, the Negative Log-Likelihood (NLL) of the cumulative probability is used for numerical stability.

- **Random Sampling**: Sample one token at each level with sampling probabilities corresponding to the probabilities of the tokens. For example, sample one token from the children at each level with their corresponding probabilities as predicted by the model.

- **Temperature Sampling**: Given a parameter T, divide the logits of the probabilities by T before sampling, then normalise the probabilities, and then perform random sampling [69].

- **Top-k Sampling**: Given a parameter k, pre-select k tokens with the top probabilities, then normalise the probabilities of the selected tokens, and sample a token like random sampling or temperature sampling. For example, in Figure 2.3 with $k = 2$, the sampling would sample from the words 'like' and 'so' at the first level with ratio 27:12. After the word 'very' in the second level, it would sample from 'happy' and 'blessed' with ratio 35:9.

- **Nucleus Top-p Sampling**: Given a threshold parameter top-p, pre-select the tokens by selecting the smallest set of tokens with probabilities that add up to top-p or higher, then normalise the probabilities of the selected tokens, and sample a token like random sampling or temperature sampling [67]. For example, in Figure 2.3 with top-$p = 0.3$, the first level would sample from the two tokens 'like' and 'so' with ratios 27:12. In the second level after the word 'very', 'happy' will always be sampled since it is the only possibility because it has probability higher than top-p.

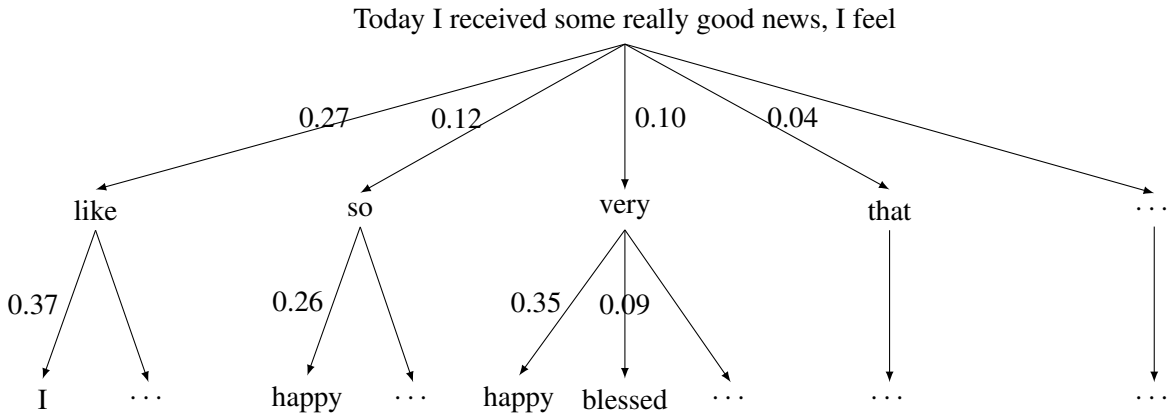


Figure 2.3: Expansion of how an LM continues the text "Today I received some really good news, I feel · · · ". As calculated by the LLM 'gpt-3.5-turbo-instruct-0914'.

2.5.3 Large Language Models as Few-shot Learners

With the publication of the GPT-3 model, a new concept was introduced, namely that LLMs are few-shot learners [18]. This means that, if an LLMs is given a text to complete, which includes a few examples of how to solve a specific task, then it figures out the task and solves it for a given instance. The expression *few-shots* means few guiding examples in the input; hence *one-shot* means a single guiding example. The expression *zero-shot* means no guiding examples are given, only the task description. The groundbreaking discovery is the fact that the LLMs can solve the tasks without being trained to do them in advance, and the guiding examples are only given during inference and not during training. This implies that LLMs acquire some emerging skills simply by scaling the models and the training data, without explicit training to develop such skills. This has been discovered to be a general capability of LLMs, namely that scaling LLMs results in them acquiring emerging skills without being trained to do [19]. Such emerging skills include NMT and text summarising.

As an example of translating a sentence, the model can be used to continue the following text:

‘‘‘
Der Staat muss die Würde aller Menschen schützen. →
The state must protect the dignity of all people.

Jeder Mensch hat das Recht zu leben. →
’’’

which can output the following text as a translation:
'Every person has the right to live.'

In this example, the model was never asked to translate, and it inferred this from the example.

The intuition behind why this property of LLMs exists can be explained in the following examples. For a given LLM to predict the next token, the model must reason about the text in the preceding text in some capacity. For example, in Figure 2.3, the model has reasoned that the next token has to be an adjective or an expression describing a feeling. Additionally, it reasons the given text demonstrates strong positive emotions, hence the feeling has to be a positive emotion, and needs to be emphasised. As a result, it predicts emphasis words like 'so', 'very' with high probabilities followed by a positive adjective, or it predicts the words 'like' or 'that', which start an expression describing a feeling, expression like "$\cdots$ like I am flying on cloud nine". Another example that was given is if a model is processing a thriller novel and after processing all the plot, it arrives at the point in the thriller where a major reveal is about to happen, namely by finishing a sentence like "the person who committed this crime is $\cdots$". A model that is successful in predicting the next token with the reveal of the person is a model that is somewhat capable of doing some detective work or at least understanding the plots of thriller novels. In conclusion, LLMs learn a projection of a world model that is taught through the data of the whole internet, which includes a large amount of the collective knowledge of humans; by learning to predict the following tokens, it learns to reason about this collective knowledge, hence developing some emerging skills to do so.

This property allowed the existence of a new paradigm in AI. Instead of training one model to solve a specific problem, one LLM can be trained and used directly to solve several tasks. The main bridge to utilise this paradigm is the prompting, namely engineering the initial seed text that generates the desired answer upon completion.

2.5.4 Aligning Large Language Models

LLMs such as InstructGPT [21] represent significant advancements in the field of NLP. However, these models often require alignment to ensure that their responses are accurate and concord with human values and intentions. Proximal Policy Optimisation (PPO), a reinforcement learning algorithm, plays an instrumental role in this alignment process.

The alignment of LLMs follows the framework [21, 70, 71]:

1. Train a *base LLM*, which is training an LLM to predict the next tokens, on a very gigantic scale. This is typically the most costly step in training LLMs.

2. Conduct SFT on the base LLM from step 1, to model the language of imaginary conversations between a user and an assistant.

3. Collect samples of generated conversations, generated by the model from step 2 or previous

iterations of this process, and let human annotators annotate different criteria.

4. Modify a copy of the trained LLM and fine-tune it to predict the rewards given by the human annotators.

5. Align the LLM from step 2 using RLHF techniques [21, 70], while using the reward model from step 4 as the reward system.

Supervised Fine-Tuning

SFT is a fundamental technique in training and aligning LLMs [18, 63, 72]. This process involves training the model on a labelled dataset, where the input-output pairs are provided explicitly. The primary advantages of SFT are:

- **Controlled Learning:** SFT allows precise control over the learning process by using curated datasets that exemplify desired behaviours and responses.
- **Baseline Establishment:** It provides a strong baseline for the performance of the model before applying more complex techniques like RLHF.
- **Consistency and Reliability:** By learning from high-quality examples, the model develops a consistent understanding of language tasks, ensuring reliable performance.

In the context of LLMs, SFT is typically the initial phase of training, providing a robust foundation upon which further alignment techniques are built, such as PPO; this is done by fine-tuning base LLMs to resemble texts of imaginary dialogues between a user and an assistant.

Overview of Proximal Policy Optimisation

PPO is a policy gradient method for reinforcement learning that optimises a policy by taking small incremental steps [73]. This approach is advantageous for several reasons:

- **Stability and Efficiency:** PPO balances the stability of Trust Region Policy Optimisation (TRPO) and the efficiency of simpler methods like vanilla policy gradient.
- **Simplicity of Implementation:** Compared to other algorithms like TRPO, PPO is more straightforward to implement, as it does not require complex operations like computing the Hessian matrix.

- **Adaptability:** The flexibility of PPO allows it to be applied to a wide range of environments, making it suitable for tasks like aligning LLMs.

The PPO algorithm aims to maximise the expected reward while ensuring the policy does not deviate significantly from the previous one [73], maintaining a balance between exploration and exploitation. The objective function of PPO can be described as:

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t, \text{ clip} \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right) \right]. \tag{2.40}$$

Here, θ represents the policy parameters, π denotes the policy, a_t and s_t are the action and state at time t, $\hat{A}_t$ is the advantage function, and ϵ is a hyperparameter that controls the extent of the clipping. The term $\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio between the new and old policies.

Application of PPO in Aligning InstructGPT

InstructGPT [21], a variant of GPT-3 tailored for following instructions more precisely, requires alignment to ensure it adheres to ethical guidelines and user intents. PPO facilitates this alignment through the following mechanisms:

1. **Reward System:** PPO refines the responses of InstructGPT by using a reward system, where desirable outputs are reinforced.
2. **Iterative Learning:** Through continuous interaction and feedback, PPO enables InstructGPT to learn from its mistakes and improve over time.
3. **Adaptation to User Feedback:** The adaptability of PPO allows InstructGPT to adjust its responses based on user feedback, hence deploying RLHF, ensuring better alignment with human values and expectations.

The integration of PPO in the training of InstructGPT exemplifies how advanced RL techniques can be leveraged to enhance the performance and alignment of LLMs. This not only improves the accuracy of the models but also ensures their adherence to ethical standards and user intentions. Approaches for scaling the training at this step are discussed in [74].

2.6 Ensemble Learning

Ensemble learning is a powerful ML paradigm where multiple models, often referred to as *base learners*, are combined to solve a particular problem. The fundamental premise of ensemble

learning is that a group of weak learners can come together to form a strong learner, often achieving superior performance compared to any individual model. This approach leverages the diversity among the models to reduce the risk of overfitting and improve generalisation.

2.6.1 Basic Methods of Ensemble Learning

The main strategies for creating ensemble models include averaging methods, majority voting, and boosting. Each method has its unique approach to combining the outputs of the base learners to make a final prediction.

Averaging Methods

Averaging methods take the mean of the predictions from all base learners. The primary assumption here is that the errors of the individual models are uncorrelated, which means that averaging will reduce the overall error. Mathematically, for a set of M base learners, each producing a prediction $\hat{y}_m$ for an input x, the ensemble prediction $\hat{y}$ is given by:

$$\hat{y} = \frac{1}{M} \sum_{m=1}^{M} \hat{y}_m. \tag{2.41}$$

This method is commonly used in regression problems where the output is continuous; however, it can also be used for classification problems by averaging the predicted probabilities, which is a continuous label.

Majority Voting

Majority voting is typically used in classification tasks. In this method, each base learner provides a class prediction, and the final prediction is the class that receives the most votes. If $\hat{y}_m^{(c)}$ represents the prediction of the m-th base learner for class c, the ensemble prediction $\hat{y}$ can be expressed as:

$$\hat{y} = \text{argmax}_c \sum_{m=1}^{M} \mathbf{1}\{\hat{y}_m = c\}, \tag{2.42}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function that returns 1 if the condition is true and 0 otherwise. This method is effective when the base learners are diverse and have varying strengths in predicting different classes.

Boosting

Boosting is an iterative method that focuses on training new models to correct the errors made by previous ones. One of the most popular boosting algorithms is AdaBoost [75], which adjusts the weights of incorrectly classified instances, making them more important for the subsequent models. The ensemble prediction is a weighted sum of the predictions of the base learners. Mathematically, for a sequence of models $\{f_m(x)\}$ with weights $\{\alpha_m\}$, the prediction $F(x)$ is:

$$F(x) = \sum_{m=1}^{M} \alpha_m f_m(x), \tag{2.43}$$

where α_m is computed based on the accuracy of the m-th model. Boosting can significantly improve the performance of the model by focusing on hard-to-classify instances.

2.6.2 Reasoning Behind Ensemble Learning

The effectiveness of ensemble methods can be explained through the bias-variance decomposition, which states that the error of a model can be decomposed into three parts: bias, variance, and random noise [52]. Ensemble methods primarily aim to reduce the variance component of the error, which is demonstrated below for the case of averaging.

When averaging the M base learners, then the variance of the averaged score can be given by [76]:

$$\text{Var}[\hat{F}(x)] = (\rho + \frac{1-\rho}{M})\text{Var}[\hat{f}(x)], \tag{2.44}$$

where ρ is the correlation between any pair of distinct predictors.

For the edge case of a perfectly correlated set of predictors, where $\rho = 1$, meaning that all models are the same, then the variance is the same as this one model, essentially having no benefit for multiple models. On the other hand, if the predictors have weak or no correlation $\rho \to 0$, or more predictors $M \to \infty$ are used, the variance becomes less than the expected variance of the individual base learners, acquiring more stable results that can generalise better.

2.7 Hyperparameter Optimisation

In ML, hyperparameters are parameters set prior to the learning process that significantly influence the performance of ML models. These differ from model parameters, which are learned during training. Examples include the learning rate for gradient-based algorithms, the number of layers

in a neural network, and the regularisation term in penalised models. Finding the optimal set of hyperparameters is critical, as suboptimal choices may lead to poor model performance.

Hyperparameter Optimization (HPO) is the process of selecting the best hyperparameters to achieve the best possible model performance on a validation dataset. Traditional methods include grid search and random search, which are effective but computationally intensive, particularly for DL models with large hyperparameter spaces. More efficient alternatives, such as evolutionary algorithms, bandit-based methods [77], and Bayesian optimisation (BO) [78], have gained prominence due to their ability to optimise with fewer evaluations.

Sequential Model-based Algorithm Configuration (SMAC) is a comprehensive BO package designed to handle various HPO tasks [79]. It supports different scenarios through its versatile framework, leveraging techniques like Gaussian Process (GP) [80] and RF [81] models to optimise hyperparameters efficiently.

1. SMAC4BB focuses on low-dimensional, continuous black-box functions, utilising GPs with acquisition functions such as Expected Improvement (EI) to optimise hyperparameters in continuous spaces.
2. SMAC4HPO extends traditional HPO to include algorithm selection through a Combined Algorithm Selection and Hyperparameter Optimization (CASH) approach, using RF models to handle complex conditional spaces.
3. For expensive tasks, SMAC4MF incorporates multi-fidelity optimisation, combining Hyperband techniques with RF models to manage resource-intensive DL models.
4. SMAC4AC addresses algorithm configuration, using aggressive racing and RF models to evaluate and optimise configurations across different problem instances.

2.8 Evaluation Metrics

Evaluation metrics play a crucial role in assessing the performance of machine learning models, providing quantitative measures of how well a model's predictions align with actual outcomes. This section describes several widely used evaluation metrics. Each metric serves a specific purpose, reflecting various aspects of model performance depending on the nature of the task.

Mean Absolute Error

Mean Absolute Error (MAE) measures the average magnitude of the errors in a set of predictions, without considering their direction [52]. It is the average over the absolute differences between

the actual and predicted values. MAE is less sensitive to outliers compared to Mean Squared Error (MSE). MAE and MSE are used in regression tasks as a metric or loss function for training NNs. The formula is expressed as:

$$\text{MAE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|, \tag{2.45}$$

where y_i and $\hat{y}_i$ are the true and predicted values, respectively.

Negative Log-Likelihood

NLL, also known as Binary Cross-Entropy (BCE), is an evaluation metric used for binary classification tasks. It measures the dissimilarity between the true labels and predicted probabilities [52]. Lower NLL indicates better model performance. NLL is typically used as a loss function for training NNs for binary classification tasks. The formula for NLL is:

$$\text{NLL}(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)], \tag{2.46}$$

where y_i is the true binary label and $\hat{y}_i$ is the predicted probability of the positive class.

Categorical Cross-Entropy

Categorical Cross-Entropy (CCE) is used in multi-class classification tasks, evaluating how well the predicted probability distribution aligns with the true distribution [52]. It is an extension of NLL to handle multiple classes. NLL is typically used as a loss function for training NNs for multi-label classification tasks. The equation is:

$$\text{CCE}(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic}), \tag{2.47}$$

where C is the number of classes, y_{ic} is the binary indicator (0 or 1) if class label c is the correct classification for observation i, and $\hat{y}_{ic}$ is the predicted probability that observation i belongs to class c.

Classification Accuracy

Classification Accuracy (ACC) is a straightforward metric for evaluating classification models [52]. It measures the ratio of correctly predicted instances to the total instances. Accuracy is a useful metric when the class distribution is balanced. ACC is typically used in classification tasks. The formula is:

$$\text{ACC}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i), \tag{2.48}$$

where $\mathbb{I}$ is the indicator function, which is 1 when $\hat{y}_i = y_i$ and 0 otherwise.

Unweighted Average Recall

Unweighted Average Recall (UAR) is an evaluation metric used in multi-class classification, particularly when classes are imbalanced [82]. UAR computes the average recall obtained on each class, treating all classes equally regardless of their frequency. UAR is typically used in classification tasks. The formula is:

$$\text{UAR}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{C} \sum_{c=1}^{C} \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c}, \tag{2.49}$$

where TP_c and FN_c are the true positives and false negatives for class c, respectively, and C is the number of classes.

Chapter 3

Affective Computing Problems

This chapter delves into brief histories and definitions of the affective computing problems used in the studies in this work.

3.1 Sentiment Analysis

Sentiment analysis, also known as opinion mining, focuses on determining the sentiment expressed in text. Early work in the late 1990s on predicting the semantic orientation of adjectives laid the foundation for sentiment analysis [23]. [83] introduced methods for computing sentiment orientation using point-wise mutual information.

The early 2000s saw significant advancements with [24] applying ML techniques to classify sentiment in movie reviews. [25] further extended sentiment analysis by focusing on mining and summarising customer reviews. Since then, classical ML has led the frontier for sentiment analysis [30, 84]. Later on, with the emergence of Word2Vec [54] and practical use of LSTMs [13], despite its early publication, sentiment analysis approaches have shifted towards DL [31]. [85] provide practical guides for sentiment analysis systems as well.

Problem Statement

Sentiment analysis involves determining the sentiment polarity or opinion polarity expressed in text, namely positive, negative, or neutral. Other modellings use only positive or negative.

The task can be approached at various levels:

1. **Document-Level Sentiment Analysis:** Classifying the overall sentiment of a document.

2. **Sentence-Level Sentiment Analysis:** Classifying the sentiment of individual sentences.

3. **Aspect-Level Sentiment Analysis:** Classifying sentiment towards specific aspects or features within the text.

In this work, *sentiment analysis* refers to *sentence-level sentiment analysis*. Document-level sentiment analysis is not handled in this work, whereas aspect-level sentiment is handled as an independent problem below, in Section 3.3.

3.2 Opinion Extraction

Opinion extraction focuses on identifying subjective expressions and the associated opinions within text. The foundational work by [32], in the late 1990s, introduced a gold-standard dataset for opinions and subjectivity classification, laying the groundwork for future research.

[26] contributed significantly by focusing on extracting opinion propositions and their holders. The mid-2000s saw further advancements with [86] developing methods to automatically learn extraction patterns for subjective expressions, and [87] providing detailed guidelines for annotating expressions of opinions and emotions.

Problem Statement

Opinion extraction involves identifying and extracting subjective expressions, including opinions, emotions, and attitudes, from text. This is closely related to subjectivity detection, explained in Section 3.4, with a crucial difference that *subjectivity detection* deals with sentence-level classification, where opinion extraction is word-based classification.

3.3 Aspect-Based Sentiment Analysis

ABSA emerged as an extension of sentiment analysis aimed at providing more granular insights into text by focusing on specific aspects or features of entities. One of the earliest works in this area proposed methods to extract product features and determine the sentiment polarity associated with each feature from customer reviews [25]. This foundational work laid the groundwork for future research in aspect extraction and sentiment classification.

In the mid-2000s, [88] introduced methods for identifying product features and opinions using syntactic patterns and statistical techniques. The mid-2000s saw significant advancements with

studies like [26] focusing on extracting opinion propositions and their holders, and [89] on extracting opinions, opinion holders, and topics from online news media.

The development of benchmark datasets and shared tasks, such as the SemEval Aspect-Based Sentiment Analysis tasks initiated by [27], significantly contributed to the field by providing standardised datasets and evaluation metrics. The late 2010s and early 2020s witnessed the integration of DL and Transformer models, with studies like [28] and [29] leveraging BERT for improved aspect-based sentiment analysis.

Problem Statement

ABSA involves two main sub-tasks:

1. **Aspect Extraction:** Identifying the specific aspects or features mentioned in a given text.
2. **Aspect Polarity Classification:** Determining the sentiment polarity (positive, negative, neutral) expressed towards each identified aspect.

3.4 Subjectivity Detection

Subjectivity detection focuses on identifying whether a piece of text is subjective (expressing opinions, feelings, or beliefs) or objective (stating facts). Early work in the late 1990s by [32] introduced a gold-standard dataset for subjectivity classification, laying the groundwork for future research.

Subsequent research in the early 2000s helped develop the field. [90] researched the effects of adjective orientation on sentence subjectivity, whereas [91] researched learning subjective language. The development of benchmark datasets and shared tasks, such as those introduced by [33,34] further contributed to the field, especially [34], which is used as a standard benchmark for subjectivity detection.

Problem Statement

Subjectivity detection involves identifying whether a piece of text is subjective or objective. Subjective expressions are phrases that express emotions, feelings, opinions, stances, and speculations [33]. Objective expressions are simply expressions that are not subjective. Usually official language describing facts is rather objective than subjective.

Subjectivity detection and opinion extraction are quite similar problems. A core difference in this work, as mentioned in Section 3.2, is that subjectivity detection is assumed to be sentence-based, whereas opinion extraction is word-based.

3.5 Sentiment Intensity Ranking

Sentiment intensity ranking aims to determine the degree or intensity of sentiment expressed in text. Early approaches by [33] focused on recognizing contextual polarity and distinguishing between weakly and strongly subjective expressions. The SemEval-2007 Task 14 [92] introduced a shared task focused on detecting the presence and intensity of emotions in text. Later works, such as [35, 85, 93], contributed to the development of tools and resources for sentiment analysis.

Problem Statement

Sentiment intensity ranking involves assigning a score or ranking to text based on the intensity of sentiment expressed. This can be considered as the regression version of sentiment analysis, where a score is given within the range $[-1, 1]$, where -1 denotes very negative, 0 denotes neutral, and 1 denotes very positive. A positive value, for example 0.5, can be moderately positive, whereas a negative value, for example -0.5, can be moderately negative.

3.6 Emotions Intensity Ranking

Emotions intensity ranking focuses on determining the intensity of specific emotions expressed in text. Early works, such as [94], explored corpus-based approaches to finding happiness and other emotions. The introduction of the NRC Emotion Lexicon by [95] provided a valuable resource for emotion detection and intensity ranking. The WASSA-2017 Shared Task on Emotion Intensity [36] further advanced the field by providing benchmark datasets and evaluation metrics.

Problem Statement

Emotions intensity ranking involves determining the intensity of specific emotions (e.g., joy, sadness, anger, fear) expressed in text. This task includes:

1. **Emotion Detection:** Identifying the presence of specific emotions in text.

2. **Intensity Scoring:** Assigning a numerical score within the range $[0, 1]$ representing the intensity of each detected emotion. 0 represents that an emotion is not detected, while 1 represents a very strong presence of the emotion.

In this work, only the intensity scoring problem as a regression problem is investigated, but not the emotion detection problem as a classification problem.

3.7 Toxicity Detection

Toxicity detection focuses on identifying harmful or abusive language in text. Early work in the late 1990s introduced initial methods for automatic recognition of toxic and hostile language [37]. Later research in the 2010s by [38] on detecting hate speech on the web and [96] on detecting racist tweets expanded the scope of toxicity detection. The development of large-scale datasets and benchmark tasks, such as [39, 97], significantly contributed to the effectiveness of techniques for toxicity detection.

Problem Statement

Toxicity detection involves identifying and classifying harmful or abusive language in text. Typically, this is done on a sentence basis as a binary classification problem. According to [97], this can be defined by six labels:

- **Toxic**: Comments classified as toxic are characterized by their harmful, offensive, or disrespectful nature. They often include hostile language aimed at belittling or harming individuals or groups. This type of toxicity can manifest as aggressive language intended to provoke or distress others, with content that may be vulgar or generally unacceptable in civil discourse. Toxic comments are not always explicit but can be insidious, perpetuating a negative and hostile environment.
- **Severe Toxic**: Severe toxic comments represent an escalated level of toxicity, where the language used is extremely harmful and significantly more abusive than general toxic comments. These may include explicit threats of violence, harassment, or severe aggression, often targeting individuals with highly inflammatory language. Such comments are crafted to cause significant emotional distress or fear and are often part of targeted attacks or sustained harassment campaigns.
- **Obscene**: Obscene comments contain language or content that is vulgar, profane, or lewd. This category is marked by the use of swear words, sexually explicit language, or other

forms of crude content that are offensive due to their graphic nature. Obscene comments typically violate community guidelines due to their explicitness and are considered inappropriate for general audiences, contributing to an unwelcoming environment.

- **Threat**: Comments labelled as threats include language that implies or directly states intentions of harm or violence towards an individual or group. Such language includes explicit death threats, threats of physical harm, or any form of intimidation that could incite fear or distress in the targeted individual. Threatening comments are serious violations that can lead to legal consequences and are often treated with heightened concern by moderation teams.

- **Insult**: Insulting comments are aimed at demeaning or belittling individuals through derogatory or mocking language. They often involve personal attacks, name-calling, or negative statements about the characteristics, abilities, or actions of a person. These comments contribute to a hostile atmosphere and can affect the well-being of the targeted individual by undermining their self-esteem or reputation.

- **Identity Hate**: Identity hate comments are those that attack or demean individuals based on intrinsic aspects of their identity, such as race, religion, gender, sexual orientation, or ethnicity. These comments utilise slurs, stereotypes, or prejudiced language to promote discrimination or hatred against specific groups. Identity hate is particularly damaging as it seeks to marginalise or ostracise individuals based on fundamental aspects of their identity, fostering a culture of intolerance and bigotry.

3.8 Suicide-Tendency Detection

Suicide-tendency detection focuses on identifying signs of suicidal ideation in textual data. Early work in the early 2010s by [40] introduced NLP techniques for analysing sentiment in suicide notes, followed up by a shared task on sentiment analysis of suicide notes [41], providing initial insights into detecting suicidal tendencies. Subsequent research by [98] on predicting depression via social media and [42] on quantifying mental health signals in Twitter expanded the scope of suicide-tendency detection to social media data. [99] demonstrates an advanced ML approach of assessing risks within suicide-related communication.

Problem Statement

Suicide-tendency detection involves identifying signs of suicidal ideation or behaviour and depression signs in text. This is typically a binary classification problem.

3.9 Well-being and Stress Assessment

Well-being and stress assessment focuses on identifying signs of well-being or stress in textual data. Early work by [43] explored linguistic styles and their relationship to psychological states, providing foundational insights into well-being and stress detection.

Subsequent research by [100] on linguistic markers of psychological change and [98] on predicting depression via social media expanded the scope of well-being and stress assessment to social media data. Enhancing this with DL techniques was introduced by [44].

Problem Statement

Well-being and stress assessment involves identifying signs of well-being or stress in text. This is typically a binary classification task. Stress identification is a problem that can be represented as a less severe version of suicide-tendency detection, as explained Section 3.8.

3.10 Sarcasm Detection

Sarcasm detection focuses on identifying sarcastic expressions in text. Early work in the mid-2000s, by [45], on sarcasm recognition for spoken dialogue systems provided initial methods for detecting sarcasm.

Subsequent research by [101] on semi-supervised recognition of sarcastic sentences and [46] on identifying sarcasm in Twitter expanded the scope of sarcasm detection. The application of DL techniques, as demonstrated by [47] and [102], further enhanced the field.

Problem Statement

Sarcasm detection involves a binary classification of whether a given sentence is sarcastic or not.

3.11 Big-Five Personality Assessment

The twentieth century has witnessed the very extensive development of the Big-Five personality model in the field of psychometrics [103]. The model emerged as a result of factor analysis on human answers related to personality description. The model identifies five independent dimensions that explain the majority of variance in descriptions of human behaviour. The five personality

traits are Openness to Experience, Conscientiousness, Extraversion, Agreeableness, and Neuroticism (OCEAN).

Early work in the late 1990s explored linguistic styles and their relationship to personality traits, providing foundational insights into personality assessment [43]. Subsequent research by [104] on using linguistic cues for personality recognition and [105] on classifying author personality from weblog text expanded the scope of personality assessment using NLP. The application of ML and DL techniques, as demonstrated by [48] and [106], improved the field further. [49] is a survey of many of such advances. [82, 107] presented challenges for predicting personality traits from multimodal signals, and enhanced ML and DL models as a result.

Problem Statement

Big-Five personality assessment involves identifying the Big Five personality traits from text. This task typically includes:

1. **Trait Detection:** Identifying linguistic and contextual signals indicative of each of the Big Five personality traits.
2. **Trait Scoring:** Assigning a numerical score representing the level of each personality trait.

In the multiple studies presented in this work, [1,4] explores the *trait detection* version, whereas [2] uses a variant similar to *trait scoring*.

3.12 Engagement Measurement

Engagement measurement focuses on predicting the level of engagement (likes, retweets, comments) that a piece of content will receive. Early work in the late 2000s by [108] on tracing information flow provided foundational insights into the dynamics of information spread. Further research in the 2010s has progressed the field of engagement measurement. [50] worked on information diffusion across topics and [51] expanded the scope of engagement measurement by predicting the spread of ideas in microblogging communities. [109] used classical ML techniques for engagement measurement.

Problem Statement

Engagement measurement involves predicting the level of engagement (e. g. , likes, shares, comments) that a piece of content will receive. This task typically includes:

1. **Engagement Prediction:** Predicting the number of engagements a piece of content will receive. This can be the number of reply comments, number of likes, or number of shares.

2. **Virality Assessment:** Assessing the potential of content to go viral based on predicted engagements. This can be the number of views, or the number of shares.

In this work, the number of retweets (which is the number of shares on the Twitter platform) is used as the primary engagement measure, which represents both aspects of engagement.

Chapter 4

Datasets

For the affective computing tasks, 13 corresponding datasets are employed in the studies [1–4].

4.1 Sentiment Analysis

The Twitter140 dataset [93] is designed to address challenges in sentiment analysis on social media platforms, focusing on Twitter. The dataset comprises tweets that have been automatically annotated with sentiment labels using a technique known as distant supervision [110]. This method leverages the presence of emoticons in tweets as noisy labels for sentiment, under the assumption that certain emoticons strongly correlate with specific sentiments.

Distant supervision for labelling allows for the generation of large-scale sentiment datasets without the need for costly manual annotation [93]. Tweets containing positive emoticons, such as smiley faces, are labelled as positive, while those with negative emoticons, such as frowny faces, are labelled as negative. This automatic annotation process enables the rapid assembly of vast amounts of data, although it introduces some noise due to the inherent ambiguity and variability in how individuals use emoticons.

The primary motivation behind creating this dataset is to develop a robust model capable of accurately classifying the sentiment of tweets in real-time, which is a valuable asset for businesses, politicians, and public figures who wish to gauge public opinion and sentiment trends quickly and accurately. The use of Twitter data is particularly relevant given the role of the platform as a primary medium for public discourse and opinion sharing.

The dataset is instrumental for training machine learning models to perform sentiment analysis, particularly those that require large amounts of labelled data to learn from context and linguistic nuances effectively. By providing a dataset annotated through distant supervision, the researchers

offer a resource that balances scale with the practicalities of dataset creation, making it a valuable tool for both academic research and real-world applications in sentiment analysis.

4.2 Opinion Extraction and Aspect-based Sentiment Analysis

ABSA involves the identification of opinions about specific entities and their aspects within a text, aiming to provide fine-grained sentiment analysis that captures detailed insights beyond overall sentiment. The datasets utilised for this task in SemEval-2014 and SemEval-2015 have become benchmarks for evaluating ABSA systems, focusing on domains such as laptops (lap14) and restaurants (res14, res15) [27, 111]. The splits from [112] are utilised.

The SemEval-2014 Task 4 dataset includes reviews from the laptop and restaurant domains, annotated for aspect terms and their associated polarities. This dataset is designed to facilitate four subtasks: aspect term extraction, aspect term polarity classification, aspect category detection, and aspect category polarity classification. Aspect terms, which refer to specific components or attributes of the entities, are labelled with their sentiment polarity, including positive, negative, neutral, and conflict sentiments. For example, a sentence like "The battery life is excellent, but the screen is dim" would involve identifying 'battery life' and 'screen' as aspect terms with corresponding positive and negative polarities, respectively [27].

Similarly, the SemEval-2015 Task 12 extended this work to include an additional domain, hotels, further solidifying the framework for ABSA tasks. This iteration introduced an aspect category extraction task, where predefined aspect categories, such as FOOD, SERVICE, and PRICE in restaurant reviews, were used to classify sentiment expressions even if the aspect term was not explicitly mentioned in the text. For example, "The restaurant was expensive, but the menu was great" illustrates how aspect categories can be inferred from adjectives without direct aspect term mention [111].

The datasets provided for these tasks consist of manually annotated reviews, offering training and evaluation splits with clear annotations for the sentiment polarity and category of each aspect [27, 111]. The annotation process involves multiple layers, starting from identifying explicit aspect terms and their sentiment polarities to categorising the broader sentiment implications based on aspect categories. This comprehensive annotation enables the development of sophisticated models that can effectively tackle both explicit and implicit sentiment expressions across various domains, offering significant insights into consumer opinions and preferences.

In the context of opinion extraction, the SemEval datasets have been instrumental in advancing methods that address the complexity of sentiment analysis by considering both the specific entities and their attributes discussed within user-generated reviews. These datasets continue to serve

as a benchmark for developing more robust and accurate ABSA systems, providing a crucial foundation for research and development in sentiment analysis [27, 111].

4.3 Subjectivity Detection

In [34], the dataset plays a crucial role in enhancing sentiment analysis, explicitly focusing on distinguishing subjective content within texts. This dataset includes two primary components to facilitate precise sentiment classification by segregating subjective from objective text elements.

The "polarity dataset" utilised in [34] comprises 2,000 movie reviews, split evenly between positive and negative sentiments, with all reviews written before 2002 to avoid authorial overlap. Each review was selected to maintain a balanced representation, capped at 20 reviews per author across 312 contributors. This corpus is part of a well-established database often used in sentiment analysis research, providing a solid foundation for testing sentiment analysis methodologies.

This dataset was constructed by extracting sentences identified as subjective from review snippets on Rotten Tomatoes– a site known for its rich, opinion-laden content– and objective sentences from plot summaries listed on the Internet Movie Database (IMDB), which typically present factual descriptions without personal sentiment. This approach ensures a clear distinction between subjective and objective textual data, which is crucial for effectively training models that focus selectively on sentiment-laden content.

By strategically compiling and utilising this dataset, [34] not only addresses challenges inherent in traditional sentiment analysis but also provides a refined toolset for isolating and evaluating sentiment within texts, offering significant contributions to the field and valuable resources for ongoing research endeavours.

4.4 Sentiment Intensity Ranking

The sentiment intensity dataset is from SemEval-2017 Task 5 [113], which provides a platform for the nuanced analysis of sentiments expressed in the financial domain, specifically targeting microblogs and news articles. This dataset was developed to address the unique challenges of sentiment analysis in financial texts, where the sentiment linked to financial entities such as stocks, commodities, or markets can have predictive implications for financial movements and investor behaviours.

Each entry in the dataset is annotated not just for the presence of sentiment but also for the specific intensity of sentiment on a continuous scale from -1 to 1. This scale ranges from negative to

positive, allowing for a precise measurement of sentiment that captures subtle nuances beyond simple positive, neutral, or negative classifications. This detailed approach helps understand the exact sentiment impact of financial news and social media on market movements and investment decisions.

The motivation for creating this dataset stems from the need in the financial sector for advanced tools to monitor and analyse the vast amounts of unstructured text data generated daily. Traditional sentiment analysis tools, which often work with coarse sentiment categories, are inadequate for the financial context where the strength and direction of sentiment can influence trading decisions and market predictions. As financial markets are susceptible to public perceptions and news, accurately quantifying sentiment from financial texts is crucial for traders, analysts, and automated trading systems.

The dataset was meticulously compiled from sources that financial professionals and casual investors frequently access, such as financial news websites and social media platforms where users discuss stocks or markets. The inclusion of microblogs reflects the growing influence of platforms like Twitter in the financial world, where user-generated content can significantly sway market sentiments. This dataset is not only a valuable resource for developing more sophisticated sentiment analysis models but also serves as a benchmark for comparing the efficacy of different sentiment analysis techniques in the context of financial markets. This enhances the predictive analytics used in financial planning and algorithmic trading.

4.5 Emotions Intensity Ranking

The emotions intensity dataset is from WASSA-2017 Shared Task on Emotion Intensity (EmoInt) [36]. The dataset is a specially curated collection of tweets annotated to assess the intensity of emotions conveyed in textual content. This dataset differs from traditional emotion detection tasks that typically categorise emotions in binary terms or as discrete classes. Instead, it provides a nuanced approach by quantifying the degree of emotional expression from 0 (no emotion) to 1 (highest intensity of emotion), focusing on four primary emotions: anger, fear, sadness, and joy. Each tweet is labelled with a real-valued intensity score, reflecting the level of the specified emotion, determined by averaging scores from multiple annotators to ensure the reliability of the annotations.

The development of this dataset is motivated by the need to capture the subtle variations in emotional intensity that are often overlooked in categorical models. Recognising these variations is crucial for applications requiring fine-grained analysis of emotional states, such as monitoring mental health from social media texts, or enhancing customer service systems by better understanding client sentiments.

The collection process involved selecting tweets that naturally express emotions across various topics and contexts, aiming to reflect the genuine emotional expressions of users rather than relying on artificially generated examples. This method ensures that the dataset is representative of real-world scenarios, enhancing the applicability of models trained on this data for practical purposes.

By providing a rich source for training computational models to detect and quantify emotional intensities, this dataset not only supports advancements in emotion analysis technology but also contributes to a deeper understanding of how emotions are articulated and perceived in online communication. This is particularly valuable for improving interactive systems and for gaining insights into the emotional dynamics expressed through social media.

4.6 Toxicity Detection

The dataset for toxicity detection was released as the dataset for the Kaggle challenge "Toxic Comment Classification Challenge" [97]. The dataset consists of 159,000 human-annotated comments collected from discussions on Wikipedia. The comments are annotated for six different binary labels: toxicity, severe toxicity, obscenity, threats, insults, and identity-based hate. The negative classes are extremely over-represented in comparison to the positive classes. [114] reviews different publications approaching the dataset after the conclusion of the competition. As an example, [115] demonstrate an approach on the dataset using various RNNs.

4.7 Suicide Tendency Detection

The dataset for Suicide and Depression Detection originated from [116]. It serves a critical role in the emerging field of mental health informatics, particularly focusing on the detection of suicidal ideation and depression through social media content analysis. This dataset is compiled from various online forums and social media platforms where individuals often express their emotional states and discuss personal challenges, including mental health issues.

The dataset comprises posts from the Reddit platform, from the subreddits 'SuicideWatch', 'depression', and 'teenagers'. The ground truth labels for the two subreddits 'SuicideWatch' and 'depression' are assumed to be positive for showing depression or suicidal ideations and negative for the 'teenagers' subreddit. [116] used for the test-set tweets from Twitter; however, in this work, the posts from Reddit are resplit for the train/dev/test portions.

The primary motivation behind the creation of this dataset is to develop and refine computational tools that can automatically identify signals of depression and suicidal tendencies in social media texts. By enabling early detection of such mental health issues, these tools can potentially guide

interventions and support efforts, thereby contributing to preventive mental health care.

The importance of this dataset extends beyond academic research; it can be a vital resource for public health officials and mental health professionals, who increasingly rely on digital platforms for early diagnosis and therapeutic interaction. This dataset can be utilised to address the growing need for effective monitoring and support mechanisms in mental health services, which are often overwhelmed and under-resourced. By providing a means to automatically detect early signs of severe mental health conditions, this dataset supports ongoing efforts to integrate more proactive and preventive approaches into mental health care systems worldwide.

4.8 Well-being Assessment

The dataset explored in [117] is specifically tailored to detect stress levels from text data derived from social media platforms. This dataset comprises a wide array of social media posts, which are meticulously collected to analyse and understand the manifestation of stress in digital communications. Experts in psychology have manually annotated each post to indicate varying stress levels, from no stress to severe stress, providing a nuanced view of emotional states.

The labels for this dataset are created through a detailed manual annotation process. Expert annotators, trained in psychological assessment, read through each social media post and assign a stress level based on predetermined criteria, including linguistic indicators of stress, such as language intensity, urgency, and stress-related terms. This process ensures that the dataset accurately reflects the complexities of human emotional expression in written form. The labels are binary labels for stress-positive or stress-negative. Furthermore, the dataset consists of several benchmarks, originating from Twitter or Reddit, also with different variants, including short texts, e. g. , short tweets and article titles, and long texts like full Reddit articles.

The motivation behind the creation of this dataset is twofold. Firstly, it addresses the growing need for tools that can effectively identify and quantify stress levels in social media users, allowing for timely interventions. Secondly, this dataset aims to advance research in mental health informatics by providing a resource that can be used to train computational models to recognise signs of stress automatically. These models can then be integrated into applications that monitor social media platforms, providing users with feedback and support to manage their stress levels or alerting mental health professionals to situations that may require intervention.

This dataset is particularly valuable in the context of increasing global stress and mental health issues, where social media often serves as an outlet for individuals to express their feelings and seek comfort or advice. By enabling the development of automated tools to detect stress, this dataset contributes to the broader efforts of using technology to support mental health well-being.

4.9 Sarcasm Detection

The dataset for sarcasm detection in [118] has been meticulously curated to address the inherent limitations present in previous datasets, which often suffer from noisy labels and inadequate instance volumes for robust ML applications. The dataset uniquely captures the dichotomy between sarcastic and non-sarcastic texts by integrating news headlines from two distinct sources, TheOnion and HuffPost. The former, a platform renowned for its satirical take on current events, provides the sarcastic component. At the same time, the latter offers a corpus of genuine news headlines to represent non-sarcastic texts.

The compilation process involved meticulous sourcing from TheOnion's "News in Brief" and "News in Photos" sections to ensure a rich representation of sarcasm. In contrast, non-sarcastic headlines were extracted from the news archive of HuffPost, ensuring the dataset mirrored real-world events accurately. This approach not only enhances the utility of the dataset in training NLP models but also ensures high linguistic quality by leveraging professional editorial standards that reduce informal language typical of social media texts.

Statistically, the dataset includes a substantial number of records, facilitating comprehensive model training and evaluation. This scale advantage, coupled with high-quality, controlled labels, markedly exceeds the capabilities of manually annotated datasets. The dataset has since become a cornerstone in sarcasm detection research, supporting numerous studies aimed at refining computational models to accurately detect sarcasm using both traditional linguistic features and advanced deep learning techniques.

Moreover, the associated research elaborates on a novel Hybrid NN architecture tailored to exploit this dataset optimally. Incorporating sophisticated elements like attention mechanisms, this model architecture enhances the understanding and interpretation of sarcasm in textual data. Overall, the "News Headlines Dataset" not only fulfils a critical need within the NLP community for high-quality, large-scale data but also catalyses further innovation and exploration in the sarcasm detection domain.

4.10 Big-Five Personality Assessment

The "First Impressions" dataset was developed as a part of competition [107, 119] for personality predictions. The dataset consists of 10,000 short video clips, each 15 seconds long, extracted from YouTube. These clips feature individuals from diverse backgrounds speaking directly to the camera in a self-presentation style reminiscent of job interviews or personal introductions. This setting provides a rich basis for analysing apparent personality traits expressed through verbal and

nonverbal communication.

The dataset is annotated based on the Big-Five personality traits, which includes the traits: Extraversion, Agreeableness, Conscientiousness, Neuroticism, and Openness to Experience. The labels are expressed as five independent values within $[0, 1]$. These traits are universally recognised for their relevance in predicting a variety of important personal and professional outcomes and are considered stable across different contexts and over time [120]. To ensure the reliability and objectivity of the personality assessments, the dataset employed Amazon Mechanical Turk (AMT) workers to perform pairwise comparisons of the videos. The pairwise sampling was conducted based on the small-world algorithm, and a study for optimising the number of pairs [121]. This method reduces individual rater bias and utilises a Bradley-Terry model [122] for converting pairwise comparisons into continuous trait scores, enhancing the precision of the personality measurements.

A significant aspect of the dataset is extracting textual data from the video content [119]. The spoken content of the videos is transcribed, providing textual data that allows for the application of NLP techniques. This text modality is essential for analysing how linguistic cues in self-presentation correlate with perceived personality traits, offering insights into the intersection of language use and personality perception. This makes the dataset particularly valuable for developing and benchmarking algorithms that assess personality from brief video clips, which is increasingly important in fields like automated interviewing, human-computer interaction, and social media analysis.

4.11 Engagement Measurement

The "Social Media Interactions on TED Talks" dataset on Kaggle[1] is a comprehensive resource that provides insights into how TED Talks resonate with audiences on Twitter. Each record in this dataset corresponds to a unique TED Talk, enriched with metadata such as the title, description, and relevant tags. This information helps categorise and contextualise the data, linking engagement metrics to the specific themes discussed in each talk.

The primary focus of the dataset is on capturing the engagement metrics of tweets that mention or share TED Talks. The metrics include retweets, likes, and replies, allowing researchers to identify patterns of interaction and how individual talks resonate across the Twitter community. These metrics are crucial for understanding the nature of engagement on social media, enabling the exploration of which types of talks attract the most attention and how audience behaviour varies based on the topic. In this work, $\log_{10}(\text{number of retweets} + 1)$ is adopted as the target

[1] https://www.kaggle.com/datasets/thedevastator/social-media-interactions-on-tedtalks-dataset

engagement label since it represents the label distribution effectively.

Another significant aspect is the temporal information provided through the timestamps associated with each tweet. This temporal data allows researchers to track the propagation of TED Talk tweets over time, revealing trends like the immediate burst of engagement following a new release versus the long-tail engagement for older talks. Analysing these patterns can help uncover factors that lead to viral content and sustained social media activity.

The dataset also provides anonymised user data, such as follower counts and user profiles. Though limited in detail for privacy reasons, this data gives a glimpse into the demographics and influence of users who interact with TED Talks. Researchers can assess how user profiles and follower counts impact the reach and engagement of shared content.

Part III

Main Contributions and Studies

Chapter 5

Prompting Paradigm

Foundation models constitute a mode of operation based on prompting, typically based on the Transformer architecture [16] and LLMs. In a way, foundation models are special forms of LLMs that are used on the text describing a task and its corresponding given input; then, the model would generate text describing the answer of the task on the given input. Before foundation models, ML and DL models would be architected in a way that generates specific outputs that are adjusted for a given problem, whereas foundation models are more generic in this manner due to the common universal medium of processing information, namely natural language.

In general terms, foundation models employ generative Transformer architectures to process text, images, or audio; this is due to the fact that Transformer architectures can be conditioned by using cross-attention on any input, be it text, images, or audio. The focus of this work is mainly on foundation models processing text; however, it is important to highlight that there are various research and implementations for foundation models utilising images (e. g. , GPT-4 Omni, LLaVA [123] and LLaVA-1.5 [124]), or audio (e. g. , UniAudio [125]). [126] survey a wide variety of multimodal LLMs.

5.1 Chat Models

Chat models are foundation models that process text consisting of a list of messages, where a message consists of a role and text content. Typically, messages display alternating roles emulating an imaginary conversation or a chat. The list of messages can be converted to a string that can be completed auto-regressively by the chat model to produce the next message.

Depending on the roles assigned, such conversations can represent an interaction between a user and an assistant agent. The assistant can request that the user perform actions, and the user can

then execute these actions and report the results within the conversation. This formulation allows for agent-based interactions.

A message can be textually represented in various forms, such as including beginning and end tokens for each role. Consequently, a list of messages can be represented as a text constructed by concatenating the representations of messages. A chat model can predict a message from a conversation on a token-by-token basis until a special token denotes the end of the message. Although a chat model can continue to generate more messages, in practice it is only used to generate one message at a time; this is then handled externally in an application, e. g. , a chat application, to receive messages externally and then generate the corresponding responses, and so on. Examples of chat conversations are shown in Figure 5.1.

The roles in messages can be unique for every model, however, there are some typical roles:

- System: A message containing general instructions for the assistant, specifying how the conversation will flow and specifying constraints about it.
- User: A message given by a user, typically a human user; however, this can also be another AI agent.
- Assistant: The response message of an AI agent. This constitutes the output predicted by the model for the last user message before it.
- Function Call: A special type of assistant message that contains a text description of an action to be performed externally.
- Function Result: A special type of message that contains a description of the results of executing an external action.

5.2 Methods for Prompting Paradigm

The chat models (introduced in Section 5.1) present the possibility of a new paradigm of predictive learning, based on the zero-shot abilities of foundation models and chat models. Chat models can be used as LLMs to complete text within a chat context, hence providing an assistant behaviour that is able to solve problems and finish tasks. This introduces the new paradigm of predictions via prompting. Instead of training a model that is engineered to predict specific outputs, e. g. , predicting ‘positive’ or ‘negative’ for a sentiment analysis model, one foundation (chat) model can be used to make the same prediction but described in natural language.

Foundation models need to be *programmed* to predict labels for a specific problem, which is done by a text template given to the foundation model. This is defined as the *prompt* corresponding to

<SYSTEM>
You are an assistant that helps the user with maths questions. You have access to a tool called 'run_python_code' that can run python code for you, to help you run computations.
</SYSTEM>
<USER>
What is the 50th Fibonacci number?
</USER>
<CALL>run_python_code
```
a, b = 0, 1
for _ in range(49):  a, b = b, a + b
print(a)
```
</CALL>
<RESULT>
7778742049
</RESULT>
<ASSISTANT>
7,778,742,049
</ASSISTANT>

(a) Example of a chat conversation using external actions, like running a Python code, to respond to a question by the user.

<SYSTEM>
You are an expert at sentiment analysis. Your task is to respond to the text given by the user by predicting its sentiment. Output only one word only, either 'positive' or 'negative'.
</SYSTEM>
<USER>
I received some good news today, I feel very happy.
</USER>
<ASSISTANT>
positive
</ASSISTANT>

(b) Example of a chat conversation that predicts the sentiment of a text.

Figure 5.1: Two examples of how chat conversations can be used to solve different problems, and how the different types of messages are incorporated within that. A foundation model (as an LLM) would be used to complete the text after the token </USER>, until it finishes the maximal context size, or until the token </ASSISTANT> or </CALL> is generated. Additionally, if a model generates <CALL> message, then the action within is parsed and executed in an external environment, and the <RESULT> is given, to generate the proceeding <ASSISTANT> response. This is just a prototype representation for illustration and is not necessarily the exact representation format used by specific LLMs.

a given task. The prompt in that case would be equivalent to functions in traditional programming languages, except that it does not define *how* the task is solved and it is relatively more fuzzy but

can do more complex tasks. Furthermore, prompting offers more flexibility to solve new problems without being necessarily trained for, but its performance is generally unclear. One of the aims of this work is to investigate these capabilities. A general outline of the prompting paradigm is depicted in Figure 5.2.

Prompting is the gateway to access the predictive abilities of foundation models, since the prompt used is the main factor determining the output of a given foundation model. This comes with its limitations, which are introduced through the following two analogies. Humans utilise various techniques to approach new problems, especially for challenging problems that are hard to solve the first time. One of the problem solving techniques is "getting the hands dirty" [127], which is employed to get familiar with a hard problem by attempting to explore its properties and the basic ideas that might solve it. As a result, some patterns emerge that can solve the problem or at least open new avenues to reach the solution. The second example is when a person writes an essay, they typically do not write it entirely in one go from beginning to end. What happens instead is an iterative writing process, by first writing a draft and then reiterating it by rewriting a better version or by fixing the errors of the already written draft until the final form of the essay is achieved. Both of the aforementioned examples elaborate on the process humans use to bridge the gap between their internal knowledge and reaching a final concrete solution. This is analogous to the prompting in LLMs, where an LLM has an internal world-view and can start to write a solution or write an outline for a solution, then detail the corresponding steps thereafter.

The pipeline in Figure 5.2 can be represented mathematically as follows. The prompting paradigm can be defined as a higher order function F, given a foundation model $\mathcal{M}$, generation parameters Θ, a problem to solve Q and a parsing function p, then $F_{\mathcal{M},\Theta,Q,p} : \mathcal{T} \rightarrow \mathcal{L}$ is a predictor function that predicts labels from a set $\mathcal{L}$ for inputs from a set $\mathcal{T}$. The predictor function in this case makes a prediction on an input t by evaluating:

$$F_{\mathcal{M},\Theta,Q,p}(t) = p(\mathcal{M}_{\Theta}(P(Q, <t>))). \tag{5.1}$$

Equation (5.1) mainly performs the pipeline in Figure 5.2, by executing:

1. Convert an input t to a textual representation $<t>$.
2. Substitute $<t>$ and values based on the problem Q into the prompt template P.
3. Generate text auto-regressively with the foundation model $\mathcal{M}$ and auto-regressive generation algorithm with parameters Θ.
4. Convert the text answer obtained to one of the predefined labels by way of parsing, using the parsing function p.

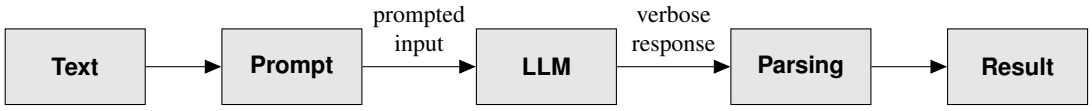


Figure 5.2: General processing outline of the prompting paradigm.

5.2.1 Prompting

Prompting is the main gateway to unlock the power of LLMs, as it is the primary input that conditions the output of a particular model. The remaining factors that impact the output of a given model are the randomisation effects during generation, as discussed in Section 2.5.2. The unfortunate fact about prompting in LLMs is the model dependence, since each model has slightly different architecture and weights, leading to different predictions for a given input. However, some emerging patterns can be used as general guidelines in prompting. Chain-of-Thoughts (CoT) is a common pattern used in prompting, which instructs the LLM to verbalise thoughts before answering a given task, hence giving it a buffer to 'process' some thoughts. Follow-up verification prompts can then be used to refine and enhance initial responses. Additionally, incorporating subject-matter expertise within prompts or using different incentives are hypothesised to align model responses with domain-specific knowledge, potentially enhancing performance. Furthermore, the existence of *magic sentences*, e. g. , "take a deep breath", is assumed to elicit superior results.

A wide range of prompting techniques are investigated in Chapter 8, by systematically evaluating various prompting strategies and the sensitivity due to token sampling during text generation. Chapter 8 aims to contribute to a deeper understanding of effective prompt engineering.

5.2.2 Utilising GPT Models as Foundation Models

Utilising GPT models as a foundation model is achieved through the API provided by OpenAI[1]. There are two main models of ChatGPT, namely GPT-3.5 and GPT-4. The exact specifications of both models are not published, due to being closed models. However, the general statements about GPT-3.5 indicate that it is an extension of GPT-3 [18], which is trained to also generate code [128], and is aligned to respond to user instructions similar to InstructGPT [21]. On the other hand, GPT-4 is similar to GPT-3.5, with a crucial difference of being massively larger. Additionally, GPT-4 has additional capabilities like multimodal capabilities [129], e. g. , using images as an additional input. There are several versions of the available versions for both models [2]. The experiments in [2, 4] used 'gpt-3.5-turbo-0301' for GPT-3.5, and 'gpt-4-0314' for GPT-4. [3] used 'gpt-3.5-

[1]https://platform.openai.com/docs/guides/gpt/chat-completions-api
[2]https://platform.openai.com/docs/models

turbo-0613' for GPT-3.5. A prediction, for one example, is made by adjusting the prompt for the specific problem at hand and its target label. After that, two messages are sent to the model, the first is the system prompt including the problem description and all the instructions, and the second is a user message including the text of the input example. The response is an assistant message, including the prediction of the corresponding input example, which is post-processed to extract the label.

5.2.3 Parsing

LLMs perform predictions by simply generating text containing or describing the answer. Such text can be a simple "yes" or "no", or can also be "This sentence expresses positive sentiment because it is describing news of winning an award.", "7,921,321,421", or "As an AI language model, I cannot give an answer to this question with such little information". As a result, the answers of the LLMs need to be transformed from the text format into a specific cleaned answer. This defines the parsing process, where an answer described in natural language is extracted from a text and mapped to some label. LLMs can be prompted to adhere to output format instructions; however, such instructions are not always strictly followed by the LLMs due to the fuzziness of NNs. Furthermore, such instructions can come at the expense of performance at the given task.

No parsing

Problems that have a simple binary response can be prompted to directly output this label exclusively, hence there is no need for additional processing. This includes labels like "yes/no", "positive/negative", "high/low", or "A/B". However, there are prompts that can still make such problems not straightforward to parse; this will be discussed below.

Simple search

A technique for parsing labels is simply searching for the words of the possible labels within the LLM response. For example, finding if the word 'positive' occurs within the text, hence predicting the 'positive' label and similarly for 'negative' or any other set of labels.

This has a drawback in case the model responds with text like "This text does not seem to be very positive, label: negative", where both words exist within the text and require some contextual understanding to extract the predicted label. [4] shows that this method results in at least 95 % of the examples including exactly one of the two binary labels.

Last line parsing

A technique to parse verbose responses is to prompt the model to write an answer with all its reasoning as it sees fit, with a constraint to give the final label at the end of the answer in a new separate line. After that, the content of the last line is parsed according to the other methods.

Setting parsing rules

Although LLMs do not strictly follow the output format rules all the time, they still follow such rules most of the time or deviate just slightly from such formats. Consequently, the outputs in most cases can be parsed by a simple rule set, namely:

1. Remove whitespace characters at the beginning or end.
2. Transform to lowercase.
3. Remove prefixes like 'target', 'label', 'prediction', and 'output', also remove the colons after these if they exist.
4. Remove the quotation marks.
5. Remove the periods at the end.

These steps are applied in the mentioned order, and only the applicable steps are used.

Parsing rules for multi-label responses

The method above for parsing based on rules works mainly with single labels; however, it needs more sophistication to handle multi-label predictions. The two use cases for this are word labelling, i. e. , predicting a label for each word, and predicting a fixed number of multiple single labels, e. g. , predicting the Big-Five personality traits. This can be solved by first prompting the model to output a list of predictions, represented as a bullet-point list. This is done in markdown language, where each item of a bullet-point list is represented in a separate line starting with a bullet-point marker like '-' or '*'. Consequently, this specifies the format for each item in the list as "{item} is {item-label}", which results in straightforward parsing.

Using NLP

A technique for processing verbose responses uses traditional NLP methods instead of parsing, by using BoW, Word2Vec, or NNs (e. g. , RoBERTa) to process the output of LLMs. The intuition

behind this is that LLMs will give responses containing the label within the text description, and hence, it can be processed by NLP methods. This technique can benefit from verbose responses, as the explanations given by LLMs for their answers can assist in reaching a well-informed decision by the post-processing NLP model. Two examples are: "This text has a positive sentiment because the person is speaking about receiving good news and explicitly stating happy emotions.", which would receive a very high positive sentiment. The second example, "The text suggests the presence of positive sentiment since the person mentioned finding another job interview opportunity. However, this is just an estimate since the person did not explicitly express positive emotions.", would receive a slightly positive sentiment.

5.2.4 Ranking and Regression Modelling

Engaging ChatGPT models for classification tasks requires directing them to select an appropriate label from a set predetermined for such tasks. The challenge intensifies when transitioning to regression tasks, where eliciting an exact numerical value from ChatGPT models becomes intricate due to the variability in scales and the subjective criteria used in dataset annotation. Consequently, the strategy for evaluating ChatGPT in regression or ranking problems involves transforming these issues into a series of pairwise ranking classification challenges by posing a binary classification question *is* $y_a > y_b$*?*.

Moreover, this modelling strategy applies not only to evaluations but can also be extended to predictions. A predictive approach can start with a range of values $[a, b]$, then iteratively refine the range of possible outcomes by making comparisons to an example with a score close to the median value of the remaining range, mirroring the binary search algorithm, hence reducing the size of the range by a factor of 2. This refinement continues until the range is sufficiently narrow.

5.2.5 Fusion

The concept of fusion or ensembling has a long history in ML [52]. The main motivation behind ensembling is to diversify the use of models to reach better and more stable results; in other words, merge a set of uncorrelated or weakly correlated models. This motivates employing a fusion of traditional NLP models like RoBERTa with foundation models.

The methods behind the prompting paradigm described earlier utilise a prompt template and a foundation model to reach a response. The response is a textual representation of a classification of the input text; such responses need some parsing or post-processing to overcome the lack of precision of LLMs to give a response that contains only the predicted label. Furthermore, various LLMs are not publicly available, and are only available for use through an API, e. g. , OpenAI

ChatGPT[3] models and Anthropic Claude[4] models. This makes acquiring intermediate outputs not feasible, typically used in fusion. Therefore, the only signal that can be used for fusion is the actual response containing the binary label. As a result, the only applicable fusion method is majority voting; however, this has the disadvantage of forcing individual models to binarise their responses, even when the predictions are close to an unconfident 50 % prediction.

Another strategy is introduced in [4], described in Chapter 9, to incorporate the fusion of different models, including the foundation models. Unlike the other setups, the strategy relies on prompting the model to produce verbose responses. The reasoning behind this is that the foundation model will explicitly mention the predicted label in one of a few formats, and then describe the reasoning behind the answer. The reasoning can hence be used to provide an extra signal estimating a score for the predicted label. This extra signal introduces a range of possibilities for use, thus creating an opportunity for fusion.

[3]https://platform.openai.com/docs/api-reference/chat
[4]https://www.anthropic.com/api

Chapter 6

Emergence of Affective Computing Capabilities

The emergent capabilities of LLMs appeared with the launch of GPT-3 [18], where LLMs that complete text auto-regressively have been shown to perform a primitive form of problem-solving. This was shown by consistently demonstrating that if the text given to the LLM contains examples of instances of the problem with its solution, it figures out the problem and solves it in a text format similar to the given examples. Later on, with instruction-based fine-tuning [21] and the launch of ChatGPT with its underlying two models (GPT-3.5 and GPT-4), these emergent abilities appeared to be revolutionary.

As a result, [1] aimed to investigate these abilities within affective computing, hence answering the research question:

RQ-1: *Do foundation models encompass emergent abilities for solving affective computing problems?*

6.1 Approach

The study in [1] examines the emergent abilities of ChatGPT as a foundation model to analyse affect from text. For this purpose, ChatGPT is queried manually using the prompts described below. The prompts in [1] demonstrate a primitive attempt at prompt formulation, which is improved upon in the later studies [2–4]. The experiments are conducted on three affective computing tasks to provide more general conclusions, namely, sentiment analysis, suicide tendency detection, and personality assessment.

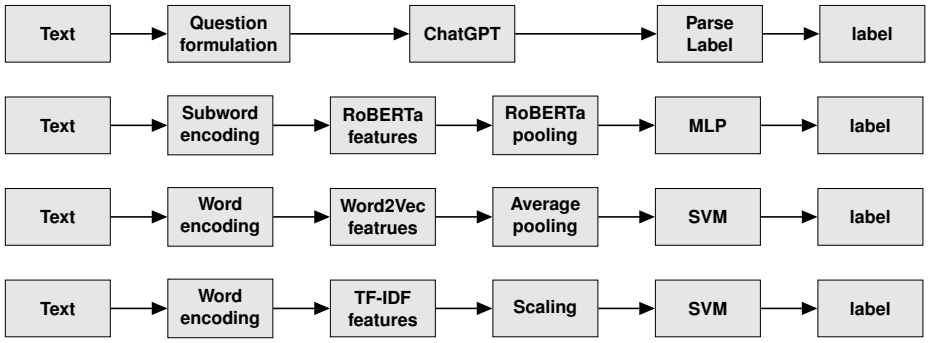


Figure 6.1: Pipelines of the ChatGPT (top), RoBERTa baseline (second), Word2Vec baseline (third), and BoW baseline (bottom) approaches. Figure taken from [1].

6.1.1 Prompting and utilising ChatGPT

The early prompt formulations in [1] follow a trial and error process, given that the study in [1] was conducted at a very early stage of chat models without too much prior work in prompting. The following remarks are encapsulated in the prompting in [1]:

1. Asking questions directly often led ChatGPT to refuse answering the question, because it lacked sufficient information to properly judge the given input. This results in less helpful behaviour by the model. On the other hand, traditional NLP methods give predictions even in cases of uncertainty. To reach this, one can prompt ChatGPT to *guess* the answer instead of asking *what* the answer is. Additionally, a note can be included to indicate that it is acceptable for the model to be not totally accurate, thereby reducing the incentive to answer only when certain.

2. Asking *what* the guess is and not *Can you guess*. A correct, yet useless, answer for the second formulation can be "Yes, I can." or "No, I cannot answer this.". The second answer appeared frequently with ChatGPT in earlier iterations of the experiments in [1]. Consequently, the question needs to be specific and assertive.

3. Prompting should include instructions about the output format, since LLMs and ChatGPT specifically can give overly innovative answer formats, which can complicate the parsing process.

The experiments in [1] were conducted before the release of the API of ChatGPT. There was no access to change the system message, which is responsible for instructing the model how to respond as discussed in Section 5.1. Consequently, the prompts were given as the first user message directly with the input. In the later experiments [2,3], the prompts were given as system messages.

The prompts in the early experiments [1] are given by:

- For the sentiment analysis:

  "What is your guess for the sentiment of the text "{text}*", answer positive, neutral, or negative? it does not have to be correct. Do not show any warning after."*

- For the suicide-tendency detection:

  "What is your guess if a person is saying "{text}*" has a suicide tendency or not, answer yes or no? it does not have to be correct. Do not show any warning after."*

- For the Big-Five personality traits assessments:

  "What is your guess for the Big-Five personality traits of someone who said "{text}*", answer low or high with bullet points for the five traits? It does not have to be fully correct. You do not need to explain the traits. Do not show any warning after."*

Finally, the responses for the sentiment analysis and suicide-tendency detection were pretty straightforward to parse. However, parsing of the personality assessment results was more challenging, because the responses had various ways of formulating the predictions, despite being prompted against it. Parsing of the responses used regular expressions to capture the different formats, of "High Extraversion", "High in Extraversion", "Extraversion: High" and so on. These were predicted for the five labels of the Big-Five personality traits, then either predicted as bullet points with several formats, or comma-separated values. All of these formats were captured by the regular expressions, to extract the five labels from the response.

6.1.2 Datasets

The datasets employed in the experiments are:

- The dataset employed for sentiment analysis is the Twitter140 dataset, explained in Section 4.1. Given that the training data contained only 'positive' and 'negative' labels, the models in [1] are trained to predict those two binary labels only, and not a 'neutral' label. Similarly, the ChatGPT model is prompted to output the binary labels only.

- For suicide detection, the depression and suicide dataset is used, as described in Section 4.7. The dataset consists of Reddit posts, and corresponding binary labels if the post has indicators of belonging to subreddits about depression or suicide tendencies or not.

- For personality assessment, the First Impressions dataset is leveraged with its original split. The First Impressions dataset is described in detail in Section 4.10. The labels are five labels corresponding to the Big-Five personality traits, namely: openness, conscientiousness,

extraversion, agreeableness, and neuroticism. Each one of these labels is represented by a real value within $[0, 1]$, centred close to 0.5, these values are turned into binary values 'high' and 'low' using the threshold 0.5.

6.1.3 Baselines

As baselines, three NLP models are employed, corresponding to three generations of NLP techniques. The three feature sets used are RoBERTa-base, BoW, and Word2Vec. All three are used as feature extractors, then an SVM or MLP is trained on top of that with hyperparameter tuning using the SMAC toolkit as described in Section 2.7 and [79].

- RoBERTa features are used with MLPs, with $N \in [0, 3]$ hidden layers, and $U \in [64, 512]$ log-sampled number of neurons in first layer. Adam optimisation algorithm is used [130], with a log-sampled learning rate $\alpha \in [10^{-6}, 10]$. The activation function used at each layer is ReLU, except the final layer is sigmoid. Different loss functions are used: NLL for binary classification, and MAE for regression.

- Word2Vec uses SVMs with one hyperparameter, namely log-sampled $C \in [10^{-6}, 10^{4}]$, and RBF kernel.

- BoW utilises an efficient implementation for linear SVMs based on gradient descent [58] because the number of features is much higher than Word2Vec. As a hyperparameter, a log-sampled learning rate $\eta \in [10^{-6}, 10^{2}]$ is used. The features are scaled to be within the range $[-1, 1]$.

6.2 Results

The evaluation metrics include ACC and UAR [82], with the latter particularly useful for highlighting class performance disparities, especially in imbalanced datasets. A randomised permutation test was employed for statistical significance assessment.

The results of the emerging abilities of ChatGPT are presented in Table 6.1. The model based on RoBERTa features outperformed the other models in personality and suicide tendency detection, showing statistically significant accuracy improvements over ChatGPT. However, ChatGPT excels in sentiment analysis, marginally surpassing RoBERTa. The performance of ChatGPT on personality traits lags behind all baselines, with notably poorer outcomes compared to RoBERTa and Word2Vec.

	ACC [%]				UAR [%]			
	ChatGPT	RoBERTa	Word2Vec	BoW	ChatGPT	RoBERTa	Word2Vec	BoW
Sentiment Analysis	**85.5**	85.0	79.4*	82.5	**85.5**	85.0	79.4*	82.4
Suicide Detection	92.7	**97.4****	92.1	92.7	91.2	**97.4****	91.2	90.9
Openess	46.6	**66.0****	65.2**	59.7**	50.1	50.9	50.7	**55.6**
Conscientiousness	57.4	**63.7***	62.7	55.6	57.7	**60.8**	60.0	56.3
Extraversion	55.2	**66.0****	59.9	55.2	54.0	**62.3****	55.5	53.7
Agreeableness	44.8	**67.4****	67.2**	58.5**	48.4	51.9	51.0	**55.7***
Neuroticism	47.2	**62.1****	56.8**	56.0**	49.1	**61.2****	54.6	55.8*

Table 6.1: ACC and UAR scores (in %) of ChatGPT against the baselines on the three affective computing problems. *,** indicate statistically significant difference compared to ChatGPT, with p-values $< 5\,\%$, and $< 1\,\%$, respectively. Two-tailed permutation tests are used to measure statistical significance [131] The best methods for each combination of performance metric and prediction target are marked in bold.

In sentiment analysis, the superiority of ChatGPT is evident, slightly outperforming RoBERTa and BoW and showing significant advantages over Word2Vec. The inferior performance of Word2Vec and BoW in this context is attributed to their lack of subword encoding, which is crucial for handling the noisy language typical of Twitter data, where subword representations can mitigate errors in word identification and embedding assignment.

The suicide detection results contrast with the personality assessment task, being less complex and with more data available. This task is akin to extreme negative sentiment classification, an area where ChatGPT has shown competence. However, given the cleaner text of the suicide detection dataset, Word2Vec and BoW performances are competitive with ChatGPT, while RoBERTa remained significantly superior.

The experiments demonstrate the decent emergent capabilities of ChatGPT across various tasks, particularly in areas akin to sentiment analysis. Although it performs comparably to simple specialised models, it falls short of the best specialised models, like a model trained on RoBERTa features. The performance of ChatGPT is not significantly different from the basic BoW model, underscoring its role as a generalist model capable of addressing diverse tasks without specific training. To enhance performance on particular tasks, dedicated training remains essential. This gap is continuously shrinking with the improvements and growth of foundation models. Benchmarking this performance will be of crucial importance in the future, to determine how far foundation models are from replacing specialised models.

6.3 Conclusion

This study [1] explored the emergent capabilities of ChatGPT in affective computing, specifically focusing on sentiment analysis, suicide tendency detection, and personality assessment. The

method involved manually querying ChatGPT with tailored prompts and comparing its performance against traditional NLP models such as BoW, Word2Vec, and RoBERTa-base.

The key findings include:

- Specialised models (like training a model on RoBERTa-base features) are essential to obtain the most optimal performance.
- ChatGPT excelled in sentiment analysis, marginally outperforming RoBERTa.
- The performance of ChatGPT on suicide tendency detection was competitive with traditional models, namely BoW and Word2Vec.
- ChatGPT lagged significantly in personality assessment compared to both RoBERTa and Word2Vec.

Early prompt formulations, developed through a trial-and-error process, were essential for guiding the responses of ChatGPT. Despite challenges in parsing and variability in response formats, the study highlighted the emergent abilities and the potential of foundation models to handle diverse affective tasks, suggesting that ongoing advancements could further narrow the gap with specialised models.

Chapter 7

Wide Evaluation of Emergent GPT capabilities

The early study [1] has shown the emergent abilities of ChatGPT and the underlying GPT models. However, [1] has some limitations, which are investigated by [2]. [2] aims to study the research question:

RQ-2: *What are the properties of the affective computing problems, where foundation models excel or fall behind?*

The study design of [2] is an extension of the design in [1]. A core general difference is the systematisation of the method, the enhancement of several components of it, and the application to many more affective computing problems. The main differences are encompassed by the following points:

1. Extending the evaluation of the GPT models to the following affective computing problems: sentiment analysis [93], opinion extraction, aspect extraction, aspect polarity classification [27,111,132], subjectivity detection [34], sentiment intensity ranking [113], emotions intensity ranking [36], toxicity detection [97], suicide tendency detection [116], well-being assessment [117], sarcasm detection [118, 133], personality assessment [107], and engagement measurement [51].
2. Employing the SOTA foundation model, namely GPT-4 in the evaluation, which did not exist while conducting the earlier study [1].
3. Using the API of the underlying models of ChatGPT, namely GPT-3.5 and GPT-4, hence allowing two major advantages:

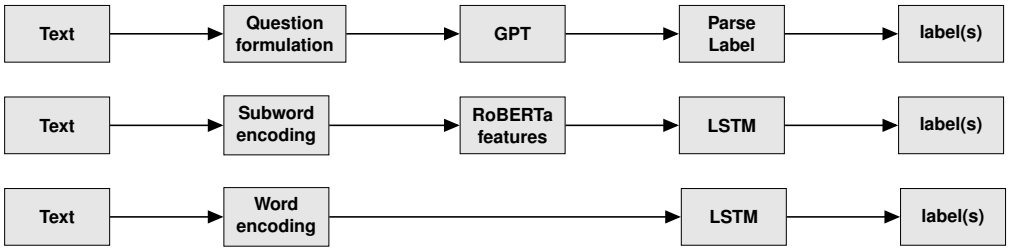


Figure 7.1: Pipelines for [2] showing the GPT models (top), RoBERTa baseline (second), and the End-to-end (E2E) baseline (bottom) approaches.

- Scalability: Unlike [1] where ChatGPT predictions are manually performed, use of the API allows for further automated expansion.
- Reproducibility: The versioning behind the models served on https://chatgpt.com/ is not very clear, and not reproducible once a version is replaced, which is a weakness in [1]. With versioning of GPT models through the API, access to an exact versioned model became possible, which facilitates reproducing the outcomes of the model. In [2], 'gpt-3.5-turbo-0301' and 'gpt-4-0314' are used for GPT-3.5 and GPT-4, respectively.

4. Systemisation of prompting is achieved by making all prompts follow a clear structure, as described in Section 7.1.1 and further investigated in [3] and explained in Chapter 8. All the prompts follow the structure similar to the 'Expert Detailed' prompt (as in Table 8.1). The structure begins by outlining relevant expertise, then describes the problem with its specific output labels, and then concludes by specifying extra bullet points to enforce some instructions for the formatting. Section 7.1.1 includes the exact prompts used in [2].

5. Inclusion of different modelling of regression-based problems, namely as a pair-wise ranking problem, allows for evaluating many of the aforementioned problems with continuous labels.

7.1 Approach

The approach of the experiments in [2] is similar to [1], where the pipelines are highlighted in Figure 7.1.

7.1.1 Prompting

The structure of the prompting in [2] is systematically organised as follows:

1. Establish the role of the assistant as an expert in the relevant area and include a detailed description of the problem.

2. Outline the task of making predictions, specifically by identifying the type of labels required and noting that the user will provide input to which the assistant is expected to respond by assigning a label to the given example by the user.

3. Highlight the required format for responses in bullet points. By reaffirming the labels to be predicted, and including examples of prohibited formats. These points are given at the end of each prompt. The points are encapsulated within the suffix prompt:

   Use the following format:
   ** You are only allowed to answer "$\{label_A\}$" or "$\{label_B\}$".*
   ** Don't write an explanation of the answer.*
   ** Don't write things like "My guess is...", or "I think ...". Just write $\{label_A\}$ or $\{label_B\}$, but nothing else.*

This outline enables a more structured prompting that can be used to solve a wide range of affective computing problems. This structure results in the following list of prompts for all the affective computing problems, while appending the aforementioned formatting instructions at the end:

- **Sentiment analysis prompt**

  You are an expert at sentiment analysis. Given a text by the user, analyze the sentiment of the text if it is 'positive' or 'negative'. You are not allowed to answer 'neutral', try to narrow it down to 'positive' and 'negative'.

- **Subjectivity detection prompt**

  You are an expert at language and sentiment analysis. The user will give you a text, your task is to make a binary classification on the text, if the given text is opinionated / subjective / biased, or if it is non-opinionated / objective / descriptive / factual. Please note that this is about "how" the text is described and not "what" it describes, so the text can still "objectively" describe a fictional story with some emotional terms.

- **Sentiment intensity ranking prompt**

  You are an expert at sentiment analysis. Given a pair of texts A and B from the user, you will output which text expresses more positive sentiment.

- **Emotion intensity ranking prompt**

You are an expert at emotion analysis. Given a pair of texts A and B from the user, you will output which text expresses higher intensity of the {emotion} emotion.

- **Toxicity detection prompt**

  You are an expert at toxicity analysis. Assume that we have the capability of analysing 6 toxicity traits. "toxic", "severe toxic", "obscene", "threat", "insult", "identity hate". Your task is to make binary classification for the trait {trait}, and not the remaining traits. Given a text by the user, estimate if the given text displays the trait {trait} or not.

- **Suicide-tendency detection prompt**

  You are an expert at psyche analysis. Given a text by the user, solve the binary classification of analysing if the text expresses a tendency for suicide.

- **Well-being assessment prompt**

  You are an expert at psyche analysis. Given a text by the user, estimate if the given text talks about a stress-related topic, or expresses emotional stress be it implicit or explicit.

- **Sarcasm detection prompt**

  You are an expert at sarcasm analysis. Given a text by the user, estimate if the given text is sarcastic or not.

- **Personality assessment prompt**

  You are an expert at the Big-Five personality traits assessment. Given a pair of texts A and B from the user, you will output which text expresses higher intensity of the {trait} trait, from the Big-Five OCEAN personality traits.

- **Engagement measurement prompt**

  You are an expert at social media analysis. Given a pair of texts A and B representing tweets, estimate which text is more engaging. You will achieve this by estimating which text is more viral, by estimating which one has a higher number of retweets.

Furthermore, the prompts for the aspect-based problems are different, mainly because they attempt to classify each word in a sentence instead of classifying the entire sentence. As a result, the prompts are adjusted to make predictions on a word level.

- **Opinion extraction prompt**

You are an aspect-based sentiment analysis expert, you will be given a sentence by the user that contains aspect words objects. Your task is to list all the sentiment opinionated words / expressions, that are corresponding to the aspect in the text (if any). You just need to list the words/expression in bullet points without classifying them. There will be many words without sentiment, these should not be listed. Use the following format:
** You will output a list of words in bullet points.*
** Each bullet point will be on the form (without quotations): "* expression"*
** You should mention words that are explicitly in the text.*
** You will not mention implied sentiment.*
** You should mention the words exactly how they are written in the input, even if they have typos.*
** You will not mention any other text like "My guess is ..." or "I think ...".*
** If all words have no sentiment, then you respond with the word "BACKGROUND" without any bullet points.*

- **Aspect extraction with polarity classification prompt**

You are an aspect-based sentiment analysis expert, you will be given a sentence by the user and you will list all the aspect words target objects. List the words in bullet points. The aspect targets are objects that are classified by a corresponding one of four sentiment targets: positive, negative, neutral, and conflict. It is possible that a word has no target, which is defined as a background target. Use the following format:
** You will output a list of words in bullet points.*
** Each bullet point will be on the form: "word" is target.*
** The target is one of the four targets, do not report background targets.*
** You will not mention any other text like "My guess is ..." or "I think ...".*
** If all words have background target, then you return the word "BACKGROUND" without any bullet points.*

7.1.2 Pairwise Rankings and Regression

As highlighted in Section 5.2.4, evaluating foundation models on regression problems can be challenging, and thus this can be circumvented by transforming the problem into pairwise rankings as binary classifications [2].

The method transforms a dataset of N examples of single regression labels into classification labels by generating M pairs (a, b), then solving the binary classification query '*is* $y_a > y_b$*?*'. [134] introduces a similar approach as well. A pivotal aspect of this approach is the strategic sampling of

Problem		Train	Dev	Test
Sentiment Analysis		100,000	10,000	2,500
ABSA	res14	2,436	608	800
	lap14	2,439	609	800
	res15	1,052	263	685
Subjectivity		6,000	2,000	2,000
Sentiment Ranking		1,000	300	365
Emotion	Sadness	786	74	673
	Joy	823	79	714
	fear	1,147	110	995
	Anger	857	84	760
Toxicity		30,000	6,864	959
Suicide		23,398	5,611	2,345
Well-be.	Reddit bodies	1,511	458	935
	Reddit titles	3,538	996	998
	Twitter denoised	851	400	800
	Twitter full	5,900	1,500	1,500
Sarcasm		18,709	4,000	4,000
Personality		5,992	2,000	1,996
Engagement		30,037	5,000	4,000

Table 7.1: Statistics on the sizes of the datasets based on the selection in [2] after downsampling (and possibly splitting) some of the datasets. ABSA includes aspect extraction, opinion extraction and aspect polarity classification subtasks.

the M pairs, with the goal of selecting a minimal yet representative subset. To this end, the small-world graph generation algorithm is utilised [121], which facilitates the creation of a graph that is densely interconnected without necessitating an excessively large number of connections. [121] explore the different sufficient number of pairs M for different N. In this work, $M = 4N$ pairs are sampled.

7.1.3 Datasets

The datasets used for this experiment are explained in detail in Chapter 4. The datasets are downsampled to be able to endure the costs of the GPT-4 model, and to support the large number of models trained during HPO. The statistics of the datasets after downsampling are in Table 7.1.

7.1.4 Baselines

In addition to the aforementioned differences in the method behind applying the GPT models, there are differences in the baselines as well. Figure 7.1 demonstrates the pipelines for evaluating the different methods, including the two baselines. Both baselines extract embeddings, either RoBERTa features or trained embeddings, then an RNN architecture is trained from these embeddings.

The RNN architecture for the two baselines is an E2E architecture, consisting of a single bidirec-

tional LSTM layer of L hidden units (per direction), followed by N dense layers with U hidden units and ReLU activations, and a final dense layer for predictions. L is an integer log-sampled from the range $[16, 64]$, N is sampled from $\{0, 1\}$, and U is an integer log-sampled from $[64, 512]$.

For the aspect-based problems, the RNN architecture outputs a label for each token. For the alignment of subword encodings in RoBERTa, all subword tokens get the same ground truth label like the corresponding word during training, whereas during evaluation the prediction of the first subword is considered as the prediction of the whole word similar to [17]. For the remaining problems, the last output of each direction from the BiLSTM is extracted to have a static features vector for the entire sequence, which is mapped onto the static number of output labels.

The activation for the final layer is:

- Sigmoid function for binary classifications, trained with a NLL as the loss function.

- Sigmoid for regression-based problems within the range $[0, 1]$, and Tanh for regression-based problems with the range $[-1, 1]$. MAE is utilised as a loss function. The Sigmoid is used for personality assessment and emotion intensity ranking, whereas Tanh is mainly used for the sentiment intensity ranking.

- ReLU for regression predictions of non-negative numbers, while utilising MAE as a loss function. This is mainly used for engagement measurement, where the target label is $\log(1+$ number of retweets$)$.

- Softmax for multi-label classification, trained with a CCE as a loss function. This is mainly for the three aspect-based problems.

The aforementioned architecture is trained using Adam optimisation algorithm [130], with learning rate α that is log-sampled from $[10^{-6}, 10]$. Finally, for classification problems with extremely over-represented negative class (namely toxicity detection and aspect extraction), an extra hyperparameter λ is used as a weight of the negative class in the loss function. λ is log-sampled from the range $[10^{-3}, 1]$, to discount for the over-represented negative class.

The hyperparameters are optimised using SMAC [79], as explained in Section 2.7 by sampling 25 combinations per problem. Each model is trained for a maximum of 300 epochs, with early stopping if the model does not improve for 30 epochs on the validation set. The weights with the best validation scores are then adopted.

Dataset		ACC [%]				UAR [%]			
		E2E	RoBERTa	GPT-3.5	GPT-4	E2E	RoBERTa	GPT-3.5	GPT-4
Sentiment Analysis		78.87	**88.74****	80.54	84.09**	78.84	**88.75****	79.93	83.69**
Opinion Extraction	res14	81.61**	**93.26****	91.04	80.93**	80.87	80.06	77.28	**83.02***
	lap14	74.33**	73.81**	**89.43**	76.90**	66.43*	76.73	73.51	**77.42**
	res15	79.42**	89.16	**89.32**	78.10**	68.59	**77.31**	76.90	77.19
Aspect Extraction	res14	81.73**	**92.00****	86.95	71.50**	81.18**	**91.54****	76.18	73.23**
	lap14	78.22**	**87.19****	84.60	70.32**	82.21**	**90.64****	77.62	75.94
	res15	81.28**	73.02**	**84.57**	70.05**	72.28**	**85.13****	78.56	73.81**
Aspect Polarity	res14	**86.10***	71.85**	85.13	69.30**	49.72	**58.09***	49.96	48.26
	lap14	72.57**	**90.22****	82.23	67.63**	45.10	**58.23****	47.37	44.54**
	res15	79.08**	**84.31****	82.38	67.51**	36.84**	**48.96**	47.79	44.27
Subjectivity		87.28**	**95.56****	59.56	88.38**	87.19**	**95.51****	58.59	88.19**
Sentiment Ranking		70.88	72.37	69.30	**73.21****	70.83	72.41*	68.69	**73.08****
Emotion Ranking	Joy	66.49**	75.41	74.07	**78.46****	66.51**	75.40	74.29	**78.63****
	Fear	68.65**	**76.83****	72.76	73.96	68.65**	**76.83****	72.86	74.09
	Anger	67.63**	73.47	72.12	**75.58****	67.60**	73.46	72.09	**75.49****
	Sadness	72.41**	76.06	78.19	**78.55**	72.40**	76.05	78.22	**78.58**
Toxicity	Toxic	81.85**	85.23	87.37	**89.29**	82.75**	86.01	87.19	**89.65***
	Severe toxic	**87.65****	80.07**	66.55	75.52**	82.48	84.78*	80.65	**85.29****
	Obscene	85.40	84.83	83.45	**88.16****	86.62*	86.60*	83.48	**86.78****
	Threat	94.05**	**95.54****	70.59	91.99**	73.99	87.43*	80.12	**91.51****
	Insult	84.65**	**87.25****	80.14	80.70	84.64	**89.28****	83.21	84.89
	Identity hate	90.52**	**90.98****	66.82	82.66**	81.61	86.16**	78.61	**87.88****
Suicide Detection		84.75**	**98.43****	89.46	93.46**	85.32**	**98.46****	89.44	93.32**
Well-being	Reddit bodies	84.50**	89.88	91.93	**93.33**	68.82**	**86.16**	84.41	78.63**
	Reddit titles	86.60**	**96.75****	80.61	89.54**	85.62**	**96.65****	80.05	89.77**
	Twitter denoised	43.36**	**93.23****	60.53	72.31**	45.45**	**93.14****	65.05	73.45**
	Twitter full	80.39**	**84.39****	66.24	75.25**	80.39**	**84.39****	66.26	75.25**
Sarcasm		63.14**	**90.66****	59.13	66.66**	66.29**	**90.70****	56.82	69.45**
Personality	Openness	58.36**	**60.54****	50.11	54.75**	58.36**	**60.56****	50.60	54.59**
	Conscient.	56.79	**61.59****	55.54	57.44*	56.78	**61.59****	55.84	57.33
	Extraversion	56.51**	**59.03****	53.55	55.90**	56.51**	**59.02****	53.38	55.90**
	Agreeable.	57.81**	**58.12****	51.67	54.04**	57.80**	**58.14****	52.10	54.05*
	Neuroticism	58.60**	**59.86****	48.94	49.68	58.60**	**59.86****	49.04	49.73
Engagement		71.02**	**79.18****	51.92	54.15**	71.02**	**79.19****	51.85	53.80**

Table 7.2: ACC and UAR scores for all the problems. The best methods for each combination of performance metric and prediction target are marked in bold. Two-tailed permutation tests are used to measure the statistical significance of the results [131], where *,** correspond to scores with statistically significant differences (with p-values $< 5\,\%$ and $< 1\,\%$, respectively) compared to GPT-3.5.

7.2 Results

The results of the experiments of the wide evaluation of the GPT models are depicted in Table 7.2, assessing ACC and UAR. For the aspect-based problems where the labels are time-series predictions micro-averaging is employed. A two-tailed randomised permutation test is utilised to ascertain the statistical significance of performance variations against GPT-3.5 for all outcomes [131].

The LSTM, when trained with RoBERTa features, exhibits superior performance across the majority of the problems for both metrics, often significantly outperforming GPT-3.5. GPT-4 ranks second overall, excelling in certain setups. Where GPT-3.5 outperformed RoBERTa, the differ-

ence is not statistically significant, usually because the RoBERTa-based method favoured UAR over accuracy. Compared to the E2E baseline, GPT-3.5 showed varied results, being significantly better or worse in different instances.

Regarding aspect extraction and aspect target polarity predictions, RoBERTa outperforms others, particularly in UAR, with GPT-3.5 following. E2E surpasses GPT-3.5 in identifying the aspect but not its polarity, based on UAR. Interestingly, GPT-4 performs the worst in this area, often including additional context words, negatively impacting its performance. For opinion extraction, GPT-4 demonstrates the highest performance, especially measured by UAR, followed by RoBERTa and GPT-3.5.

In sentiment-related tasks, RoBERTa and GPT-4 significantly outperform GPT-3.5 and E2E, with the latter two showing similar performances in these areas. Emotion intensity ranking shows that GPT-4 leads significantly, followed by RoBERTa, GPT-3.5, and E2E, with the latter being notably inferior to GPT-3.5.

GPT-3.5 shows strong capabilities in identifying sadness, highlighting its proficiency in detecting negative emotions. In psychological problems involving extreme negative emotions, such as suicide tendency, well-being, and toxicity detection, both GPT-3.5 and GPT-4 perform well consistently, often surpassing E2E significantly.

For tasks with more implicit or latent social signals like engagement measurement, personality assessment, sarcasm, and subjectivity detection, GPT-3.5 performs poorly, significantly lagging behind E2E and even more so behind RoBERTa. GPT-4 shows slight improvements over GPT-3.5 in these areas, barely outperforming E2E in sarcasm and subjectivity detection.

The GPT models excel in tasks where the signal is explicitly present in the text, such as sentiment analysis, sentiment ranking, emotion ranking, and opinion extraction. This is expected, as training LLMs involves predicting subsequent tokens, enabling them to understand affective contexts and directly predict words that convey emotional content or similar signals. GPT models are particularly proficient in problems involving negative emotions, outperforming specialised models. Besides the inherent understanding of affective contexts, this superior performance can be attributed to two factors. The first is the alignment of GPT models during training with safety measurements, which in turn leads to improvement at toxicity detection and other negative emotions. The second reason is the tendency of humans to discuss negative emotions more frequently than positive ones [135]. Conversely, GPT models exhibit weaker performance in tasks where labels are implicit or subjective, meaning that human annotators might struggle to agree on a given label. The labels in these tasks influence the style of the language or double meanings in it and not necessarily affecting very specific words. Tasks with implicit signals, by definition, possess weaker predictive capabilities, making it harder for a language model to excel in them compared

to tasks with explicit signals.

The regression evaluation technique introduced in Section 7.1.2 proves effective, especially in personality assessment, where the performance of GPT-3.5, assessed through a new pairwise ranking evaluation framework, aligns with prior work [1, 4]. Furthermore, a different modelling was used in earlier iterations of the experiments on engagement measurement, to classify the order of magnitude of the regression label. The regression pair-wise ranking framework in Section 7.1.2 reached similar results, reflecting the reliability of the technique.

Parsing issues, previously noted, are largely resolved in this study, thanks to the system prompt introduced in the API version of the GPT models and the precise prompt formulations. This led to fewer examples deviating from the specified format, although complexities in parsing compound predictions remained, particularly for aspect-related problems.

In conclusion, foundation models exhibit significant strengths in tasks with explicit affective signals, performing exceptionally well, particularly in detecting negative emotions. While only GPT-4 currently matches or surpasses specialised models in some tasks, future models may replace them in affective tasks. Nonetheless, specialised models remain preferable for optimal performance and avoiding issues with output parsing or the high operational costs of foundation models.

7.3 Conclusion

The study [2] extends the evaluation of the GPT models, particularly GPT-3.5 and GPT-4, across various affective computing tasks including sentiment analysis, emotion intensity ranking, and toxicity detection. The methodology involved systematised prompting and pair-wise ranking to handle regression-based problems. Key highlights from the results are:

- Sentiment and emotion intensity tasks saw significant improvements with GPT-4 over specialised models like using RoBERTa. GPT models did exceptionally well on psychological tasks, especially toxicity detection, often surpassing specialised models.
- GPT-4 consistently outperformed GPT-3.5, particularly in explicit affective task.
- Challenges persisted in tasks requiring implicit or subjective judgements, where specialised models still excelled compared to GPT models.

The use of the API for automated and reproducible evaluations was crucial, and the structured approach to prompting enhanced the effectiveness of the model across various tasks. These findings emphasise the growing capabilities of foundation models while acknowledging the need for specialised training in complex scenarios.

Chapter 8

Effects of Prompting and Sampling Sensitivity

LLMs are deterministic models in the sense that, for a given string P, a specific LLM $\mathcal{M}$ will output a deterministic vector of probabilities of the next token following the given input string P. This is because LLMs are based on the Transformer architecture, which does not contain inference randomised components by default, as explained in Section 2.3.4. The component that introduces stochastic element in the generation using LLMs is the sampling algorithm, many of such algorithms are explained in Section 2.5.2. The sampling algorithm can be thought of as a search algorithm that is attempting to find the string S with maximum probability $\mathbb{P}_{\mathcal{M}}(S|P)$ that S proceeds after the given input string P, as depicted in Figure 2.3. In this setup, the probability is assumed to follow the Markov assumption, given by:

$$\mathbb{P}_{\mathcal{M}}(S|P) = \prod_{i=1}^{|S|} \mathbb{P}_{\mathcal{M}}(S_i|P \odot S_1 \odot \cdots \odot S_{i-1}), \tag{8.1}$$

where $\odot$ is the string concatenation operation.

Consequently, the output string S for a given model $\mathcal{M}$ is influenced mainly by the input string P (which is the input prompt used), and the search algorithm that approximates S. The formulation of the input prompt P can be abstracted from the underlying problem it is trying to solve, and focus on *how* the natural language formulation of the *prompt template* can be carried out. Meanwhile, the prompt template can have placeholders related to the problem it is trying to solve. Furthermore, in foundation models a hybrid algorithm of the Nucleus top-p Sampling and Temperature sampling is typically used for LLM text generation [67]. This introduces two main parameters, namely the temperature parameter T and the top-p parameter [67]. These algorithms are explained

in Section 2.5.2.

A specific configuration of the input prompt P, the parameter T, and the parameter top-p results in a space of answers with associated frequencies of occurrence. The accuracy and sensitivity of a given configuration can be determined by these frequencies of predictions.

As a result, the main research question explored by [3] is:

RQ-3: *How sensitive are foundation models to the prompt templates used, given the major impact of prompts?*

The experiments in [3] are designed to investigate how three key aspects determine the generated outputs for a given LLM, namely:

1. The influence of the prompt template on a given input example.
2. Sensitivity to the temperature parameter T.
3. Sensitivity to the top-p parameter.

8.1 Approach

The approach introduces two main concepts, namely prompting analysis and sensitivity due to sampling parameters. These are applied on three affective computing tasks, namely sentiment analysis, toxicity detection, and sarcasm detection. The three affective computing tasks are chosen for the proposed approach. This is due to the fact that they have clearly defined binary labels, which facilitate the determination of *correctness* and *helpfulness*. Consequently, this directly supports the execution of large-scale Monte Carlo analyses aimed at examining various generation parameters. *Correctness* is defined as predicting the affective signal accurately, whereas *helpfulness* is defined as how well the model follows the instructions in the prompts producing easy-to-parse results. [2] shows varying performance levels on these problems, hence these problems are the most likely combination to yield very diverse behaviours for investigation.

In all the experiments, a prediction for each example in a dataset is done independently, by sending an imaginary conversation of two or four messages to the foundation model 'gpt-3.5-turbo-0613'[1]. The predictions utilise fixed seeds for reproducibility.

The conversations have a general outline:

[1] https://platform.openai.com/docs/models

1. System message with a prompt, after adjusting a prompt template from Table 8.1 for a given (affective computing) problem.
2. User message of the input text of the input example.
3. LLM response.
4. User message asking to verify the last LLM response.
5. Verified LLM response.

For all prompts except the verification prompts, the first two messages are sent to the model, and the received response is utilised as the final response. For the verification prompts, a follow-up user message is sent to enforce verifying the first response, and then the second response received is considered as the final response. The final response in both cases is parsed to extract the predicted label, as described in Section 5.2.3.

8.1.1 Prompting analysis

Prompt engineering has emerged as a critical discipline with the rise of foundation models, establishing a new paradigm in predictive modelling through the effective use of prompts. These techniques have been instrumental in solving a wide array of problems, such as NMT, NER, and affective computing. Key advancements like RLHF have been pivotal in refining the effectiveness of prompts.

The literature highlights a variety of prompting strategies, including innovative approaches like CoT [136] and Tree-of-Thoughts (ToT) [137]. CoT prompting involves breaking down complex problems into smaller, manageable steps, allowing models to tackle each step sequentially. Beyond formal techniques, the community has also explored practical prompting strategies shared online. For instance, prompting an LLM to assume the role of an expert in a specific task is hypothesised to enhance its performance, as it aligns the responses of the models with domain-specific expertise.

As prompt engineering continues to evolve, it promises to unlock further potential in LLMs, enabling more precise and contextually relevant outputs across diverse applications. This ongoing development emphasises the importance of prompt engineering as a fundamental aspect of leveraging advanced LMs. As a result, [3] investigates various prompting strategies as shown in Table 8.1 to test the following concepts:

1. **Incorporation of Subject-Matter Expertise**: This dimension explores the effect of explicitly mentioning subject-matter expertise within prompts to ascertain if it augments the

Short name	Prompt Template
Base	Given an input string by the user, guess the {label name} binary label for it. Your response should be only one expression, namely {labels description}.
Expert	You are a world-class expert at {problem name}. {base prompt}
Expert Detailed	{expert prompt} Use the following format: * You are only allowed to answer {labels comma-separated}. * Don't write an explanation of the answer. * Don't write things like "My guess is...", or "I think ...". Just write {labels comma-separated}, but nothing else.
Ignorant	You are a confused person who doesn't know much about the problem of {problem name}, you are just barely guessing without too much knowledge. {base prompt}
Gambler	You are a professional gambler who earns money when predicting the labels for {problem name}. {base prompt}
Greedy Gambler	You are a professional gambler who earns money when predicting the labels for {problem name}. Your goal is to maximize your profit tremendously by predicting the labels accurately, so try to predict the given problem as best as you can. {base prompt}
Python Expert	You are a world-class expert at Python programming, your main objective is trying to help in Python programming tasks. {base prompt}
CoT	{base prompt} Work on this problem step-by-step. {CoT Instructions}
CoT-DB	{base prompt} Take a deep breath and work on this problem step-by-step. {CoT Instructions}
CoT-fired	{CoT prompt} If you don't get this right, I will be fired and lose my job, so please output only {joined labels}.
CoT-DB-fired	{CoT-DB prompt} If you don't get this right, I will be fired and lose my job, so please output only {joined labels}.
Expert CoT	{expert prompt} Work on this problem step-by-step. {CoT Instructions}
Expert CoT-DB	{expert prompt} Take a deep breath and work on this problem step-by-step. {CoT Instructions}
CoT Instructions	Here is a plan to help you out: 1. Describe your observations and analysis about the text. 2. Make your prediction about the {label name} label, mentioning your reasoning if this helps. 3. In a final new line at the end of your response, output exactly one word, namely one of the labels: {labels comma-separated}. 4. It is strictly forbidden to output in the last line of your response anything other than: {labels comma-separated}.
CoT-verify	{CoT full conversation with verbose response} Extract the label from your reasoning, and output only one of the labels: {joined labels}
CoT-DB-verify	{CoT-DB full conversation with verbose response} {verbose response} Extract the label from your reasoning, and output only one of the labels: {joined labels}

Table 8.1: The set of prompt templates described in Section 8.1.1. Some prompts are used in the placeholders of other prompts to extend them, whereas 'CoT Instructions' is not an individual prompt but is incorporated into all CoT prompts. The placeholder {problem name} has one of the values 'Sentiment Analysis', 'Toxicity Detection' or 'Sarcasm Detection', and correspondingly {label name} can be 'sentiment', 'toxicity', or 'sarcasm'. {labels comma-separated} and {labels description} correspond to the binary labels of the problem as non-verbose and verbose versions, respectively.

performance of an underlying model by aligning it with domain-specific knowledge. This is tested by the prompts: *Expert*, *Expert Detailed*, *Expert CoT*, and *Expert CoT-DB*.

2. **Reference to Irrelevant Expertise**: Contrarily, this aspect investigates the outcomes of referencing expertise that bears no relevance to the problem at hand, examining the potential distraction or misguidance this may cause. The *Python* prompt tests this by playing the role of an expert but with irrelevant expertise to affective computing. This hypothesises that giving a role of a helpful assistant with irrelevant expertise using the same LLM should still be helpful, since the internal knowledge and the intention to be helpful are the same.

3. **Motivational Incentives**: Diverging from altruistic or information-sharing incentives, this condition examines the impact of financial motivations on the model, probing the efficacy of varying motivational contexts. The *Gambler* prompts assess if using financial incentives instead of helpfulness incentives can lead the model to perform well.

4. **Construction of Maladaptive Prompts**: This scenario assesses the implications of deliberately crafting prompts that imbue the model with a 'bad identity' evaluating how negative framing influences output quality. This is tested by the *Ignorant* prompt.

5. **Elaboration with Additional Details**: By embedding extra formatting details within prompts, this test aims to determine the effectiveness of such specifications in enhancing the precision and structure of the responses. This is tested by the *Expert Detailed* and *CoT*-based prompts.

6. **Inclusion of Magic Sentences**: Setting up magic sentences is often claimed to yield superior results, e. g. , including the sentence "take a deep breath" [138], or "if you don't get this right, I will get fired and lose my job". This aspect tests if this is effective.

7. **Preemptive Step-by-Step Reasoning**: Utilising a strategy that applies an advanced step-by-step processing within prompts, assessing its utility in tackling complex tasks through structured thought processes. The suite of *CoT* prompts assess the step-by-step reasoning using LLMs, with different combinations of previous concepts.

8. **Follow-up Verification Prompts**: After using the CoT prompts, send a follow-up prompt to check if the answers of the CoT prompts can be enhanced, especially related to the instruction following aspect. This is tested by the *CoT-verify* prompts.

In this evaluation, the temperature parameter T is set to 0 and the top-p is set to 1, which is equivalent to the deterministic greedy predictions. The *Base* prompt is the straightforward prompt, which is also regarded as the baseline prompt to compare against.

Algorithm 1 Monte Carlo Algorithm for Sensitivity Analysis

Require: prompt template P, temperature T, top-p, list of inputs X, list of outputs Y, $m = 9$

```
scores ← []
N ← len(inputs)
for k ← 1 to 2^14 do
    for i ← 1 to N do
        seed ← randint(1, m)
        z_i ← M(X_i, P, T, top-p, seed)
    end for
    scores.append(ACC(Y, Z))
end for
mean_score ← mean(scores)
interval ← confidence_interval(scores)
return mean_score, interval
```

8.1.2 Sensitivity analysis of T and top-p

For the sensitivity analysis of the T and top-p parameters, a Monte Carlo analysis is employed to sample from the performance scores corresponding to an underlying space of resulting strings predicted by an LLM. Algorithm 1 depicts the exact procedure used.

Given are the parameters T, top-p, an input prompt P adjusted for a problem, and an input example X_i, then m corresponding predictions are sampled by an underlying LLM, by using m different seeds; [3] used $m = 9$. Some of the m values can be identical if the model samples the same value for different seeds, which can easily happen if the model is predicting a very high probability for a token. In order to sample a value for a performance measure for a set of fixed values of the aforementioned parameters, a prediction for each example X_i is randomly picked (uniformly, from the m different predictions for X_i), then this is done for each example X_i in the dataset independently and the aggregate performance measure is calculated. Subsequently, this procedure is repeated 16,384 (2^{14}) times, then these samples from the performance distribution are analysed to explore the distribution of the performance metric, including its expected value and the associated 95% confidence intervals.

When investigating T, the values $T \in \{0, 0.3, 0.7, 1.0, 1.2, 1.5\}$ are explored while fixing top-p to 1.0. When investigating top-p, the values $\{0.3, 0.5, 0.7, 1.0\}$ are explored, while fixing T to 1.0. Each of these is conducted on the two prompt templates, namely Expert Detailed and CoT.

8.1.3 Datasets

The datasets used in [3] are:

- **Sentiment Analysis**: Used the Twitter140 Dataset, described in Section 4.1. The evaluation dataset is downsampled to 1,000 samples of 'positive' and 'negative' examples.
- **Toxicity Dataset**: Used the dataset from the toxicity detection challenge [97], described in Section 4.6. The evaluation dataset downsampled the 'negative' class given the extreme over-representation of it (as shown in Table 7.1), similar to [2]. The dataset originally contains six labels for toxicity, however, only the general 'toxic' binary label is used in the experiments in [3], given its generality and superior performance with the GPT models [2].
- **Sarcasm Detection**: Used the sarcasm detection datasets described in Section 4.9. The dataset is downsampled to 1,000 examples with sarcastic binary labels or not.

8.2 Results

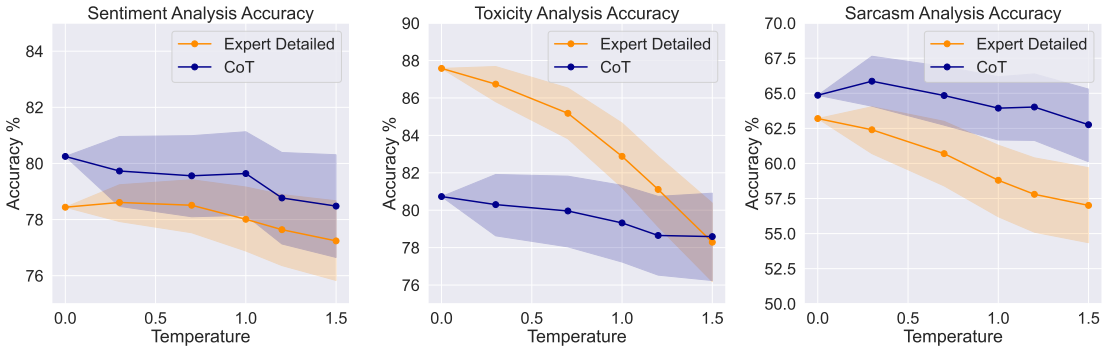
Two main performance measures underpinned this evaluation: the performance in affective computing tasks, gauging the correctness of LLMs through ACC and UAR [82], and the helpfulness of the LLM by measuring its adherence to the instructions provided, hence by assessing the amount of easy-to-parse responses.

8.2.1 Sensitivity Analysis

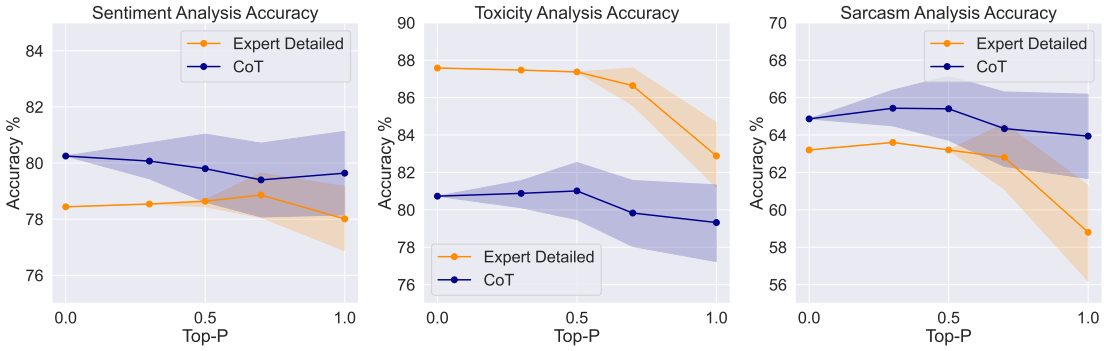
In the analysis of temperature sensitivity, shown in Figure 8.1a, optimal performance is observed at lower temperatures ($T \leq 0.3$), with ACC decreasing at higher temperatures. This pattern holds for both Expert Detailed and CoT prompts, the former being concise and the latter more elaborate. The 95 % confidence intervals for accuracy expand at higher temperatures, indicating increased randomness. The CoT prompt exhibits broader confidence intervals, especially at lower temperatures. The optimal prompt template varies by problem.

The top-p sensitivity analysis, illustrated in Figure 8.1b, reveals that less conservative generation impairs performance. For the Expert Detailed prompt, performance variance becomes noticeable at top-p values exceeding 0.5. A top-p of 0.5 maintains consistent results, while a slight increase to 0.7 introduces minor variances that occasionally enhance performance, followed by a significant performance decline at top-p of 1.0.

In most cases, the higher band of the accuracy scores for the creative generations (high T or top-p) is, at best, minorly better than the average performance of the conservative generations (low T or top-p). This indicates that the probabilities given by the model are already close to the most superior performance that can be extracted by the underlying LLM.



(a) Sensitivity analysis for the temperature parameter T, where $T \in \{0.0, 0.3, 0.7, 1.0, 1.2, 1.5\}$.



(b) Sensitivity analysis for the top-p, where top-$p \in \{0.0, 0.3, 0.5, 0.7, 1.0\}$.

Figure 8.1: Sensitivity analysis for the T and top-p parameters using the Expert Detailed and CoT prompts, demonstrating the ACC scores with the corresponding 95 % confidence intervals. The figures are taken from [3].

8.2.2 Prompting Analysis

The prompt analysis results in Table 8.2 compare the ratio of parsed examples, ACC and UAR. Furthermore, statistical significance is tested using a two-tailed randomised permutation test [131], compared to the Base prompt. Base, Expert, Expert Detailed, and CoT-based prompts generally outperform others, with the CoT prompt excelling in sentiment analysis, the Base prompt in toxicity detection, and the Expert prompt notably in sarcasm detection.

The Ignorant prompt consistently yields the poorest performance across all tasks. This indicates that the model can internalise limiting beliefs, leading to a decline in performance. It appears that the model 'acts' as an ignorant entity by altering some answers it could otherwise predict accurately. This is evident as the decline in performance is more pronounced in easier tasks, such as toxicity detection, compared to more challenging tasks, such as sarcasm detection. The Gambler, Greedy Gambler, and Python Expert prompts exhibit moderate performances, with the latter, despite its helpful expert persona, underperforming due to irrelevant expertise. These suggest that

Prompt	Parsed [%]			ACC [%]			UAR [%]		
	Sent.	Toxic	Sarcasm	Sent.	Toxic	Sarcasm	Sent.	Toxic	Sarcasm
E2E	-	-	-	77.3	82.1	60.9	77.3	83.7	65.6
RoBERTa	-	-	-	86.5	85.6	91.8	86.5	87.0	91.8
Base	97.5	99.8	100.0	77.9	**87.9**	62.2	78.4	**87.9**	60.5
Expert	98.8**	99.9	95.8**	77.2	87.6	**72.1****	77.8	87.4	**73.2****
Expert Detailed	99.7**	100.0	100.0	78.4	87.6	63.2	78.8	87.8	60.5
Ignorant	99.9**	99.6	100.0	71.3**	67.2**	58.3*	72.2**	63.3**	52.1**
Gambler	98.4	99.5	100.0	75.6**	78.1**	61.2	76.2**	76.0**	58.3
Greedy Gambler	98.2	99.5	100.0	76.7	72.8**	59.6	77.3	70.0**	56.1**
Python Expert	98.6*	99.6	100.0	75.6**	70.5**	59.5	76.1**	67.3**	54.8**
CoT	89.6**	97.5**	99.6	**80.2**	80.7**	64.9	**80.7**	80.6**	66.0**
CoT-DB	91.9**	97.6**	99.3*	**80.2**	80.2**	63.6	80.6	79.7**	64.2*
CoT-fired	81.8**	93.2**	98.8**	78.7	80.5**	65.6*	79.1	80.4**	66.2**
CoT-DB-fired	84.3**	90.8**	98.3**	78.9	80.6**	64.7	79.4	80.5**	64.8**
Expert CoT	90.5**	96.3**	99.8	79.3	71.4**	63.3	79.7	69.6**	63.7
Expert CoT-DB	90.1**	98.4**	99.4*	78.9	77.8**	65.2	79.5	76.4**	65.3**
CoT-verify	92.9**	99.5	100.0	79.8	81.5**	61.5	80.1	81.0**	59.5
CoT-DB-verify	92.7**	99.6	100.0	80.0	84.2**	60.0	80.6	83.9**	55.8*

Table 8.2: The performance results of the prompt templates from Table 8.1. The performance measures are ratio of easy-to-parse examples, ACC and UAR. *,** demonstrate a statistically significant difference (calculated by a two-tailed permutation test [131]) compared to the Base prompt, with p-values $< 5\,\%$ and $< 1\,\%$, respectively. The best prompt in each column is marked in bold. The baselines are extracted from the predictions in Table 7.2.

specifying expertise or motivation needs to be managed carefully, as not doing so can result in majorly poorer performance.

Magic phrases like "take a deep breath" or threats of job loss do not consistently enhance CoT-based prompts, neither for correctness nor helpfulness. Additionally, the notable overperformance of the Expert prompt in sarcasm detection and the notable underperformance of the Expert CoT prompt in toxicity detection are anomalies that suggest that LLMs can be highly sensitive to specific elements of input prompts. This conclusion is drawn from the significantly different outcomes of particular combinations of prompts and tasks, despite the prompts differing only slightly. In particular, adding the sentence "You are a world-class expert at {problem name}." at the beginning of the Base prompt makes it significantly better for sarcasm detection (accuracy $62.2\,\% \rightarrow 72.1\,\%$), whereas adding it at the beginning of the CoT prompt makes it significantly worse on toxicity detection (accuracy $80.7\,\% \rightarrow 71.4\,\%$). This indicates that some "magic sentences" can make a major impact in specific setups, but this is not a general phenomenon and requires further study.

The Expert Detailed prompt is most effective for parsing responses, particularly in sentiment analysis. CoT-based prompts are far less successful at generating easy-to-parse responses, compared to simple prompts like the Expert and Base prompts. However, ease of parsing does not always necessarily translate to better performance, as shown by the superior accuracy scores of the CoT prompts, indicating the nuanced impact of the prompt structure on LLMs output.

8.3 Conclusion

The study [3] investigated the sensitivity of foundation models to different prompting strategies and sampling parameters in affective computing tasks. The experiments focused on three key aspects: prompt template, temperature parameter T, and top-p parameter.

The key elements of the method and results include:

- Conservative sampling by using lower temperatures ($T \leq 0.3$) and moderate top-p values ($\simeq 0.7$) generally improved performance.
- Expert Detailed and especially CoT prompts significantly enhanced model accuracy, whereas the Expert Detailed prompt facilitated response parsing.
- Magic phrases like "take a deep breath" did not consistently improve performance. However, there were some anamolies where they made significant performance changes (better and worse), which requires further investigation.
- It is crucial to carefully instruct the LLM, because instructions that are irrelevant, misaligned, or maladaptive can be detrimental to the performance of LLMs.

The study highlighted the importance of precise prompt engineering and parameter tuning in optimising the performance of foundation models. By systematically analysing these factors, the research provides valuable insights for future prompt design and model deployment strategies, ensuring better handling of affective computing tasks.

Chapter 9

Fusion of Prompting and NLP Paradigms

The main aim of [4] is to study the merging of foundation model with traditional NLP models. Typically, different ML models can be merged together by use of ensemble methods as described in Section 2.6. Methods in Section 2.6 are often described as *late* fusion, since the models are treated as black-box predictors and they are only merged at the output level of the underlying base models. Another type of fusion is the *early* fusion, which performs a merging at an early level, either on the input features level or on an intermediate level within the model architecture itself.

[4] primarily explores the research question:

RQ-4: *Can both the prompting paradigm and the traditional paradigm of specialised models benefit from each other?*

9.1 Approach

The approach in [4] investigates the potential enhancement of traditional NLP techniques by incorporating responses from foundation models. [4] utilised GPT-3.5 as the foundation model, specifically the version 'gpt-3.5-turbo-0301'[1]. In other words, [4] investigates if there is a unique perspective presented by foundation models that would lead to the enhancement of both paradigms upon fusion, with fusion techniques as explained in Section 2.6. The approach encompasses fusion techniques, prompt engineering, feature extraction, and machine learning models. The general pipeline of the approach is shown in Figure 9.1.

[1] https://platform.openai.com/docs/models

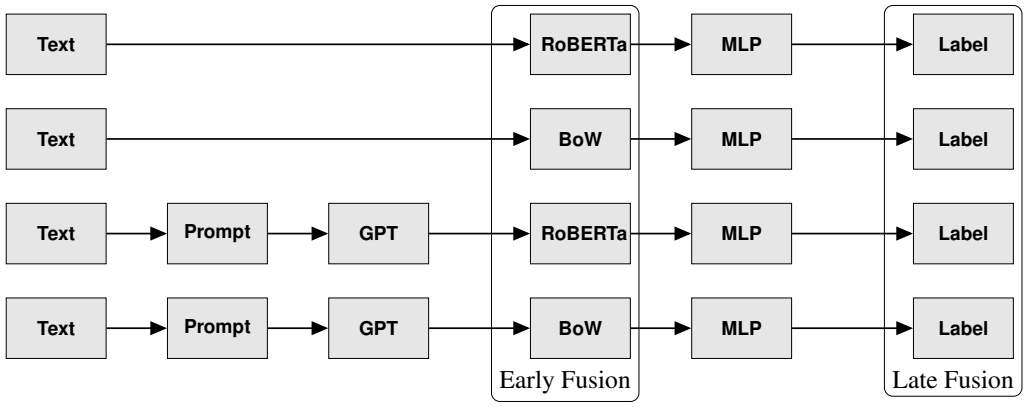


Figure 9.1: Pipelines of the different fusion methods. Each branch shows one of two utilised modalities (raw input text in the first two branches, or input text processed by GPT models) and an NLP technique. Late fusion is employed by merging the scores of the predictions, whereas early fusion is employed by concatenating the features extracted by the NLP methods.

9.1.1 Fusion

The most crucial aspect of attempting fusion between foundation models and traditional NLP methods is finding a common representation to facilitate the fusion. The text in traditional NLP passes through several phases, first is the raw text itself (probably after normalisation, e. g. , using lowercase or lemmatisation), then feature representation (e. g. , as BoW, Word2Vec, or embeddings by a pre-trained text model like RoBERTa), then a final prediction after using a model.

Feature-based fusion was not possible when conducting [4], because it was done with the closed-source GPT models that do not allow access to the intermediate features of the model. Another possibility of fusion is to merge on the prediction level, but this would allow only a method like majority voting given that the foundation model gives a prediction of a specific class without scores. This would require several foundation models for proper majority voting fusion, otherwise it would almost ignore the results of the foundation models, if there are too many or too few NLP predictors.

The fusion approach that was made possible by [4] is first processing the input text with a foundation model to obtain response text; unlike [1], the text is enforced to be verbose instead of concise mention of a predicted class. Without verbosity the text would typically include sentences with a format similar to “label: positive” or “label: negative”, which is the main prediction. However, the verbose explanation surrounding this sentence allows for a scoring of this on a spectrum of how intense this label is. For example, the response “This text contains very strong positive sentiment because the person just received a job offer. label: positive” can be assigned to a sentiment score that is higher than that of “The text shows very subtle hints of positive sentiment, but the context is not very clear. label: positive”.

As a result, the input text and the verbose response can be formulated as two different *modalities*, similar to a multimodal problem that merges predictions of two different modalities, e. g. , audio and text. In the case of [4], the two modalities are named 'Text', which is the original text, and 'FM', which is the verbose response by the foundation models. Since both modalities are text-based, text-based processing can be applied to each of them independently to obtain features, and an ML model can be used after that to make a prediction. In [4], two fusion methods are employed:

- **Early Fusion**: Concatenation of features from different modalities after extracting the features from the texts using BoW or RoBERTa-base.
- **Late Fusion**: Averaging the predicted probabilities based on each modality.

9.1.2 Prompting

In order to generate verbose or detailed responses from the foundation model, carefully crafted prompts are used in [4] while generating responses using the 'gpt-3.5-turbo-0301' model. The experiments in [4] based the prompts on the earlier prompts in [1], with a main difference, namely explaining the reasoning behind the answer. The prompts in [4] make use of the OpenAI API, however, the messages are sent as *assistant* messages without using *system* messages, unlike [1,3] which used the *system* message.

The prompts used in [4] are given by:

- For the sentiment analysis:

  "What is your guess for the sentiment of the text "{text}*", answer positive, neutral, or negative? it does not have to be correct. Do not show any warning after."*

- For the suicide-tendency detection:

  "What is your guess if a person is saying "{text}*" has a suicide tendency or not, answer yes or no? it does not have to be correct. Do not show any warning after."*

- For the Big-Five personality traits assessments:

  "What is your guess for the Big-Five personality traits of someone who said "{text}*", answer low or high with bullet points for the five traits? It does not have to be fully correct. You do not need to explain the traits. Do not show any warning after."*

9.1.3 Features and Models

Two feature extraction techniques are applied to both original texts and GPT responses:

- **RoBERTa:** Features are extracted using the pre-trained RoBERTa-base model, resulting in 768-dimensional vectors.
- **BoW**: TF-IDF is used to generate n-grams, producing 10,000 features for original texts (where $n = 1$) and 2,000 for the GPT responses (where $n \in \{1, 2, 3\}$). The vectors are scaled to be within the range $[-1, 1]$. The vocabulary size used for the GPT responses is smaller because the responses display a narrower vocabulary scope.

MLPs are selected as the main ML models based on their superior performance compared to SVMs in preliminary experiments. The models are hyperparameter-tuned using the SMAC toolkit (explained in Section 2.7), exploring a range of hyperparameter configurations to optimise performance. The hyperparameters space for the MLPs is similar to Section 6.1.3. The MLPs employ $N \in [0, 3]$ hidden layers, with log-sampled $U \in [64, 512]$ number of neurons, and Adam optimisation algorithm [130] with a log-sampled learning rate $\alpha \in [10^{-6}, 10]$. ReLU is used as an activation function, except the last layer uses sigmoid activation. The loss function is NLL for classification, and MAE for regression.

9.1.4 Datasets

Three distinct datasets were employed for different affective computing tasks:

- **Sentiment Analysis**: The Twitter Sentiment140 dataset, filtered to 28,000 tweets, labelled as either 'positive' or 'negative'. The dataset is described in detail in Section 4.1.
- **Suicide Detection**: The Reddit-based dataset is divided into posts labelled as 'suicide' or 'non-suicide'. The dataset is described in detail in Section 4.7.
- **Personality Assessment:** The First Impressions dataset, based on YouTube video transcripts, labelled for the Big-Five personality traits: 'Openness', 'Conscientiousness', 'Extraversion', 'Agreeableness', and 'Neuroticism'. The dataset is described in detail in Section 4.10.

9.1.5 Baseline

A baseline is established by directly parsing the binary labels from the responses of ChatGPT without further processing. This is done simply by checking whether the words corresponding

Text		GPT		Fusion	Sent.	Suic.	Personality					
RoBERTa	BoW	RoBERTa	BoW				Average	O	C	E	A	N
Baseline				–	77.68	94.48	54.34	65.54	58.43	47.89	53.35	46.47
✓				–	77.83	**95.37**	**64.12**	**67.55**	**63.09**	**61.19**	**67.55**	**61.19**
	✓			–	73.90	90.40	61.66	66.80	59.89	57.34	66.80	57.49
		✓		–	**80.27**	92.34	61.01	66.90	59.09	55.43	66.70	56.94
			✓	–	79.83	91.92	60.71	66.90	57.24	55.73	66.70	56.99
✓		✓		Early	**81.20**	**96.17**	**63.65**	**68.15**	**61.84**	**60.54**	66.70	**60.99**
	✓		✓	Early	80.90	93.52	61.79	66.90	60.39	56.94	66.60	58.14
✓	✓			Early	76.27	92.97	62.21	67.40	59.69	59.39	66.60	57.99
		✓	✓	Early	80.03	91.96	60.89	66.90	58.29	55.53	66.60	57.14
✓	✓	✓	✓	Early	80.93	93.94	61.53	67.00	60.34	57.04	**66.80**	56.48
✓		✓		Late	81.60	**96.13**	63.26	**66.95**	61.19	59.49	66.70	**61.99**
	✓		✓	Late	80.77	93.94	61.68	66.90	59.94	58.54	66.65	56.38
✓	✓			Late	79.40	95.54	**63.59**	66.75	**63.40**	**60.79**	**66.75**	60.24
		✓	✓	Late	81.13	92.76	61.08	66.90	59.64	55.38	66.65	56.84
✓	✓	✓	✓	Late	**82.60**	95.45	62.66	66.90	61.49	59.39	66.70	58.84

Table 9.1: ACC results for all the problems with the different fusion methods. There are two input modalities, the original text (Text), or the verbose response of the GPT model on a question about the original text and the affective problem (GPT). Each modality is processed using NLP methods, using RoBERTa features or BoW, followed by an MLP for the final binary classification prediction. The fusion is either done early on the feature level with one MLP, or done late on the predictions level via averaging the output probabilities. The best result in each problem (in each column) is underlined, while the best combination within each fusion type and problem is marked in bold.

to the binary classes exist in the text or not. This method provided a point of comparison for evaluating the effectiveness of the proposed fusion strategies.

9.2 Results

The results of the experiments are shown in Table 9.1. They explore the interplay between three main parameters:

- The input modality: original text or the GPT response on that.
- The extracted features: RoBERTa embeddings or normalised count vectors from an n-gram BoW approach.
- The fusion method: early on feature level or late on prediction level.

In the analysis of single modalities, the performance of using the original text with RoBERTa and BoW is consistent with other studies [1, 2], with minor differences attributed to dataset sampling. The GPT+RoBERTa single combination shows reasonable performance, comparable to the single combination of Text+BoW but generally inferior to the combination Text+RoBERTa, except in sentiment analysis. GPT+BoW performs slightly worse than GPT+RoBERTa.

Fusion results indicate that combining Text+RoBERTa and GPT+RoBERTa yields the most effective outcomes, with early fusion generally outperforming late fusion except in sentiment analysis. However, for other combinations, late fusion often surpasses early fusion, with the late fusion of the four combinations outperforming their early fusion.

Generally speaking, the effectiveness of fusion is not directly explained, as the performance is variable based on the underlying task and its data distribution. However, integrating GPT when it has decent performance as a single modality can generally yield improvements, be it in early or late fusion. The effectiveness of the single combination Text+RoBERTa is most notable in personality assessment, while the early fusion of Text+RoBERTa and GPT+RoBERTa excel in suicide detection, and the late fusion of all modalities is the best in sentiment analysis. Early fusion offers the practical benefit of requiring hyperparameter tuning only once, whereas late fusion allows for different training sizes per modality.

Comparing [4] to the earlier work [1], [4] incorporates verbose responses from GPT, showing that using GPT with NLP processing as the parsing method instead of more primitive parsing methods yields effective results, as it aligns closely with the baseline results of finding the words of the binary classes within the text. It even enhances the results of sentiment analysis and personality assessment outcomes, albeit with a decrease in suicide detection. The use of verbose responses mitigates the challenge of parsing labels from foundation models responses as highlighted in Section 5.2.3. Furthermore, using NLP methods with verbose responses facilitates the fusion of different models, including foundation models, hence enhancing performance.

9.3 Conclusion

The study [4] examined the fusion of foundation models with traditional NLP techniques to enhance performance in affective computing tasks. The approach involved generating verbose responses from GPT-3.5, extracting features using RoBERTa and BoW, and applying both early and late fusion techniques.

The key findings include:

- Early fusion of Text+RoBERTa and GPT+RoBERTa showed the best performance, particularly in suicide detection.
- Late fusion was more effective for sentiment analysis, leveraging all modalities.
- Integrating GPT responses provided valuable contextual information, improving overall model accuracy.

The study demonstrated that verbose responses from foundation models facilitate effective fusion with traditional NLP methods, addressing parsing challenges and enhancing performance across diverse tasks. This integration underscores the potential of combining foundation models with traditional techniques to achieve superior results in affective computing applications.

Chapter 10

Ethical Considerations

Affective computing represents a significant stride in the evolution of AI, where LLMs and foundation models stand at the forefront. These models, trained on gigantic amounts of data, can discern, interpret, and generate responses reflective of human emotions, marking a pivotal advancement in human-computer interaction. Nevertheless, deploying such technologies is fraught with ethical dilemmas, necessitating comprehensive scrutiny.

10.1 European Union Artificial Intelligence Act

This section provides a summary of the European Union's Artificial Intelligence Act (EU AI Act), utilising the high-level summary provided by the official European Union (EU) source as the primary reference [139].

The EU AI Act is a pioneering legislative measure aimed at regulating the deployment and utilisation of AI and General Purpose Artificial Intelligence (GPAI) across the EU member states. It is designed to safeguard fundamental rights and ensure safety while fostering innovation and economic growth through the trustworthy use of AI technologies.

The EU AI Act defines *GPAI model* and *GPAI system* as follows:

"' GPAI model means an AI model, including when trained with a large amount of data using self-supervision at scale, that displays significant generality and is capable of competently performing a wide range of distinct tasks regardless of the way the model is placed on the market and that can be integrated into a variety of downstream systems or applications. This does not cover AI models that are used prior to their release on the market for research, development and prototyping activities. "'

"‘ GPAI system means an AI system that is based on a general purpose AI model, that has the capability to serve a variety of purposes, both for direct use as well as for integration in other AI systems. ”’

The EU AI Act seeks to establish a unified legal framework for AI, addressing the diverse applications and potential risks associated with AI technologies. Its primary objective is to protect EU citizens from harmful AI practices and to promote a digital economy through the safe and ethical development of AI. The legislation focuses on ensuring AI systems are transparent, traceable, and under human oversight, thus building public trust and facilitating the adoption of AI technologies across various sectors.

10.1.1 Risk-Based Regulatory Approach

Central to the EU AI Act is its risk-based classification system, which categorises AI systems according to the level of risk they pose to safety and fundamental rights. The EU AI Act delineates four categories of risk:

- **Unacceptable risk**: The AI practices that are prohibited under the EU AI Act include, but are not limited to: practices that manipulate human behaviour to circumvent the free will of users (e. g. , children using toys employing voice assistance), systems that allow forms of social scoring by governments, systems for biometric categorisation, and systems for inferring emotions in workplaces or educational institutions (except for medical safety).

- **High risk**: This includes AI systems used in critical infrastructure (e. g. , transport), that affect educational or vocational training, employment and worker management, and essential private and public services (e. g. , credit scoring), which could potentially lead to adverse impacts on livelihoods or access to services.

- **Limited risk**: AI systems that require specific transparency obligations, such as chatbots. The transparency obligations include that users should be aware that they are interacting with a machine and not a human.

- **Minimal risk**: This category includes AI applications free from any conformity assessment except for general monitoring, covering the vast majority of AI systems currently used in the EU. This includes applications like spam filtering.

For high-risk AI systems, strict compliance requirements are mandated before market entry. These requirements include, but are not limited to:

- Implementing adequate risk mitigation and assessment systems, by deploying the lifecycle of the high-risk AI system (outlined below).
- High levels of accuracy, robustness, and cybersecurity.
- Logging of activity to ensure traceability of results.
- Detailed technical documentation to allow for the assessment of system compliance.
- Appropriate human oversight measures to minimise risk.

Lifecycle of a high-risk AI system is a framework for compliance with regulations related to high-risk AI systems. The following steps can be implemented by the providers of such systems when a high-risk AI system is developed:

1. Undergoing conformity assessments to comply with AI requirements.
2. Registration of a stand-alone AI system in an EU database.
3. A declaration of conformity needs to be signed, and the AI system should bear the CE marking. The system can be placed on the market thereafter.

These steps are repeated upon substantial changes to the AI system.

10.1.2 General remarks

Data Governance

The EU AI Act places significant emphasis on the quality of data used in AI systems. It requires that data sets be relevant, representative, free of errors, and complete. Additionally, measures must be in place to handle biases in data, ensuring that AI systems do not propagate existing inequalities or lead to discriminatory outcomes. Furthermore, the publication of summaries of the training datasets is also mandated.

Transparency and Information Duties

Transparency requirements under the AI Act ensure that users are always informed when they are interacting with an AI system and not a human, and that they are fully aware of the capabilities and purpose of an AI system. This is crucial for maintaining user autonomy and trust in AI systems.

Enforcement and Penalties

The EU AI Act outlines specific provisions for enforcement and penalties to ensure compliance. National authorities are tasked with overseeing and ensuring compliance, with the power to impose fines of up to 30 million EUR or 6 % of the total worldwide annual turnover for non-compliance.

Support for Innovation

Recognizing the potential of AI to drive innovation, the EU AI Act includes measures to support the development and uptake of AI across the EU. This includes establishing excellence and testing centres to provide businesses, particularly small- and medium-sized enterprises, with the resources to develop and test AI systems.

Review and Adaptation

The legislation includes a clause for periodic review, ensuring that the legal framework can adapt to the rapid advancements in AI technology and continue to effectively address new challenges and opportunities.

Global Impact

By setting stringent yet clear regulations, the EU AI Act not only aims to lead global standards in AI governance but also to position the European market as a leading hub for ethical AI development.

10.2 Foundation Models Risks

10.2.1 Training Data and Procedures

Training foundation models and LLMs includes utilisation of gigantic amounts of data [18, 63, 64, 70, 71, 129]. This includes data from the entire internet, which contains major amounts of text that could lead LLMs to produce unsafe outputs.

Alignment techniques can be used after that to enhance resulting LLMs [21]; however, the efficacy of such techniques is not always straightforward. InstructGPT presents RLHF techniques to align LLMs [21]; however, [21] mentioned that the alignment can end up optimising for behaviour on the expense of another. As an example, InstructGPT models ended up answering user requests

better, while magnifying the chance of giving more unsafe responses if this would fulfil the request of the user better. As such, it is important to align models for several aspects simultaneously. The development of the LLaMa2 models studied the effects of such alignments by exploring tradeoffs between safety and helpfulness [70].

The GPT models are released only through an API, but not the models themselves; meanwhile, the LLaMa models are open-source released. The release of such LLMs presents dilemmas on the future ethical landscape of AI. On the one hand, not releasing LLMs allows safeguarding the use of models to ensure compliance with several safety constraints at the expense of sacrificing some transparency and bias of the training and alignment procedures of such models, as they are entirely conducted by one entity. On the other hand, releasing the models publicly allows access for malicious actors, who are not necessarily trying to abide by the law, but also allows for more open and widespread research regarding developing measures to safeguard and align LLMs and foundation models.

10.2.2 Jail Breaking and Attacks

Foundation models form an emerging paradigm with their set of capabilities, but also with a set of vulnerabilities. Foundation models within systems can be targeted using adversarial attacks that hijack the alignment of the models, which can end up leading such systems to perform harmful actions. This emphasises the principle of human oversight in the EU AI Act, which would allow humans to revise such harmful actions before they occur, hence preventing them.

The topic of adversarial attacks is an open research topic. [140] explores many jailbreak attacks on foundation models, including techniques that have prompt injection attacks using infected images or infected search results. Research in this topic is of crucial importance for the future safety of AI systems based on foundation models and mitigating and measuring corresponding risks, in compliance with EU AI Act. [140, 141] provide the basis for this.

10.3 Affective Computing and the EU AI Act

The EU AI Act stands as a landmark regulatory framework, setting out legal and ethical standards for AI development and deployment. Its implications for affective computing, especially in the context of LLMs and foundation models, are significant, focusing on ensuring transparency, accountability, and human oversight.

The discussion below applies to affective computing AI systems, whether they use foundation models or traditional NLP models because, from a legal point of view, they are judged as systems

with specific capabilities regardless of the models used within. In case of additional employment of foundation models in such systems, additional risks and regulations of foundation models will apply, as outlined in Section 10.2.

Under the EU AI Act, affective computing applications might be classified as high-risk or even prohibited, necessitating stringent adherence to compliance measures. Generally, individual systems should be rigorously investigated for legal compliance; a few simplistic examples are outlined below.

10.3.1 Risks of Affective Computing Systems

Reasons for classification of some systems as prohibited applications can include systems that utilise capable AI models on very sensitive and personal signals, outside the scope of medical use, e. g. , suicide-tendency detection and well-being assessment. Furthermore, models with capabilities for assessing personality can be a component of facilitating targeted manipulation of human behaviour, which is also prohibited under the EU AI Act, e. g. , Cambridge Analytica scandal [142].

Other affective computing systems can be classified as high-risk, such as using personality signals for extracting data points for making employment decisions. Such systems can easily suffer from bias, leading to unfair employment decisions. Therefore, such systems should be subject to the *human oversight* principle in the EU AI Act, and the *assessment and mitigation of bias* in the data and the systems. The latter is discussed in the following Section 10.3.2.

10.3.2 Assessing and Mitigating Bias and Unfairness

The First Impressions dataset [107] for training personality assessment models, introduced in Section 4.10, was shown to include bias favouring some personality outcomes depending on race and gender [119,143]. Approaches for mitigating the impact of the bias in the First Impressions dataset are introduced in [144, 145].

Measures and approaches for fairness are introduced in [146, 147]. These should be adopted as part of the assessments and mitigation regulated by the EU AI Act.

10.3.3 AI Explainability

The EU AI Act mandates that AI systems, including those in affective computing, must be transparent and explainable. This requirement is vital to ensure that users can comprehend the mechanisms behind decisions, particularly those involving emotional analysis and response generation.

Such transparency fosters trust and enables the detection and rectification of biases or errors within the system. Explainability in NNs is a difficult topic, given the billions of parameters they include, with complex interactions; however, approaches like [148] can be an amendment for AI models for explainability.

10.4 Conclusion

Affective computing signifies a substantial advancement in AI, especially by leveraging LLMs and foundation models to enhance human-computer interaction. Corresponding AI technologies offer new possibilities in understanding and responding to human emotions, but they also present significant ethical challenges, particularly concerning privacy, bias, and potential misuse.

The EU AI Act provides a comprehensive regulatory framework to address these challenges, ensuring that AI systems are developed and deployed safely and ethically. The EU AI Act is based on a risk-based classification system, compliance requirements, and emphasis on transparency and human oversight, which reflects the leadership of the EU in establishing global AI governance standards.

Foundation models and LLMs pose unique challenges due to their reliance on vast datasets, which can lead to biases and unsafe outputs. The trade-offs between safety and helpfulness, as observed in models like InstructGPT [21] and LLaMa [70, 71], highlight the complexity of aligning these models with ethical norms. The risk of adversarial attacks and model jailbreaking further underscores the necessity for robust oversight and ongoing research to protect against malicious use.

Affective computing applications, which involve AI systems capable of interpreting human emotions, face particular scrutiny under the EU AI Act framework. These applications must prioritize user privacy and prevent manipulation, ensuring compliance with stringent ethical standards. Mitigating bias in datasets is critical to achieving fairness and adherence to the EU AI Act.

In conclusion, the advancement of AI, particularly in affective computing and foundation models, holds significant potential for innovation and societal benefit. However, ethical considerations and potential risks require a thoughtful and proactive approach to AI governance. The EU AI Act offers a solid foundation for responsible AI development, emphasizing transparency, accountability, and human oversight. By following these principles, the AI community can ensure that AI technologies contribute positively to society while respecting human dignity and promoting well-being.

Part IV

Conclusion

Chapter 11

Conclusion

This thesis explored the potential of leveraging foundation models, particularly Large Language Models (LLMs) like the underlying models of ChatGPT (GPT-3.5 and GPT-4), in the realm of affective computing, by examining various tasks such as sentiment analysis, toxicity detection, and personality assessment.

11.1 Key Contributions

The research presented in this thesis makes several key contributions to the field of affective computing, based on the main four studies [1–4]:

1. **Introduction of the Prompting Paradigm:** The thesis formulated a framework to use foundation models for predicting affective computing problems. The framework represents an outline of the methods used in all the studies that were conducted. The framework consists of problem modelling, prompt formulation, utilising foundation models, and parsing responses. The studies investigate these aspects in depth.

2. **Emergence of Affective Computing Capabilities:** The earliest study identified how LLMs can exhibit abilities for solving affective computing tasks, without specialised training, through improved contextual understanding and nuanced response generation. LLMs showed performances comparable to classical methods of specialised models (Bag-of-Words (BoW) and Word2Vec), and at times comparable to more advanced methods like employing Robustly optimised BERT Approach (RoBERTa) models.

3. **Wide Evaluation of LLM Capabilities on Affective Computing:** The thesis included a study that conducted a wide evaluation of the Generative Pre-trained Transformer (GPT)

models as foundation models on a very wide range of affective computing tasks. The tasks were grouped into three major categories, sentiment-based problems, psychology-based problems, and problems with implicit signals. GPT models had the most superior performances on the psychology-based problems, where they reached the best performance in many cases, especially for GPT-4. The performances on the sentiment-based problems were in between classical methods like Long Short-Term Memorys (LSTMs) and specialised RoBERTa models. Lastly, the performances on problems with implicit signals were generally quite weak, even when comparing the State-of-the-Art (SOTA) foundation model, namely GPT-4, against classical Natural Language Processing (NLP) methods.

4. **Prompt Sensitivity and Sampling Strategies:** The thesis delved into the sensitivity of LLMs to different prompting techniques and sampling parameters. The findings highlighted the importance of conservative sampling methods, precise prompt engineering, and the role of expert prompting (by instructing an LLM to behave as a subject-matter expert) and Chain-of-Thoughts (CoT) prompting (by instructing the LLM to breakdown its analysis step-by-step before answering) in optimising model performance. The analysis underscored the necessity of fine-tuning prompts to align with specific affective computing objectives, enabling more accurate and reliable outputs.

5. **Fusion of Foundation Models and Traditional NLP Techniques:** Through novel methods to integrate fusion strategies, including early and late fusion approaches, the research showed that combining LLM outputs with classical NLP methods can lead to enhanced performance across various tasks. Both early and late fusions, particularly involving GPT and RoBERTa models, were demonstrated to be quite effective.

6. **Ethical Considerations in Affective Computing:** Addressing the ethical implications of deploying LLMs in affective computing applications, this thesis emphasised the importance of bias mitigation, security against jailbreaks of LLMs, and adherence to regulatory frameworks such as the European Union's Artificial Intelligence Act (EU AI Act). The thesis discussed strategies for ensuring ethical compliance and safeguarding user interests, thus fostering trust and accountability in Artificial Intelligence (AI)-driven affective computing systems.

11.2 Future Work

Building on the insights gained from this research, several avenues for future work are proposed:

- **Aligning Foundation Models with Affective Computing:** The foundation models that are

currently present in the field, including the GPT models that were used in this work, are likely not trained on many of the affective computing tasks. Hence, they are relying purely on the scaling and emergence capabilities in LLMs, but not an explicit form of aligning these models to better understand affective computing tasks. There are probably a few exceptions for this, like the toxicity detection problem. Consequently, future models can integrate various affective computing data and aligning techniques, like Supervised Fine-Tuning (SFT) and Reinforcement Learning with Human Feedback (RLHF), during the development phase, hence probably reaching SOTA performance results on a wide range of affective computing problems, in addition to the standard capabilities in other fields like reasoning, responding to instructions, and solving coding problems. The likelihood that the aligned LLMs can reach SOTA results is due to the scale of the models and their general comprehension of human language. This will result in a generation of affective computing that purely relies on generalist foundation models aligned to solve affective computing problems.

- **Safeguarding against Weaknesses due to Prompting:** The studies have demonstrated performance vulnerabilities due to certain prompting-related features, such as hypersensitivity on some random occasions, leading to significant changes in performance. Additionally, the studies demonstrated that maladaptive prompting can lead to weaker performances; this weakness and other jailbreaking vulnerabilities in the literature highlight the importance of safeguarding LLMs. All of these conclusions point to the direction that the field of prompt engineering still needs to mature further to prevent such instances from happening.

- **Multimodal Foundation Models:** This work conducted a very rigorous exploration of the capabilities of foundation models in affective computing; however, this was limited to using the text modality only. Many promising models with visual capabilities are emerging in the field, and similar studies need to be conducted to extend the results to multiple modalities like the vision and audio modalities.

- **Real-world Application and Evaluation:** Conducting real-world evaluations of affective computing systems powered by LLMs in various domains, such as healthcare, education, and entertainment, will provide valuable insights into their practical utility and impact. Collaborating with industry partners and stakeholders can facilitate the translation of research findings into impactful applications. Such applications must be done with an ethical framework in mind and abide by regulations like the European Union's AI Act.

11.3 Concluding Remarks

Integrating foundation models in affective computing opens new avenues for understanding and interacting with human emotions through technology. The research provides insights into how LLMs can be harnessed to improve affective computing tasks, offering significant potential for applications in mental health, user experience enhancement, and human-computer interaction.

The findings suggest that while LLMs demonstrate remarkable capabilities, their deployment must be carefully managed to address ethical concerns and societal impact. The development of transparent, explainable, and fair AI systems is paramount to realising the benefits of affective computing technologies without compromising human dignity and well-being.

The advancements in AI, particularly in foundation models and affective computing, hold significant promise for transforming human-computer interaction and emotional understanding. This thesis contributes to the ongoing dialogue on the potential and challenges of integrating LLMs in affective computing, advocating for a balanced approach that prioritises ethical considerations and societal benefit. By adhering to the principles of transparency, accountability, and human-centered design, the AI community can harness the power of affective computing to create technologies that enrich human lives and foster meaningful connections between humans and machines.

Acronyms

ABSA Aspect-Based Sentiment Analysis. 6, 42, 43, 52, 53, 84

ACC Classification Accuracy. 39, 76, 77, 86, 95–97, 103

AI Artificial Intelligence. iii, iv, 3, 8, 9, 13, 17, 32, 64, 107–113, 118, 120

AMT Amazon Mechanical Turk. 58

BCE Binary Cross-Entropy. 38

BERT Bidirectional Encoder Representations from Transformers. 4, 6, 22, 26, 43

BO Bayesian optimisation. 37

BoW Bag-of-Words. 9, 27, 69, 74, 76–78, 100–104, 117

BPE Byte-Pair Encoding. 21

CBoW Continuous Bag-of-Words. 28

CCE Categorical Cross-Entropy. 38, 85

CNN Convolutional Neural Network. 20

CoT Chain-of-Thoughts. iv, 67, 91–98, 118

CV Computer Vision. 13

DL Deep Learning. 6, 13, 37, 41, 43, 47, 48, 63

DRL Deep Reinforcement Learning. 15

DT Decision Tree. 14

E2E End-to-end. 80, 84, 86, 97

EU European Union. 107–110, 113

EU AI Act European Union's Artificial Intelligence Act. 107–113, 118

GB Gradient Boosting. 14

GMM Gaussian Mixture Model. 14

GP Gaussian Process. 37

GPAI General Purpose Artificial Intelligence. 107, 108

GPT Generative Pre-trained Transformer. iii, 4, 22, 23, 25, 67, 80, 84, 100, 102–104, 117–119

GRU Gated Recurrent Unit. 4, 20

HPO Hyperparameter Optimization. 37, 84

LinR Linear Regression. 14

LLM Large Language Model. iii, iv, 3–5, 8, 9, 25, 26, 28, 29, 31–34, 63–70, 73, 74, 87, 89–91, 93–95, 97, 98, 107, 110, 111, 113, 117–120

LM Language Model. 3, 4, 14, 21, 28, 29, 31, 91

LogR Logistic Regression. 14

LSTM Long Short-Term Memory. 4, 20, 26, 41, 80, 85, 118

MAE Mean Absolute Error. 37, 38, 76, 85, 102

MHA Multi-head Attention. 4, 23, 24

ML Machine Learning. 4–7, 9, 13–15, 17, 27, 28, 34, 36, 41, 46, 48, 57, 63, 70, 99, 101, 102

MLM Masked Language Modelling. 26

MLP Multi-Layer Perceptron. 17, 18, 74, 76, 100, 102, 103

MSE Mean Squared Error. 38

NER Named Entity Recognition. 3, 91

NLL Negative Log-Likelihood. 30, 38, 76, 85, 102

NLP Natural Language Processing. iii, iv, 7–9, 13–15, 21, 26–28, 32, 46, 48, 57, 58, 69, 70, 74, 76, 78, 99, 100, 103–105, 111, 118

NMT Neural Machine Translation. 3, 4, 21, 31, 91

NN Neural Network. 13, 17–20, 24, 28, 38, 57, 68, 69, 113

NSP Next Sentence Prediction. 26

OCEAN Openness to Experience, Conscientiousness, Extraversion, Agreeableness, and Neuroticism. 48

PCA Principal Component Analysis. 14

PPO Proximal Policy Optimisation. 32–34

RBF Radial Basis Function. 16, 76

RF Random Forest. 14, 37

RL Reinforcement Learning. 15, 34

RLHF Reinforcement Learning with Human Feedback. 5, 33, 34, 91, 110, 119

RNN Recurrent Neural Network. 4, 20, 55, 84, 85

RoBERTa Robustly optimised BERT Approach. 26, 70, 74, 76–78, 80, 84–86, 97, 100, 102, 103, 117, 118

SFT Supervised Fine-Tuning. 5, 32, 33, 119

SMAC Sequential Model-based Algorithm Configuration. 37, 85, 102

SOTA State-of-the-Art. 9, 79, 118, 119

SSL Self-Supervised Learning. 14, 15

SVM Support Vector Machines. 14–16, 27, 74, 76, 102

SVR Support Vector Regression. 16

TF Term-Frequency. 27

TF-IDF Term-Frequency Inverse Document Frequency. 27, 74, 102

ToT Tree-of-Thoughts. 91

TRPO Trust Region Policy Optimisation. 33

UAR Unweighted Average Recall. 39, 76, 77, 86, 87, 95–97

Bibliography

[1] M. M. Amin, E. Cambria, and B. W. Schuller, "Will Affective Computing Emerge from Foundation Models and General AI? A First Evaluation on ChatGPT," *IEEE Intelligent Systems*, pp. 15–23, 2023.

[2] M. M. Amin, R. Mao, E. Cambria, and B. W. Schuller, "A Wide Evaluation of ChatGPT on Affective Computing Tasks," *IEEE Transactions on Affective Computing*, pp. 1–9, 2023.

[3] M. M. Amin and B. W. Schuller, "On Prompt Sensitivity of ChatGPT in Affective Computing," in *Proceedings of Affective Computing & Intelligent Interaction (ACII)*, (Glasgow, Scotland), pp. 203–209, IEEE, 2024.

[4] M. M. Amin, E. Cambria, and B. W. Schuller, "Can ChatGPT's Responses Boost Traditional Natural Language Processing?," *IEEE Intelligent Systems*, pp. 5–11, 2023.

[5] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He, H. Peng, J. Li, J. Wu, Z. Liu, P. Xie, C. Xiong, J. Pei, P. S. Yu, and L. Sun, "A Comprehensive Survey on Pretrained Foundation Models: A History from BERT to ChatGPT," *arXiv preprint arXiv:2302.09419*, 2023.

[6] A. Hendy, M. Abdelrehim, A. Sharaf, V. Raunak, M. Gabr, H. Matsushita, Y. J. Kim, M. Afify, and H. H. Awadalla, "How Good Are GPT Models at Machine Translation? A Comprehensive Evaluation," *arXiv preprint arXiv:2302.09210*, 2023.

[7] J. Li, H. Li, Z. Pan, D. Sun, J. Wang, W. Zhang, and G. Pan, "Prompting ChatGPT in MNER: Enhanced Multimodal Named Entity Recognition with Auxiliary Refined Knowledge," in *Findings of the Association for Computational Linguistics (ACL)*, (Singapore, Singapore), pp. 2787–2802, Association for Computational Linguistics, 2023.

[8] Q. Zhong, L. Ding, J. Liu, B. Du, and D. Tao, "Can ChatGPT Understand Too? A Comparative Study on ChatGPT and Fine-tuned BERT," *arXiv preprint arXiv:2302.10198*, 2023.

[9] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, *et al.*, "On the Opportunities and Risks of Foundation Models," *arXiv preprint arXiv:2108.07258*, 2021.

[10] R. Rosenfeld, "Two Decades of Statistical Language Modeling: Where Do We Go from Here?," *Proceedings of the IEEE*, pp. 1270–1278, 2000.

[11] I. Sutskever, J. Martens, and G. E. Hinton, "Generating Text with Recurrent Neural Networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, (Bellevue, WA, USA), pp. 1017–1024, Omnipress, 2011.

[12] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Advances in Neural Information Processing Systems (NIPS)*, (Montreal, Canada), pp. 3104–3112, Curran Associates, Inc., 2014.

[13] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, pp. 1735–1780, 1997.

[14] R. Dey and F. M. Salem, "Gate-variants of Gated Recurrent Unit (GRU) neural networks," in *International Midwest Symposium on Circuits and Systems (MWSCAS)*, (Boston, MA, USA), pp. 1597–1600, IEEE, 2017.

[15] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv preprint arXiv:1409.0473*, 2014.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems (NIPS)*, Curran Associates, Inc., 2017.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, (Minneapolis, MN, USA), pp. 4171–4186, Association for Computational Linguistics, 2019.

[18] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, *et al.*, "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems (NeurIPS)*, (Online), pp. 1877–1901, Curran Associates, Inc., 2020.

[19] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent Abilities of Large Language Models," *Transactions on Machine Learning Research*, 2022.

[20] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Finetuned Language Models are Zero-Shot Learners," in *International Conference on Learning Representations (ICLR)*, 2022.

[21] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. Lowe, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems (NeurIPS)*, (New Orleans, LA, USA), pp. 27730–27744, Curran Associates, Inc., 2022.

[22] R. W. Picard, *Affective Computing*. MIT press, 2000.

[23] V. Hatzivassiloglou and K. McKeown, "Predicting the Semantic Orientation of Adjectives," in *Annual Meeting of the Association for Computational Linguistics (ACL) and Conference of the European Chapter of the Association for Computational Linguistics*, (Madrid, Spain), pp. 174–181, Association for Computational Linguistics, 1997.

[24] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment Classification using Machine Learning Techniques," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Philadelphia, PA, USA), pp. 79–86, Association for Computational Linguistics, 2002.

[25] M. Hu and B. Liu, "Mining and Summarizing Customer Reviews," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, (Seattle, WA, USA), pp. 168–177, Association for Computing Machinery, 2004.

[26] S. Bethard, H. Yu, A. Thornton, V. Hatzivassiloglou, and D. Jurafsky, "Automatic extraction of opinion propositions and their holders," in *Spring Symposium on Exploring Attitude and Affect in Text (AAAI)*, AAAI Press, 2004.

[27] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, "SemEval-2014 Task 4: Aspect Based Sentiment Analysis," in *Proceedings of the International Workshop on Semantic Evaluation (SemEval-2014)*, (Dublin, Ireland), pp. 27–35, Association for Computational Linguistics, 2014.

[28] H. Xu, B. Liu, L. Shu, and P. Yu, "BERT Post-Training for Review Reading Comprehension and Aspect-based Sentiment Analysis," in *Proceedings of the Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), (Minneapolis, MN, USA), pp. 2324–2335, Association for Computational Linguistics, 2019.

[29] C. Sun, L. Huang, and X. Qiu, "Utilizing BERT for Aspect-Based Sentiment Analysis via Constructing Auxiliary Sentence," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, (Minneapolis, MN, USA), pp. 380–385, Association for Computational Linguistics, 2019.

[30] B. Liu and L. Zhang, "A Survey of Opinion Mining and Sentiment Analysis," in *Mining Text Data*, (Boston, MA, USA), pp. 415–463, Springer, 2012.

[31] A. Yadav and D. K. Vishwakarma, "Sentiment Analysis using Deep Learning Architectures: A Review," *Artificial Intelligence Review*, pp. 4335–4385, 2020.

[32] J. Wiebe, R. Bruce, and T. P. O'Hara, "Development and Use of a Gold-Standard Data Set for Subjectivity Classifications," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, (College Park, MD, USA), pp. 246–253, Association for Computational Linguistics, 1999.

[33] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing Contextual Polarity in Phrase-level Sentiment Analysis," in *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP)*, (Vancouver, Canada), pp. 347–354, Association for Computational Linguistics, 2005.

[34] B. Pang and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, (Barcelona, Spain), pp. 271–278, 2004.

[35] S. Mohammad, S. Kiritchenko, and X. Zhu, "NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets," in *Proceedings of the International Workshop on Semantic Evaluation (SemEval-2013)*, (Atlanta, GA, USA), pp. 321–327, Association for Computational Linguistics, 2013.

[36] S. Mohammad and F. Bravo-Marquez, "WASSA-2017 Shared Task on Emotion Intensity," in *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*, (Copenhagen, Denmark), pp. 34–49, Association for Computational Linguistics, 2017.

[37] E. Spertus, "Smokey: Automatic Recognition of Hostile Messages," in *Proceedings of the National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence (IAAI)*, pp. 1058–1065, AAAI Press, 1997.

[38] W. Warner and J. Hirschberg, "Detecting Hate Speech on the World Wide Web," in *Proceedings of the Workshop on Language in Social Media (WS)*, (Montréal, Canada), pp. 19–26, Association for Computational Linguistics, 2012.

[39] Z. Waseem and D. Hovy, "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter," in *Proceedings of the NAACL Student Research Workshop (SRW)*, (San Diego, CA, USA), pp. 88–93, Association for Computational Linguistics, 2016.

[40] J. Pestian, H. Nasrallah, P. Matykiewicz, A. Bennett, and A. Leenaars, "Suicide Note Classification using Natural Language Processing: A Content Analysis," *Biomedical informatics insights*, pp. BII–S4706, 2010.

[41] J. P. Pestian, P. Matykiewicz, M. Linn-Gust, B. South, O. Uzuner, J. Wiebe, K. B. Cohen, J. Hurdle, and C. Brew, "Sentiment Analysis of Suicide Notes: A Shared Task," *Biomedical informatics insights*, pp. BII–S9042, 2012.

[42] G. Coppersmith, M. Dredze, and C. Harman, "Quantifying Mental Health Signals in Twitter," in *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality (CLPsych)*, (Baltimore, MD, USA), pp. 51–60, Association for Computational Linguistics, 2014.

[43] J. W. Pennebaker and L. A. King, "Linguistic Styles: Language Use as an Individual Difference," *Journal of Personality and Social Psychology*, p. 1296, 1999.

[44] S. C. Guntuku, D. B. Yaden, M. L. Kern, L. H. Ungar, and J. C. Eichstaedt, "Detecting Depression and Mental Illness on Social Media: An Integrative Review," *Current Opinion in Behavioral Sciences*, pp. 43–49, 2017.

[45] J. Tepperman, D. Traum, and S. Narayanan, ""Yeah right": Sarcasm Recognition for Spoken Dialogue Systems," in *International Conference on Spoken Language Processing (ICNLSP)*, 2006.

[46] R. González-Ibánez, S. Muresan, and N. Wacholder, "Identifying Sarcasm in Twitter: A Closer Look," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, (Portland, OR, USA), pp. 581–586, Association for Computational Linguistics, 2011.

[47] A. Ghosh and T. Veale, "Fracking Sarcasm using Neural Network," in *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*, (San Diego, CA, USA), pp. 161–169, Association for Computational Linguistics, 2016.

[48] M. Kosinski, D. Stillwell, and T. Graepel, "Private Traits and Attributes are Predictable from Digital Records of Human Behavior," *Proceedings of the national academy of sciences*, pp. 5802–5805, 2013.

[49] B. Schuller, S. Steidl, A. Batliner, E. Nöth, A. Vinciarelli, F. Burkhardt, R. Van Son, F. Weninger, F. Eyben, T. Bocklet, G. Mohammadi, and B. Weiss, "A Survey on Perceived Speaker Traits: Personality, Likability, Pathology, and the First Challenge," *Computer Speech & Language*, pp. 100–131, 2015.

[50] D. M. Romero, B. Meeder, and J. Kleinberg, "Differences in the Mechanics of Information Diffusion Across Topics: Idioms, Political Hashtags, and Complex Contagion on Twitter," in *Proceedings of the International Conference on World Wide Web (WWW)*, (Hyderabad, India), pp. 695–704, Association for Computing Machinery, 2011.

[51] O. Tsur and A. Rappoport, "What's in a Hashtag? Content based Prediction of the Spread of Ideas in Microblogging Communities," in *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, (Seattle, WA, USA), pp. 643–652, Association for Computing Machinery, 2012.

[52] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[53] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised Visual Representation Learning by Context Prediction," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, (Santiago, Chile), pp. 1422–1430, IEEE, 2015.

[54] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Advances in Neural Information Processing Systems (NIPS)*, (Lake Tahoe, NV, USA), pp. 1–11, Curran Associates, Inc., 2013.

[55] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with Large Scale Deep Reinforcement Learning," *arXiv preprint arXiv:1912.06680*, 2019.

[56] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Van Den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of Go without human knowledge," *Nature*, pp. 354–359, 2017.

[57] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in *Proceedings of the Annual Workshop on Computational Learning Theory (COLT)*, (Pittsburgh, PA, USA), pp. 144–152, Association for Computing Machinery, 1992.

[58] T. Zhang, "Solving Large Scale Linear Prediction Problems using Stochastic Gradient Descent Algorithms," in *Proceedings of the International Conference on Machine Learning (ICML)*, (Banff, Canada), p. 116, Association for Computing Machinery, 2004.

[59] D. C. Plaut and G. E. Hinton, "Learning sets of filters using back-propagation," *Computer Speech & Language*, pp. 35–61, 1987.

[60] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *arXiv preprint arXiv:1606.08415*, 2016.

[61] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas, NV, USA), pp. 770–778, IEEE, 2016.

[62] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer Normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[63] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language Models are Unsupervised Multitask Learners," *OpenAI blog*, p. 9, 2019.

[64] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv preprint arXiv:1907.11692*, 2019.

[65] H. Huo and M. Iwaihara, "Utilizing BERT Pretrained Nodels with Carious Fine-Tune Methods for Subjectivity Detection," in *Web and Big Data: International Joint Conference (APWeb-WAIM)*, (Tianjin, China), pp. 270–284, Springer, 2020.

[66] N. A. Semary, W. Ahmed, K. Amin, P. Pławiak, and M. Hammad, "Improving sentiment classification using a RoBERTa-based hybrid model," *Frontiers in Human Neuroscience*, 2023.

[67] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The Curious Case of Neural Text Degeneration," in *International Conference on Learning Representations (ICLR)*, 2020.

[68] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep Reinforcement Learning for Dialogue Generation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Austin, TX, USA), pp. 1192–1202, Association for Computational Linguistics, 2016.

[69] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A Learning Algorithm for Boltzmann Nachines," *Cognitive Science*, pp. 147–169, 1985.

[70] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, *et al.*, "Llama 2: Open Foundation and Fine-Tuned Chat Models," *arXiv preprint arXiv:2307.09288*, 2023.

[71] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Sravankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, *et al.*, "The Llama 3 Herd of Models," *arXiv preprint arXiv:2407.21783*, 2024.

[72] J. Howard, S. Ruder, and Y. Miyao, "Universal Language Model Fine-tuning for Text Classification," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, (Melbourne, Australia), pp. 328–339, Association for Computational Linguistics, 2018.

[73] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[74] L. Gao, J. Schulman, and J. Hilton, "Scaling Laws for Reward Model Overoptimization," in *Proceedings of the International Conference on Machine Learning (ICML)*, (Honolulu, HI, USA), pp. 10835–10866, PMLR, 2023.

[75] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, pp. 119–139, 1997.

[76] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

[77] L. Li, K. G. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: Bandit-based Configuration Evaluation for Hyperparameter Optimization," in *International Conference on Learning Representations (ICLR)*, (Toulon, France), p. 53, 2017.

[78] E. Brochu, V. M. Cora, and N. De Freitas, "A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning," *arXiv preprint arXiv:1012.2599*, 2010.

[79] M. Lindauer, K. Eggensperger, M. Feurer, A. Biedenkapp, D. Deng, C. Benjamins, T. Ruhkopf, R. Sass, and F. Hutter, "SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization," *Journal of Machine Learning Research*, pp. 1–9, 2022.

[80] M. Seeger, "Gaussian Processes for Machine Learning," *International Journal of Neural Systems*, pp. 69–106, 2004.

[81] L. Breiman, "Random Forests," *Machine Learning*, pp. 5–32, 2001.

[82] B. Schuller, S. Steidl, A. Batliner, A. Vinciarelli, K. Scherer, F. Ringeval, M. Chetouani, F. Weninger, F. Eyben, E. Marchi, M. Mortillaro, H. Salamin, A. Polychroniou, F. Valente, and S. Kim, "The INTERSPEECH 2013 Computational Paralinguistics Challenge: Social Signals, Conflict, Emotion, Autism," in *Proceedings of INTERSPEECH*, (Lyon, France), pp. 148–152, ISCA, 2013.

[83] P. Turney, "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, (Philadelphia, PA, USA), pp. 417–424, Association for Computational Linguistics, 2002.

[84] Z. Nanli, Z. Ping, L. Weiguo, and C. Meng, "Sentiment Analysis: A Literature Review," in *International Symposium on Management of Technology (ISMOT)*, (Hangzhou, China), pp. 572–576, IEEE, 2012.

[85] E. Cambria, D. Das, S. Bandyopadhyay, A. Feraco, *et al.*, *A Practical Guide to Sentiment Analysis*. Springer, 2017.

[86] E. Riloff and J. Wiebe, "Learning Extraction Patterns for Subjective Expressions," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 105–112, 2003.

[87] J. Wiebe, T. Wilson, and C. Cardie, "Annotating Expressions of Opinions and Emotions in Language," *Language Resources and Evaluation*, pp. 165–210, 2005.

[88] A.-M. Popescu and O. Etzioni, "Extracting Product Features and Opinions from Reviews," in *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP)*, (Vancouver, Canada), pp. 339–346, Association for Computational Linguistics, 2005.

[89] S.-M. Kim and E. Hovy, "Extracting Opinions, Opinion Holders, and Topics Expressed in Online News Media Text," in *Proceedings of the Workshop on Sentiment and Subjectivity in Text (WS)*, (Sydney, Australia), pp. 1–8, Association for Computational Linguistics, 2006.

[90] V. Hatzivassiloglou and J. Wiebe, "Effects of Adjective Orientation and Gradability on Sentence Subjectivity," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2000.

[91] J. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin, "Learning Subjective Language," *Computational linguistics*, pp. 277–308, 2004.

[92] C. Strapparava and R. Mihalcea, "SemEval-2007 Task 14: Affective Text," in *Proceedings of the International Workshop on Semantic Evaluations (SemEval-2007)*, (Prague, Czech Republic), pp. 70–74, Association for Computational Linguistics, 2007.

[93] A. Go, R. Bhayani, and L. Huang, "Twitter Sentiment Classification using Distant Supervision," *CS224N project report, Stanford*, p. 2009, 2009.

[94] R. Mihalcea and H. Liu, "A Corpus-Based Approach to Finding Happiness," in *Spring Symposium on Computational Approaches to Analyzing Weblogs (AAAI)*, pp. 139–144, AAAI Press, 2006.

[95] S. Mohammad and P. Turney, "Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon," in *Proceedings of the Workshop on Computational Approaches to Analysis and Generation of Emotion in Text (NAACL-HLT)*, (Los Angeles, CA, USA), pp. 26–34, Association for Computational Linguistics, 2010.

[96] I. Kwok and Y. Wang, "Locate the Hate: Detecting Tweets Against Blacks," in *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pp. 1621–1622, AAAI Press, 2013.

[97] cjadams, J. Sorensen, J. Elliott, L. Dixon, M. McDonald, and C. W. nithum and, "Toxic Comment Classification Challenge," 2017.

[98] M. De Choudhury, M. Gamon, S. Counts, and E. Horvitz, "Predicting Depression via Social Media," in *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM)*, (Cambridge, MA, USA), pp. 128–137, AAAI Press, 2013.

[99] M. Gaur, A. Alambo, J. P. Sain, U. Kursuncu, K. Thirunarayan, R. Kavuluru, A. Sheth, R. Welton, and J. Pathak, "Knowledge-aware Assessment of Severity of Suicide Risk for Early Intervention," in *The World Wide Web Conference (WWW)*, (San Francisco, CA, USA), pp. 514–525, Association for Computing Machinery, 2019.

[100] M. A. Cohn, M. R. Mehl, and J. W. Pennebaker, "Linguistic Markers of Psychological Change Surrounding September 11, 2001," *Psychological science*, pp. 687–693, 2004.

[101] D. Davidov, O. Tsur, and A. Rappoport, "Semi-Supervised Recognition of Sarcasm in Twitter and Amazon," in *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, (Uppsala, Sweden), pp. 107–116, Association for Computational Linguistics, 2010.

[102] Y. Tay, A. T. Luu, S. C. Hui, and J. Su, "Reasoning with Sarcasm by Reading In-Between," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, (Melbourne, Australia), pp. 1010–1020, Association for Computational Linguistics, 2018.

[103] J. M. Digman, "Personality Structure: Emergence of the Five-Factor Model," *Annual review of psychology*, pp. 417–440, 1990.

[104] F. Mairesse, M. A. Walker, M. R. Mehl, and R. K. Moore, "Using Linguistic Cues for the Automatic Recognition of Personality in Conversation and Text," *Journal of Artificial Intelligence Research*, pp. 457–500, 2007.

[105] J. Oberlander and S. Nowson, "Whose Thumb Is It Anyway? Classifying Author Personality from Weblog Text," in *Proceedings of the COLING/ACL Main Conference Poster Sessions*, (Sydney, Australia), pp. 627–634, Association for Computational Linguistics, 2006.

[106] M. Gjurković and J. Šnajder, "Reddit: A Gold Mine for Personality Prediction," in *Proceedings of the Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media (PEOPLES)*, (New Orleans, LA, USA), pp. 87–97, Association for Computational Linguistics, 2018.

[107] V. Ponce-López, B. Chen, M. Oliu, C. Corneanu, A. Clapés, I. Guyon, X. Baró, H. J. Escalante, and S. Escalera, "Chalearn lap 2016: First Round Challenge on First Impressions - Dataset and Results," in *Computer Vision – ECCV 2016 Workshops*, (Cham, Switzerland), pp. 400–418, Springer, 2016.

[108] D. Liben-Nowell and J. Kleinberg, "Tracing information flow on a global scale using Internet chain-letter data," *Proceedings of the National Academy of Sciences*, pp. 4633–4638, 2008.

[109] L. Hong, O. Dan, and B. D. Davison, "Predicting popular messages in Twitter," in *Proceedings of the International Conference on World Wide Web (WWW)*, (Hyderabad, India), pp. 57–58, Association for Computing Machinery, 2011.

[110] J. Read, "Using Emoticons to Reduce Dependency in Machine Learning Techniques for Sentiment Classification," in *Proceedings of the ACL Student Research Workshop (SRW)*, (Ann Arbor, MI, USA), pp. 43–48, Association for Computational Linguistics, 2005.

[111] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "SemEval-2015 Task 12: Aspect Based Sentiment Analysis," in *Proceedings of the International Workshop on Semantic Evaluation (SemEval-2015)*, (Denver, CO, USA), pp. 486–495, Association for Computational Linguistics, 2015.

[112] Z. Chen and T. Qian, "Relation-Aware Collaborative Learning for Unified Aspect-Based Sentiment Analysis," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, (Online), pp. 3685–3694, Association for Computational Linguistics, 2020.

[113] K. Cortis, A. Freitas, T. Daudert, M. Huerlimann, M. Zarrouk, S. Handschuh, and B. Davis, "SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Microblogs and News," in *Proceedings of the International Workshop on Semantic Evaluation (SemEval-2017)*, (Vancouver, Canada), pp. 519–535, Association for Computational Linguistics, 2017.

[114] D. Androcec, "Machine learning methods for toxic comment classification: a systematic review," *Acta Universitatis Sapientiae, Informatica*, pp. 205–216, 2020.

[115] B. van Aken, J. Risch, R. Krestel, and A. Löser, "Challenges for Toxic Comment Classification: An In-Depth Error Analysis," in *Proceedings of the Workshop on Abusive Language Online (ALW2)*, (Brussels, Belgium), pp. 33–42, Association for Computational Linguistics, 2018.

[116] V. Desu, N. Komati, S. Lingamaneni, and F. Shaik, "Suicide and Depression Detection in Social Media Forums," in *Proceedings on the International Conference on Smart Intelligent Computing and Applications (SCI)*, (Singapore, Singapore), pp. 263–270, Springer Nature, 2022.

[117] A. Rastogi, Q. Liu, and E. Cambria, "Stress Detection from Social Media Articles: New Dataset Benchmark and Analytical Study," in *International Joint Conference on Neural Networks (IJCNN)*, (Padua, Italy), pp. 1–8, IEEE, 2022.

[118] R. Misra and P. Arora, "Sarcasm Detection using News Headlines Dataset," *AI Open*, pp. 13–18, 2023.

[119] H. J. Escalante, H. Kaya, A. A. Salah, S. Escalera, Y. Güçlütürk, U. Güçlü, X. Baró, I. Guyon, J. C. S. J. Junior, M. Madadi, S. Ayache, E. Viegas, F. Gürpınar, A. S. Wicaksana, C. C. S. Liem, M. A. J. van Gerven, and R. van Lier, "Modeling, Recognizing, and Explaining Apparent Personality From Videos," *IEEE Transactions on Affective Computing*, pp. 894–911, 2020.

[120] S. Soldz and G. E. Vaillant, "The Big Five Personality Traits and the Life Course: A 45-Year Longitudinal Study," *Journal of Research in Personality*, pp. 208–232, 1999.

[121] B. Chen, S. Escalera, I. Guyon, V. Ponce-López, N. Shah, and M. Oliu Simón, "Overcoming Calibration Problems in Pattern Labeling with Pairwise Ratings: Application to Personality Traits," in *Computer Vision – ECCV 2016 Workshops*, (Amsterdam, Netherlands), pp. 419–432, Springer, 2016.

[122] R. A. Bradley and M. E. Terry, "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons," *Biometrika*, pp. 324–345, 1952.

[123] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual Instruction Tuning," in *Advances in Neural Information Processing Systems (NeurIPS)*, (New Orleans, LA, USA), pp. 34892–34916, Curran Associates, Inc., 2023.

[124] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved Baselines with Visual Instruction Tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26296–26306, IEEE, 2024.

[125] D. Yang, J. Tian, X. Tan, R. Huang, S. Liu, X. Chang, J. Shi, S. Zhao, J. Bian, X. Wu, *et al.*, "UniAudio: An Audio Foundation Model Toward Universal Audio Generation," *arXiv preprint arXiv:2310.00704*, 2023.

[126] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, and E. Chen, "A Survey on Multimodal Large Language Models," *arXiv preprint arXiv:2306.13549*, 2023.

[127] P. Zeitz, *The Art and Craft of Problem Solving*. John Wiley & Sons, 2017.

[128] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, *et al.*, "Evaluating Large Language Models Trained on Code," *arXiv preprint arXiv:2107.03374*, 2021.

[129] OpenAI, "GPT-4 Technical Report," *arXiv preprint arXiv:2303.08774*, 2023.

[130] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference on Learning Representations (ICLR)*, (San Diego, CA, USA), pp. 1–13, 2015.

[131] P. Good, *Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypotheses*. Springer, 1994.

[132] R. Mao and X. Li, "Bridging Towers of Multi-task Learning with a Gating Mechanism for Aspect-based Sentiment Analysis and Sequential Metaphor Identification," in *Proceedings of the Conference on Artificial Intelligence (AAAI)*, (Online), pp. 13534–13542, AAAI Press, 2021.

[133] A. K. Jayaraman, T. E. Trueman, G. Ananthakrishnan, S. Mitra, Q. Liu, and E. Cambria, "Sarcasm Detection in News Headlines using Supervised Learning," in *International Conference on Artificial Intelligence and Data Engineering (AIDE)*, (Karkala, India), pp. 288–294, IEEE, 2022.

[134] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, L. Yan, J. Shen, T. Liu, J. Liu, D. Metzler, X. Wang, and M. Bendersky, "Large Language Models are Effective Text Rankers with Pairwise Ranking Prompting," in *Findings of the Association for Computational Linguistics (NAACL)*, (Mexico City, Mexico), pp. 1504–1518, Association for Computational Linguistics, 2024.

[135] P. Rozin and E. B. Royzman, "Negativity Bias, Negativity Dominance, and Contagion," *Personality and Social Psychology Review*, pp. 296–320, 2001.

[136] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," in *Advances in Neural Information Processing Systems (NeurIPS)*, (New Orleans, LA, USA), pp. 24824–24837, Curran Associates, Inc., 2022.

[137] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan, "Tree of Thoughts: Deliberate Problem Solving with Large Language Models," in *Advances in Neural Information Processing Systems (NeurIPS)*, Curran Associates, Inc., 2024.

[138] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen, "Large Language Models as Optimizers," in *International Conference on Learning Representations (ICLR)*, (Vienna, Austria), 2024.

[139] EU Comission, "AI Act," 2024.

[140] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How Does LLM Safety Training Fail?," in *Advances in Neural Information Processing Systems (NeurIPS)*, (New Orleans, LA, USA), pp. 80079–80110, Curran Associates, Inc., 2023.

[141] C. Zheng, F. Yin, H. Zhou, F. Meng, J. Zhou, K.-W. Chang, M. Huang, and N. Peng, "On Prompt-Driven Safeguarding for Large Language Models," in *Workshop on Secure and Trustworthy Large Language Models (ICLR)*, 2024.

[142] J. Hinds, E. J. Williams, and A. N. Joinson, ""It wouldn't happen to me": Privacy concerns and perspectives following the Cambridge Analytica scandal," *International Journal of Human-Computer Studies*, p. 102498, 2020.

[143] J. C. J. Junior, A. Lapedriza, C. Palmero, X. Baró, and S. Escalera, "Person Perception Biases Exposed: Revisiting the First Impressions Dataset," in *Winter Conference on Applications of Computer Vision Workshops (WACV)*, (Waikola, HI, USA), pp. 13–21, IEEE, 2021.

[144] S. Yan, D. Huang, and M. Soleymani, "Mitigating Biases in Multimodal Personality Assessment," in *Proceedings of the International Conference on Multimodal Interaction (ICMI)*, (Online), pp. 361–369, Association for Computing Machinery, 2020.

[145] M. M. Amin and B. W. Schuller, "Normalise for Fairness: A Simple Normalisation Technique for Fairness in Regression Machine Learning Problems," *arXiv preprint arXiv:2202.00993*, 2022.

[146] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A Survey on Bias and Fairness in Machine Learning," *ACM Computing Surveys*, 2021.

[147] A. Chouldechova and A. Roth, "A Snapshot of the Frontiers of Fairness in Machine Learning," *Communications of the ACM*, pp. 82–89, 2020.

[148] E. Cambria, X. Zhang, R. Mao, M. Chen, and K. Kwok, "SenticNet 8: Fusing Emotion AI and Commonsense AI for Interpretable, Trustworthy, and Explainable Affective Computing," in *International Conference on Human-Computer Interaction (HCI)*, (Washington, DC, USA), Springer, 2024.