Final Year Project Report

Large Laughter Models: Humour & Large Language Models

Adityaa Gupta

Student ID: 20324976

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science

Supervisor: Professor Tony Veale

UCD School of Computer Science

University College Dublin

May 1, 2024

Abstract

Generative AI being the latest progression in this new and rising era of widespread AI-based applications that have seamlessly integrated into our lives, academically, professionally and also recreationally, Large Language Models (LLMs) exhibit a notable linguistic prowess in engaging with its larger human audience, with a notable mention being tools such as GPT (Generative Pre-trained Transformer). However, a conspicuous shortfall persists in their capacity to authentically replicate an essential human characteristic; humour; LLMs are fun yet not funny, they lack emotional intelligence, common sense, as well as a fundamental creative spark to be able to discern the different linguistic nuances in basic joke compositions and structures, as well as the hidden layers beneath the preface of explicit content to recreate a joke to the same effect. This project aims to explore this quality of humour generation alongside the limitations that current LLMs face in the field of humour synthesis from a computational viewpoint

Acknowledgements

This project was aided by my supervisor, Prof. Tony Veale. I would like to extend him my most profound thanks, not only for his continued guidance and patience but also for his continuous suggestions and feedback on this project that helped not only guide me on the right track but also for helping to shape the project. The database as well as the preliminary papers he provided were an indispensable resource to this project.

I would also like to thank the masses of respondents, my friends and family for helping me form the analysis for the jokes generated and surveyed for this project, the sheer number of respondents made it possible to have such a meaningful analysis.

Finally, I'd want to express my gratitude to the university for providing me with the chance to conduct my thesis on this topic.

Table of Contents

Chapter 1: Project Specification

Problem Statement:

Large Language Models (LLMs) have demonstrated impressive performance in applications such as ChatGPT. They are fluent, divergent and highly productive in the generation of novel and coherent texts, and can adapt to different genres, styles, registers and personalities with ease. But as a recent paper has noted, "ChatGPT is fun but it is not funny." [1] they are facile in the generation of witty responses that play with words and idioms, and seemingly have an answer for everything, yet they still fall short in generating what we would call "real jokes." Rather, it appears that the LLM has been over-trained on a small selection of jokes, and is unable to generate new ones of its own that have the punchiness, incongruity and humour of the real thing.

Drawing inspiration from the rich tapestry of research in the psychology of humour, a field steeped in historical scholarship and inquiry, the study will explore different interventions ranging from prompt engineering to chain-of-thought guidance and fine-tuning of an LLM. The objectives are to characterise the model's baseline capacity for humour, to identify (and quantify) its weakest and strongest suits, and to design interventions that build on this baseline.

The primary objective of this study is not only to meticulously characterise the baseline humour capacity of LLMs, and discern their strengths and weaknesses in the domain of humour generation, but also to review novel interventions for advancements that can best enable these models to be elevated to a point where they can accurately reflect the intricacies of human humour. This involves not only identifying these facets but quantifying them in a tangible way in order to provide a comprehensive understanding of the model's humour capabilities. The interventions developed in the course of this project will be based on both empirical findings and theoretical frameworks derived from the psychology of humour.

General Information:

Project framework can be found on the following Gitlab Code Repository:
https://csgitlab.ucd.ie/adwgupta/LargeLaughterModel

The forms used for evaluation of models can be found using the following links:
Survey Form1:
https://forms.office.com/Pages/DesignPageV2.aspx?subpage=designFormId=icUOQmao0EqaV-YEng07wIXTZFxHYlxPnYscdi5GuTpUQkJGRDhBVU4zWlJUTUNHUjdOQkpBOVJaSS4u Token=4937d63827344750b271adef33c601e5 [2]

Survey Form 2:
https://forms.office.com/Pages/DesignPageV2.aspx?subpage=designFormId=icUOQmao0EqaV-YEng07wIXTZFxHYlxPnYscdi5GuTpUN1VYOUI5SFlVTEdLRTVRSDBDR0xVT0M2OS4u Token=5740787a1cd34fb1b4ccc971397288c9 [3]

Aims:

Core Aims

1. Background reading
2. Implementation of LLM to learn the composition of a joke
3. Train LLM to correctly group similar jokes together

Advanced

1. Fine-tune GPT LLM to produce sufficient dataset of jokes
2. Compare the quality of the produced joke set to the training data & samples from similar prompt engineering methods.
3. Extend the application to include additional evaluation metrics and algorithms

Chapter 2: Introduction

Historically, Artificial Intelligence has been made popular via fictitious AI models, the likes of JARVIS (Marvel), or Cortana (Video Game Series), these sophisticated models are extremely popular due to their humorous and often snarky personalities as their comedic capabilities are on par with their human counterparts.

In today's digital landscape, real AI systems, commonly in the form of Large Language Models (LLMs), have begun catching up as they play an increasingly integral role in our daily interactions, serving academia, professional needs, and recreational purposes alike. AI-driven applications have begun to spread everywhere from a simple holiday booking recommender systems, photo and video editing tools such as generative fill, to more complex models such as medical imaging analysis, with models being able to detect anomalies within X-rays, MRIs, and CT scans with high accuracy.

As more and more people realise the importance and power of AI systems, the exploration for designing AI to mimic human traits have become an active topic in the field of the artificial intelligence community over the past decade. One of these key traits is humour; we as humans effortlessly weave humour into their conversations, drawing on cultural references, linguistic subtleties, and a deep well of creativity [4]. However, existing AI systems, commonly in the form of Large Language Models (LLMs), find it difficult to replicate this intrinsically human character because they rely mostly on clear patterns and prepared datasets.

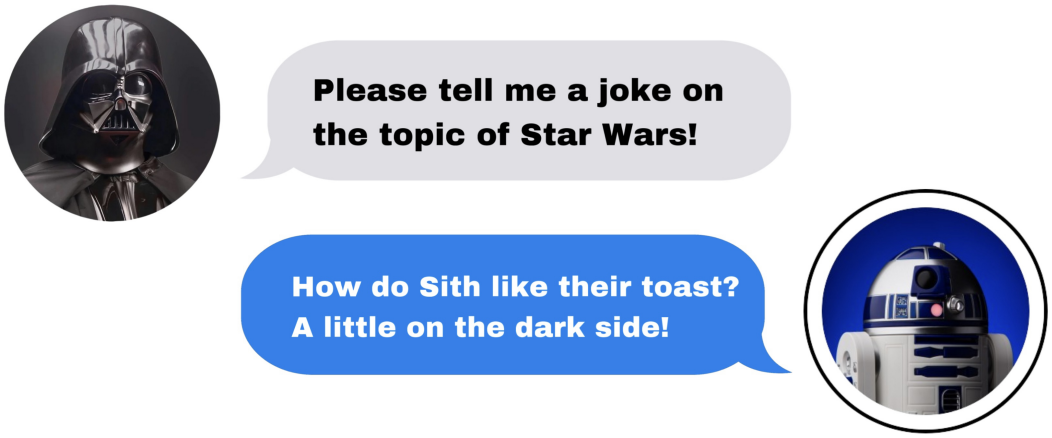

Figure 2.1: Exemplary dialogue between an LLM and a human user. [1]

This has given rise to a niche field within the subject of AI, with case studies and research being led into the intricacies of infusing LLMs, ChatGPT (Generative Pre-trained Transformer)[1]or Phind[2], with humour as well as thoroughly investigating the observation that these LLMs are not very good at creating original, comedic jokes, [1], but rather they are simply programmed to be that way [5]. Therein lies a fascinating challenge; a rigorous and scientific evaluation procedure is necessary to be able to investigate, analyse and quantify this component of humour development, particularly in light of recent research, conducted by Goes et al, which demonstrates that, despite these model's limitations at genuine comedic synthesis, they are highly efficient joke assessors, [6] and can be trained to rank and judge jokes [7].

[1] https://openai.com/blog/chatgpt
[2] https://www.phind.com/about

Chapter 3: Background Research

This chapter will focus on the background and related work. The first section will discuss the underlying core concepts of humour analysis, with subsequent sections used to analyse the conducted literature review of the primary related studies.

3.1 Humour

Laughter is the best medicine, an idea that is believed to have originated from Proverbs 17:22 [1], at its heart its an idea that is centred around one of our best abilities as humans, the ability to generate humour can be used as a 'cure-all' to many problems. As investigated by Kanuck [4], humour is one of the key aspects to our personalities, our every day lives and even our socialism, and as such, any form of artificial intelligence we create, in the hopes of imitating human intelligence, must have a baseline to replicate our capacity for humour.

LLMs approach humour computationally [8] rather than linguistically, their analysis involves a complex interplay of cognitive processes, including pattern recognition, social cognition, and emotional processing. This interplay is what allows humour to be both entertaining and insightful, as it often challenges our preconceived notions and encourages us to think in new ways. This computational approach to humour is an interdisciplinary field that intersects with NLPs and humour theory [9].

Humour theory explores what makes something funny, including incongruity, superiority, and relief theories [9]. humour is complex, involving linguistic nuances like wordplay, puns, sarcasm, and cultural idioms, and is highly context-dependent. It's a universal yet subjective and culturally nuanced form of communication, making it challenging for AI to grasp.

Applying humour theory to Large Language Models (LLMs) involves assessing their ability to understand, generate, and respond to humour. This includes testing their cognitive sophistication, interpretation, and ability to continue in a humorous vein. humour in LLMs can be categorised into style humour, cultural reference humour, situation humour, and interpersonal violation humour. However, humour's subjectivity and cultural dependency pose challenges in establishing a universal standard for evaluating humour across different LLMs. Additionally, LLMs might inadvertently cross the line between being amusing and being offensive due to their lack of understanding of human emotions and social nuances [10]

To effectively apply humour theory to LLMs, it's crucial to develop a nuanced understanding of humour and its cultural contexts, as well as continually updating the model to adapt to changing societal norms and humour trends [11]. Proficiency in humour is not just an added feature but a significant indicator of an LLM's overall ability to comprehend and replicate human-like communication, reflecting the model's capacity to grasp the finer aspects of human communication, often unspoken and laden with emotional and cultural significance. Therefore in order to tackle the inability for LLMs to be comedic, we must analyze humour computationally instead of linguistically.

[1] https://www.bible.com/bible/59/pro.17.22

3.1.1 Joke Structure

A joke, at its core, is a form of communication that leverages language and context to evoke laughter or amusement. The essence of humour lies in its ability to play with words, ideas, and social norms in a way that is unexpected and often absurd. [4] This playfulness is what makes a good joke resonate with audiences, as it taps into shared experiences and cultural references that are universally relatable. The effectiveness of a joke is not just about the cleverness of the punchline but also about how well it integrates with the setup and the cultural context in which it is presented. This integration is what makes a good joke not just funny but also meaningful, as it often reflects on societal norms, human behaviours, or the absurdities of life. [12]

In its essence a joke can be split into two parts; the Setup and its Punchline.

The setup, often either a simple question or a introductory statement establishes the context, luring the audience into a familiar or relatable scenario. It sets an initial context for expectations and creates a foundation of curiosity for what the audience can predict the punchline to be as they draw upon personal experience and their own sense of humour.

The punchline complimenting the setup, serves to satisfy the curiosity created by the setup, delivering an unexpected twist or surprise, subverting the audience's expectations and generating laughter either through comedic irony or a comically "bad" punchline. This is most often done using subtext, a keyword or phrase, that connects the set-up to the punchline and vice versa, without this 'connector' [8], the joke itself would fall apart and go over the heads of the audience.

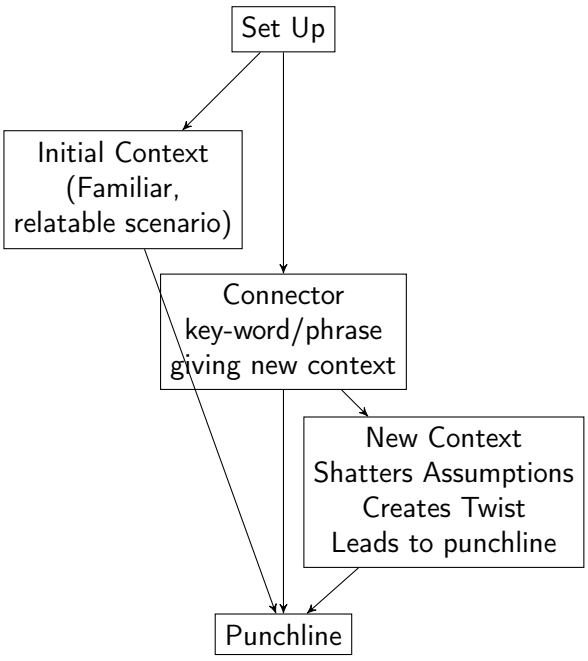

Figure 3.1: Joke Composition

This composition be considered as a sequence of steps, starting from the set-up and taking the audience through a story with a twist and turn that ultimately leads to a shattering conclusion in the form of a punchline. In the context of LLMs, understanding and replicating this setup-and-punchline dynamic is crucial. While existing models may comprehend linguistic patterns, capturing the essence of surprise and novelty inherent in punchlines remains a challenge.

The goal is to train LLMs to recognise patterns leading to an expected outcome and then skilfully deviate from those patterns to create humorous and unexpected twists. An example of what we expect a Large Laughter Model to produce following this ideology would be as follows:

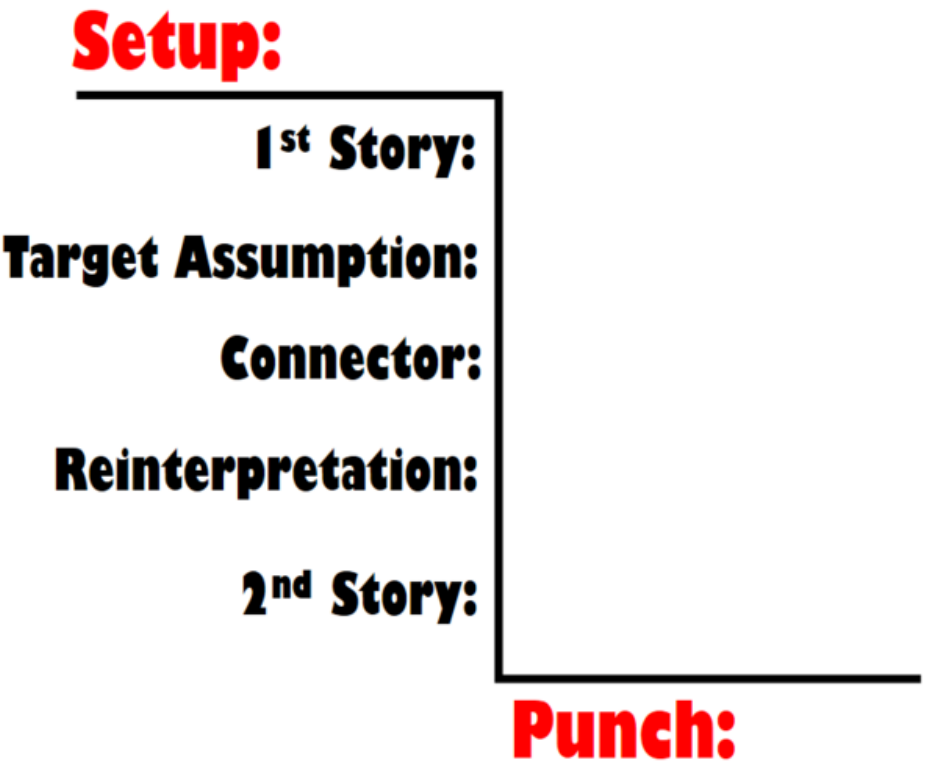

Figure 3.2: Joke Composition to Teach a Large Laughter Model [13]

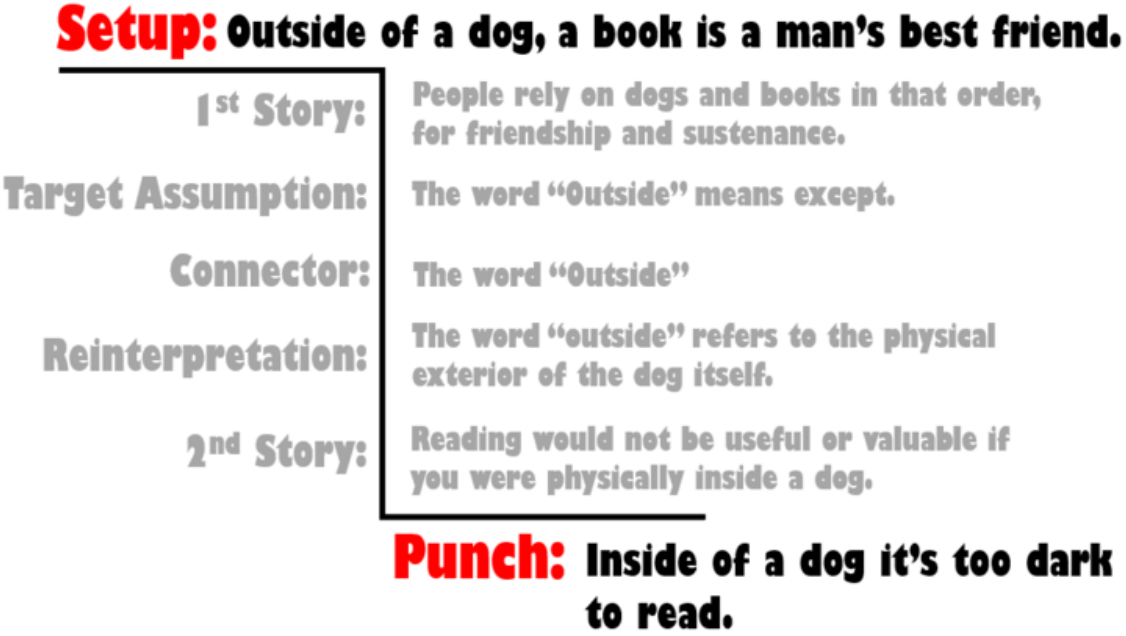

Figure 3.3: Example of a produced joke [13]

As shown above, the ideal way for the LLM to produce the joke would be to take the set-up, in this case, "Outside of a dog, a book is a man's best friend." and start to predict what the 1st story

could be (Reliance on dogs for companionship, etc), and therefore make its target assumption (outside = except -> twist incoming).

Once this has been executed, the LLM can use the connector word 'outside' to reinterpret the joke now that it classified and understood the twist and where it is coming from, break down the joke and analyse it until the LLM logically concludes that as the subject is a book and a dog, with the connector being outside, it must be along the lines of "being physically inside a dog wouldn't be of much use for reading" and finally producing a punchline similar to "Inside of a dog, it's too dark to read!" [2]

3.2 Insights from Toplyn

This section aims to delve deeper into the structure of jokes, examining the role of linguistic elements, timing, and context in creating humour through Joe Toplyn's work[3], a well-known professional comedian and writer, known for his work. He has contributed significantly to the understanding and application of LLMs in various domains such as humour through a series of novel patents; systems for fine-tuning and prompt engineering Large Language Models to produce better comedic results.

Joe Toplyn's patents [14–16] have greatly contributed to the field of humour analysis and they offer valuable insights into the mechanics of joke composition. These insights are greatly helpful to provide a fresh perspective for advancing the capabilities of LLMs to better understand and replicate the structure of jokes. The focus is on exploring the humour beyond the surface, aiming to uncover the underlying layers that contribute to the creation of original, humorous content. Toplyn's work on prompt engineering and model fine-tuning has played a significant role in enhancing the creativity and effectiveness of AI-generated humour. Research by Chen, Shi, and Si [17] supports this, demonstrating that by carefully crafting prompts and fine-tuning the AI model, it's possible to produce humour that is both amusing and resonates with human audiences.

Toplyn's algorithm for writing jokes is based on a practitioner's understanding of the structure and mechanics of humour. The algorithm leverages the AI's ability to understand and mimic the nuances of human language, including the use of irony, exaggeration, and wordplay, to create humour. [12]. It involves the use of prompts that guide the AI towards generating content that is both funny and contextually relevant. [18].

1. Topic Sentence Creation: The process begins with selecting a topic, often based on a news item or current event. The goal is to create a topic sentence that is engaging, factually accurate, and not inherently funny, setting the stage for humorous commentary.
2. Identifying Handles and Associations: Within the topic sentence, handles are identified. These are interesting or peculiar words or phrases that can serve as the foundation for developing the joke. Following this, associations are created for each handle. Associations are concepts or ideas related to the handles, which will be used to develop the punchline.
3. Developing the Punchline: The punchline is crafted by combining associations from different handles in a way that creates a surprising connection. The punchline should evoke a negative emotion towards the first major entity in the topic, which is a key element in generating humour according to Toplyn's theory.

[2] https://gregdeancomedy.com/standup-comedy-glossary/joke-structure-by-greg-dean/
[3] https://joetoplyn.com/

4. Generating the Angle: The angle is a bridge between the topic and the punchline, ensuring a smooth transition that maintains the flow of the joke. It's designed to connect the topic to the punchline in a natural-sounding way, enhancing the overall humor effect.

5. Feedback and Iteration: The algorithm is capable of refining its approach based on feedback. If a generated joke is not funny or does not fit the context, the AI can learn from this feedback and adjust its future outputs accordingly, allowing for continuous improvement in humour generation.

By following these steps, the algorithm can produce jokes that resonate with human audiences, demonstrating the potential of AI in mimicking the nuances of human humour.

In addition to Toplyn's algorithm, Witscript,[4] is a model inspired by Toplyn's work that has been developed to specifically focus on humour generation. [19]. Witscript leverages Toplyn's algorithm and prompt engineering techniques to generate humour that is both funny and culturally relevant. The model's success in humour generation is attributed to its ability to understand and mimic the nuances of human language, including the use of irony, exaggeration, and wordplay, as outlined in Toplyn's patents. [20]. Through systems like Witscript as well as his patents for both joke generation and recognition [21–23], shedding light on the importance of dissecting and comprehending the fundamental elements that make a joke effective. The analysis of joke structure according to Toplyn involves dissecting the components that contribute to the humour quotient.

3.2.1 Wordplay and Linguistic Nuances

The complexity of humour synthesis in LLMs is significantly influenced by the nuanced aspects of language, including wordplay and linguistic subtleties. These elements are crucial for crafting effective jokes, as they allow for multiple layers of meaning and interpretation. [24] However, these models often struggle with these intricate linguistic plays, leading to literal or incoherent interpretations of jokes that fail to capture their comedic essence [24].

Toplyn's work [16] highlights the importance of wordplay and linguistic nuances in the crafting of effective jokes. Puns, double entendres, and clever use of language contribute significantly to the wit and humour of a joke.[25] However, LLMs, including ChatGPT, often fail to replicate the intricate play on words that characterises human humour. These models often interpret jokes literally, missing the double entendres that is crucial for maintaining the comedic effect of a joke.

To address this limitation, interventions aimed at improving LLMs' humour synthesis should focus on incorporating linguistic subtleties, cultural references, and wordplay into the models' training data from a wide range of datasets to avoid over-training the model on a very specific [26]. This approach would help the models to better understand and generate language-based humour, allowing them to weave clever and contextually relevant jokes. Interventions aimed at improving LLMs' humour synthesis should address the nuanced aspects of language. [12]. This involves incorporating linguistic subtleties, cultural references, and wordplay into the models' training data. Additionally, fine-tuning processes can focus on enhancing the understanding and generation of language-based humour, allowing LLMs to weave clever and contextually relevant jokes.

Toplyn's system [21] for recognising and identifying wordplay in jokes is a significant step towards enhancing LLMs' humour synthesis capabilities. In this system, an LLM would theoretically break down a joke's setup into its basic words and identify tuples that share similarities, such as same or similar beginnings, lemmas, semantics, etc. The system would then continue its logical analysis to determine all possible meanings behind the setup, both on a surface and subtext level. This

[4] https://witscript.com/

process would enable the LLM to understand the nuanced aspects of language that contribute to the humour in a joke, thereby improving its ability to generate and understand jokes.

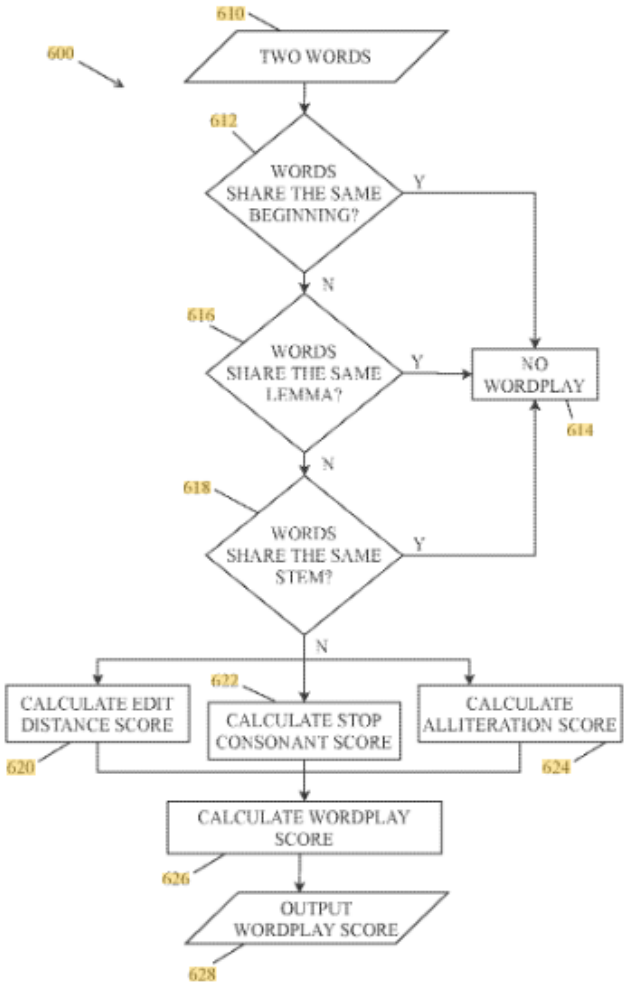

Figure 3.4: Toplyn's system for ranking wordplay within jokes [27]

As illustrated in the figure, the architecture for Toplyn's system for recognising and identifying wordplay showcases the computational thinking process as it systematically breaks down the set-up into its basic word, identifying similarity tuples (similar lemmas, semantics and/or meaning, etc), and continues its logical analysis determining the wordplay behind the joke, allowing the model to calculate a "score" for the joke based on how simple/complex present wordplay in the joke equates to. This showcases the potential of such a system to enhance the humour synthesis capabilities of LLMs. By incorporating linguistic subtleties and cultural references into the training data and focusing on enhancing

3.2.2 Timing and Delivery

Effective comedic timing and delivery, the nuanced interplay between the setup and punchline, as well as the rhythm and flow of language, plays a paramount role in the impact of a joke. Toplyn's insights underscore the critical importance of not only what is said but also how it is said, highlighting the multifaceted nature of humour [14]

For Large Language Models (LLMs), mastering the art of timing and delivery transcends mere text generation. It demands a deep understanding of the rhythm and flow of language, which

are essential for comedic timing [28]. This understanding is crucial for fine-tuning interventions that aim to imbue LLMs with a sense of timing, enabling them to deliver punchlines with the appropriate cadence and emphasis [29].

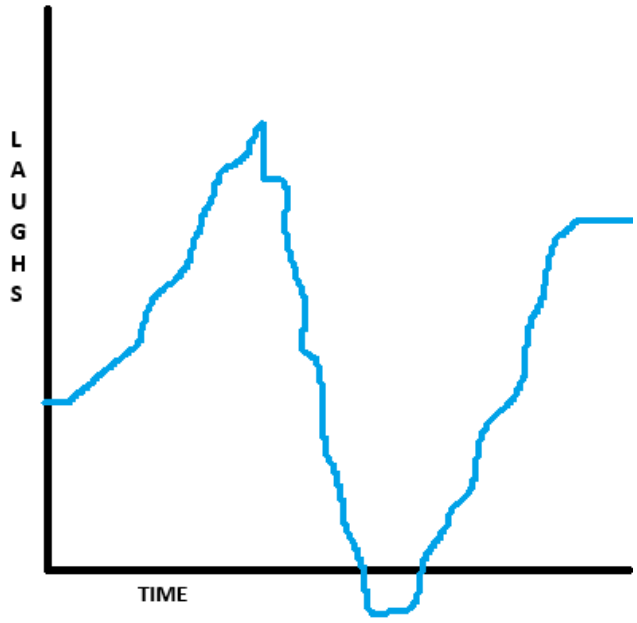

Figure 3.5: Exemplary "graph"/joke illustrating comedic timing & delivery.[30]

As an example, the graph above is a bit presented by a stand-up comedian, told line, who asks if the audience has ever heard a funny joke about a graph and that the comedian was going to do one. Now as there is nothing inherently comedic on the subject of graphs given no context, the bit itself played on comedic timing as well as irony which is what made the joke funny, with the line being drawn as the audience laughs in response to the joke and the realisation of the joke itself, causing a dip or surge in the laughs received, with the comedian almost leading the audience to awkwardly laugh along as he rose the line showing positive values indicating "positive" values (laughter), along with negative scores on the laughter axis indicating when the audience showed little interaction (laughter) showcasing his joke "dying out" and "rising" back up. The delivery itself is what turned the joke from a "dead" one to a funny one where the audience laughed as the comedian prompted them to.

This is a prime example of how a simple joke which isn't inherently funny nor is a particularly good joke can be made funny simply by being able to connect with the audience through tone, emotions and a shared experience through its delivery.

Unlike humans, along with their inability to have a verbal conversation to convey tone, LLMs lack personal experiences, emotions, and an inherent sense of humour; making it challenging for them to fully grasp and predict what humans find funny. humour often relying on wordplay, puns, or subtle nuances that can be difficult for AI to identify and interpret correctly. [24] Moreover, the nuances of comedic timing and delivery, which are crucial for making a joke funny, are beyond the capabilities of text-based AI [26]

Given these constraints, the focus for LLMs should shift towards enhancing their delivery rather than attempting to create comedic timing. [31] This involves making the joke itself punchy and self-explanatory, avoiding verbose explanations that might detract from the humour. By focusing on brevity and punchiness, LLMs can craft jokes that are concise and catchy, making them more memorable and effective rather than ones which are verbose, and therefore losing their comedic

spark the further a reader is forced to read and think about said joke [25].

3.3 Large Language Models

In the realm of artificial intelligence, Large Language Models (LLMs) have emerged as transformative entities, playing a pivotal role across various facets of our daily lives. Their versatility and adaptability make them indispensable in diverse applications, contributing significantly to the ever-evolving landscape of AI-driven technologies, with their prowess lying within in their adeptness at comprehending/producing text with fluency, coherence, and adaptability across various linguistic styles, genres, and personalities. [32]

LLMs stand at the forefront of natural language processing, boasting an unparalleled ability to process and generate human-like text on a vast scale having been trained on vast amounts of textual data. They represent a significant advancement in artificial intelligence and natural language processing, opening up new possibilities for human-machine interaction and automation, aiding to streamline processes and enhancing productivity with their capacity to understand complex concepts and articulate them in a manner akin to human discourse.

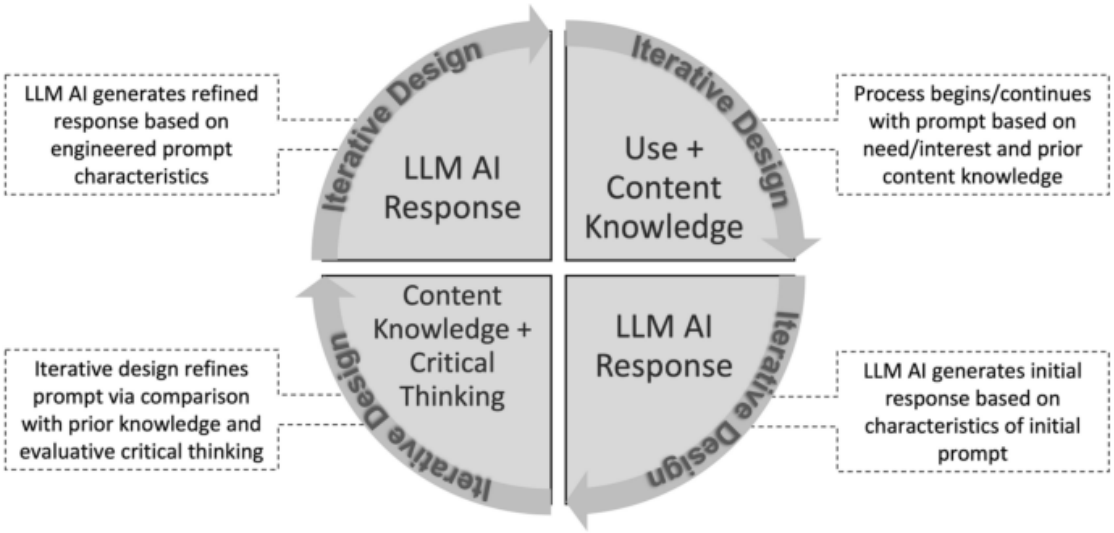

Figure 3.6: Basic LLM Design Overview [33]

The training of LLMs is a significant factor in their ability to perform tasks requiring pattern recognition, especially in the field of content creation. This process begins with feeding large amounts of text to a neural network, which is trained on various internet archives such as Wikipedia, scholarly essays, Reddit, and digitised books. The text is then translated into a format that machines can analyze and manipulate, a process known as Natural Language Processing (NLP). A crucial part of NLP is tokenisation, which assigns numerical values to the frequency of certain text, allowing the models to understand and generate the next token in a sequence of tokens [34].

For example, Turano and Strapparava's work introduces the SCRIPTS dataset [28], sourced from stand-up comedy shows, to emphasise the importance of understanding humour within a broader narrative context. This shift in perspective challenges the conventional approach of training LLMs solely on standalone jokes and prompts a reconsideration of the scope of humour synthesis through a well defined and structured dataset.

3.3.1 ChatGPT

GPT [5] (Generative Pre-trained Transformer), is a flagship LLM Model developed by OpenAI, has become a pivotal figure in driving AI-driven applications [32]. It has demonstrated remarkable capabilities in analysing and generating human-like text with its performance in code generation, mathematical deduction and performance on standardised exams compared to human efforts being msot noteworthy. A study by Joshi et al, investigated the usage of the chatbot by university students [35], where the primary conclusion was that the majority of the surveyed students main purposes was generating and debugging code, brainstorming on new ideas, learning new concepts, getting feedback on their own solutions as well and creating new content such as reports and emails.

However, the study also noted that there were notes of concern among the students on the model's reliability and accuracy, particularly among more creative tasks, such as brainstorming ideas or creative writing. LLMs are trained to recognise, translate, predict, and generate speech, not to feel emotions, [36] with studies done to analyse ChatGPT failing to identify utterances of humour [37]

This has been further investigated by Gómez-Rodríguez and Williams [38] where their investigation provided insights into this limitation of humour generation with the the claim that the popular LLM such as GPT's creative output is largely constrained by the data it has been trained on, leading to a lack of originality in its storytelling and joke generation as it struggles to provide genuine original content.

3.3.2 Issues and Challenges with ChatGPT

As powerful and ever-changing as these models such as GPT are, one of the main pitfalls of LLMs, particularly in the context of humour generation, is their struggle to mimic human capabilities, such as originality, effectively.

Humour, being a complex and subjective form of communication, often relies on cultural context, personal experiences, and nuanced understanding that current LLMs find challenging to replicate. This distinction is further supported by Holl [39], who argues that while LLMs are intelligent, they should be considered narrow AI (weak AI), designed for specific utility functions rather than possessing the general skills applicable to a broad variety of utility functions nor the broader understanding and adaptability required for general AI (strong AI) applications.

On this topic the biggest concern regarding the limitations of LLMs, are that "while fun, they are not funny" [1], as investigated by Jentzsch and Kersting. Their investigation found that while GPT can correctly identify, reproduce, and explain puns that fit into a learned pattern, it fails to meet puns of other kinds, resulting in a limited reflection of humour. They came to the conclusion that the model also struggles to confidently create intentionally funny **original** content, as it kept repeating the same 25 jokes from its pre-trained dataset [40].

They also found that GPT started to develop a very particular approach in its approach to learn humour from its training data and started learning a specific style of "humour", and while Jentzsch and Kersting did propose that this may possibly be because of GPT finding that style of humour funny over others, there was inconclusive evidence to support this theory, with the more simpler hypothesis being that GPT simply wasn't adapting the different nuances in humour correctly as it was trying to adapt a computational approach to humour as opposed to a lingustic one.

A recent study on the cultural understanding of LLMs to better analyse this limitation was con-

[5] https://openai.com/blog/chatgpt

ducted by Li et al, [41] with a novel approach by fine-tuning GPT among other LLMs to analyse and recognise puns in a multitude of languages with the hopes of making these LLMs develop a "sense of humour". This study was complemented by research done by Popova and Dadić [42] where in their paper, they analysed the composition of wordplay and puns and trained using fine-tuned LLMs. However the results were unsatisfactory, further cementing the claim that GPT isn't naturally funny as even out-of-box thinking wasn't able to make GPT develop a "personality" naturally using these novel approaches and similar to Jentzsch and Kersting's work.

3.3.3 Reliance on trained data

Another critical aspect to consider is the overtraining of LLMs on datasets that may not fully represent the diversity of human experiences and cultural nuances. As stated by my supervisor Tony Veale in his work, GPT isnt naturally funny programmed to be funny, it is simply programmed to act that way [5] and often fails to resonate with human audiences from a lack of cultural understanding and an inability to grasp subtleties of humour because to its insistence on learning humour in a particular way due to its overtraining and reliance on trained data. with its reliance on pre-trained datasets a critical aspect influencing its performance.

Thorp claims that this is because GPT is simply "not an author", GPT can help expand and extend on creative work but it cannot generate its own satisfactory creative pieces[43], which is further backed up research done by Koco'n et all [44] where in their investigation on the the limitations of ChatGPT across various tasks, including humour generation, they argue that the model's performance is heavily influenced by the specific datasets it was trained on. GPT has simply been "over-trained to do a large variety of tasks to a high degree" [45] but fails to simply meet expectations when prompted on niche subjects or asked to aid in specialised endeavours.

This over reliance on certain datasets, especially those that may not encompass a wide range of humour styles or cultural contexts, can lead to a narrow understanding of humour, further limiting the model's ability to generate universally relatable humour.

To tackle this issue a potential intervention could be along the lines of the work researched by Chen et al [36] where they discuss the complexities of humour generation and understanding within current LLMs as well as the limitations, primarily due to the lack of a comprehensive humour corpus and the difficulty in capturing the nuances of humour, which often relies on tacit knowledge and commonsense.

The authors created the largest Chinese Explainable humour Response Dataset, enriched with chain-of-humour and humour mind map annotations. This dataset is crucial for evaluating and enhancing LLMs' humorous response capabilities. The inclusion of chain-of-humour and humour mind map annotations is significant as it captures the structure and reasoning behind humour, often overlooked in traditional datasets.

Additionally, the paper introduces humour-related auxiliary tasks to complement the dataset. These tasks target aspects of humour that LLMs often miss, such as timing, irony, and cultural references. This novel approach aims to improve humour generation in LLms, bridging the gap between current capabilities and the complexity of human humour. The paper's findings are based on extensive evaluations on the proposed dataset and auxiliary tasks.

The results show that the dataset and tasks effectively enhance LLMs' humorous response capabilities, indicating their potential to improve humour generation. This improvement is significant as it suggests the dataset and tasks have successfully addressed humour generation challenges in LLMs, setting the stage for future research in this area.

As the above findings prove, the emphasis should remain on understanding humour not just as

standalone jokes but as a dynamic element within broader linguistic contexts, crucial for developing models that can generate humour that is both universally relatable and culturally nuanced. Therein lies the need for a more nuanced approach to training LLMs, incorporating a wider range of data sources and focusing on the integration of humour within broader linguistic and narrative contexts, offering a fresh intervention to overcome the barrier that current LLMs face due to its reliance on pre-trained data. The emphasis is on understanding humour not only as a set of standalone jokes but as a dynamic element embedded in broader linguistic contexts. We must rethink our approach to structuring the development on LLMs through new interventions [46].

3.3.4 Humour Evaluation

Yet, despite its limitations in creative tasks, interestingly enough GPT's potential in evaluating capabilities is also noteworthy. The study by Goes et al (2022), [7] introduces a novel method for critiquing and evaluating jokes using trained, fine-tuned LLMs as "judges", showcasing the LLMs, in particular ChatGPT's, potential in this niche area.

Going one step further to extend their previous work, Goes et al (2023) [6] specialised their study on the performance and capability of GPT's ability to assess the quality of jokes, demonstrating that the model can indeed provide valuable insights into humour, such as what makes a particular joke funny, by analysing its core elements. This work demonstrates that GPT models can indeed provide valuable insights into humour, such as what makes a particular joke funny, by analysing its core elements.

To support this claim, Gomez-Rodriguez et al [38], provides additional evidence on the subject matter. Together, they evaluated a range of recent LLMs on English creative writing, a challenging task that requires imagination, coherence, and style to specifically address the evaluation of humour in creative writing, showing that some state-of-the-art commercial LLMs match or slightly outperform human writers in most dimensions, including humour. This study provides evidence that LLMs have the potential to understand and evaluate humour. This train of thought gives rise to the question; are we simply training Large Laughter Models, such as GPT, incorrectly?

Furthermore, the study [47] done by Eller, Fulda, and Seppi examines the use of humour prediction datasets in humour generation tasks. The research focuses on the most popular dataset included in the 2020 SemEval's Task 7 and teaches a model to translate normal text into humorous text. The model's performance was evaluated against humorous human-generated headlines, showing that it was preferred equally in A/B testing with the human-edited versions and was significantly better than a random baseline, indicating that humour prediction datasets indeed provide potential for future humour generation and evaluation systems.

The ability of LLMs to evaluate jokes raises intriguing questions about the underlying mechanisms that enable them to discern humour. Surely, if an LLM can evaluate the quality of a joke, then it should be logical request for the model to be able to not only imitate the joke but use it as a source of inspiration to create a joke it hasn't been exposed to yet - a novel, self generated comedic joke.

Summary

Humour, being a deeply subjective and culturally dependent form of communication, presents a unique challenge for LLMs. It requires a nuanced understanding of context, timing, and societal norms, which are often difficult to capture and replicate through training on large datasets alone.

The variability in humour across different cultures, languages, and contexts necessitates the use of diverse and representative datasets for training. However, simply exposing LLMs to a wide array of humorous content is not sufficient, as humour often relies on subtle, layered, and context-dependent elements that are challenging for AI to grasp and replicate.

As highlighted within this literature review, to overcome these comedic barriers, novel interventions must leverage a comprehensive and multi-faceted dataset to prevent overfitting. Incorporating Toplyn's insights into the training process is crucial for enhancing humour synthesis capabilities. This includes recognising the importance of setup and punchline dynamics, linguistic nuances, timing, and relatability, developing a comprehensive framework for developing more authentic and engaging jokes.

To address these challenges, interventions should focus on several key areas:

- **Diverse and Representative Datasets:** Training LLMs on datasets that capture the broad spectrum of humour across different cultures, languages, and contexts is essential. This includes not only humorous content but also annotated datasets that provide information about the structure, content, and context of humour, aiding the AI in recognising similar patterns in new data. [40, 45]
- **Sophisticated Machine Learning Algorithms:** The development of sophisticated machine learning algorithms capable of learning from examples and making predictions based on learned patterns is crucial. These algorithms should be designed to understand the nuances of humour, including wordplay, cultural references, and unexpected narrative twists. [8, 48]
- **Continuous Learning and Adaptation:** Given the evolving nature of humour, LLMs must be capable of continuous learning and adaptation to remain relevant and sensitive to current social contexts. This involves updating the models with new data and refining their understanding of humour as societal perceptions shift. [49]
- **Cultural and Contextual Awareness:** LLMs need to be trained to recognise and apply various forms of humour, such as puns, satire, or irony, which require different levels of linguistic and cultural awareness. This involves understanding the complex interplay between language and human social dynamics. [4]
- **Ethical Considerations:** Given the potential for humour to cross the line between being amusing and being offensive, it's important to implement ethical guidelines and monitoring mechanisms to ensure that LLMs generate humour that is appropriate and respectful, while still allowing for creative wordplay and suggestive/edgier content.

Chapter 4: Design Architecture

This chapter covers a detailed specification of the project structure, and the design used as well as a discussion of alternative methods, and interventions possible for infusing an LLM with humour. and then presents the implementation as well as a methodology for analysing the models' performances and results.

Figure 4.1 shows a high-level diagram of the architecture behind the project's Large Laughter Model, showing a breakdown of the system, indicating three key phases in the development; training, collecting data and analysing.

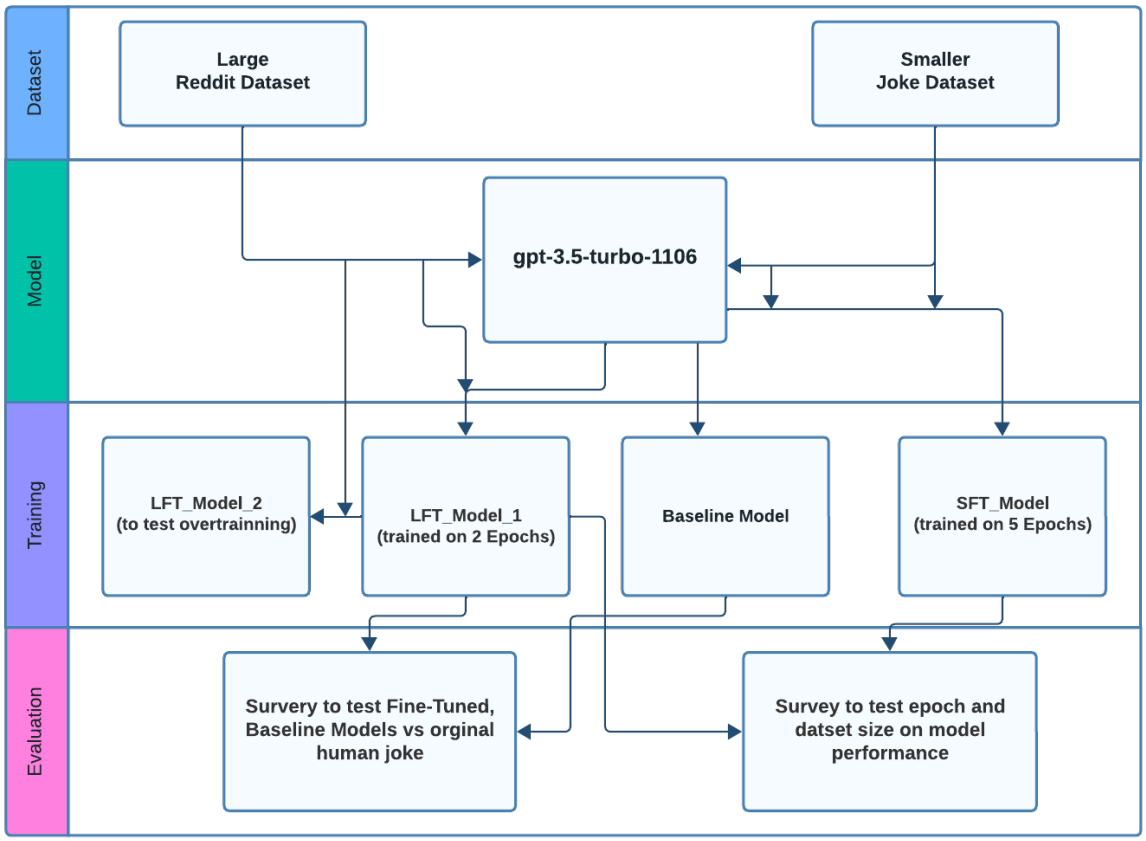

Figure 4.1: Project Design Architecture

4.1 Dataset

As we are attempting to teach a pre-trained model what a good joke looks like, it is imperative that we use real world examples, and as such, the dataset chosen is a cleaned and filtered dataset taken from Reddit jokes, which have been split into their respective setups and punchlines

However as the examples within our dataset are real world examples and many of the jokes involve

certain words or phrases which appeal to certain themes of humour which very may offend different cohorts of the greater population differently, there merits a thorough filtration process which allows only appropriate jokes to be exposed to the LLM training phase while still retaining more edgier and funny content, allowing genuine creativity to be shown from the fine-tuned model.

4.2 Data Storage and Evaluation

Analysing the jokes from the model must be stored in a sufficient dataset to allow evaluation to be performed, both quantitative and qualitative. Therefore Excel spreadsheets will be used as a method of easy access as well as visual inspection.

Once data has been collected, quantitative evaluation (statistical tests, averages, evaluation metrics such as Coherency, Profanity and Verbosity) as well as qualitative evaluation (human feedback, statistical tests on model performance) must be conducted in a vigirious manner.

4.3 LLM Choice

The core of our approach lies in utilising the OpenAI GPT-3.5 architecture as the Large Language Model (LLM) for our project. Currently the following models are offered by OpenAI for fine-tuning purposes,

- gpt-3.5-turbo-0125
- gpt-3.5-turbo-1106
- gpt-3.5-turbo-0613
- babbage-002
- davinci-002
- gpt-4-0613

Gpt-4-0613 would be the obvious choice, being less over-trained, more recent and the more efficient model, however at current times this model is experimental and not open to the general public for fine-tuning. Therefore, gpt-3.5-turbo-0125 is chosen as OpenAI recommends this model above the older models for its state-of-the-art natural language processing capabilities and its ability to generate coherent and contextually relevant text, especially in terms of results and ease of use.

Now the choice remains in choosing our method of intervention to obtain a Large Laughter Model; fine-tuning or prompt engineering?

After having done a review over a substantial amount of case studies and research within the literature review in Section 3 on the topic there seems to be one pressing decision ahead on having to settle on a specific method of intervention to infuse humour into a LLM to create our Large Laughter Model; Fine-tuning and/or Prompt Engineering?

4.4 Intervention 1: Prompt Engineering

4.4.1 Overview

The literature on prompt engineering, exemplified by studies like Wang et al.'s evaluation in code intelligence tasks [50], introduces an alternative approach to fine-tuning. Prompt engineering, as demonstrated in natural language processing tasks, shows promising results in providing task-specific knowledge to pre-trained models [51]. The focus is on understanding how prompts can serve as effective tools to guide LLMs in generating more contextually relevant and humorous responses [17]. There are many different facets to the manner in which an LLM can be prompted as shown in the figure below.

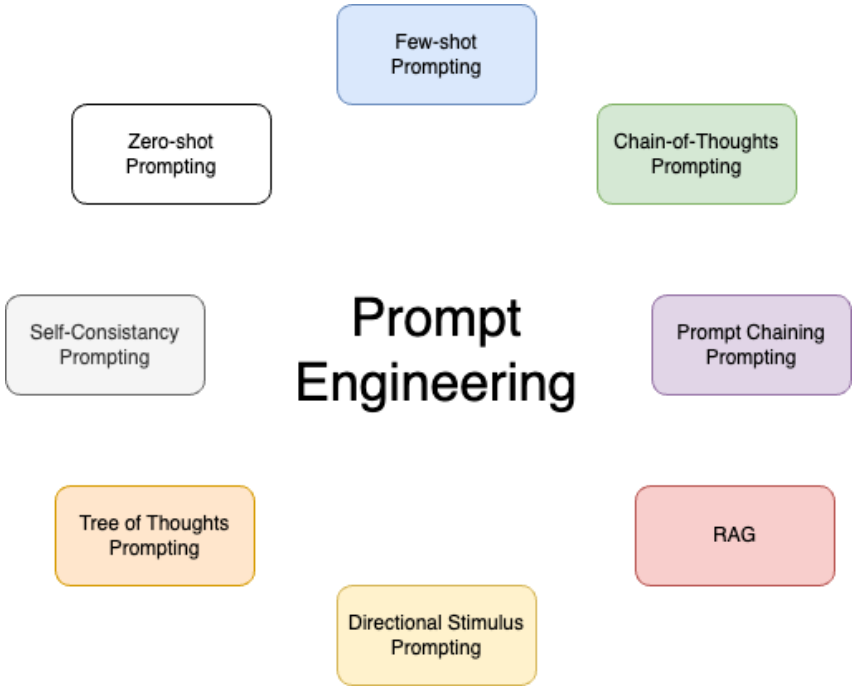

Figure 4.2: Nuances of Prompt Engineering

This is especially important when it comes to models like GPT which have been over-trained on a wide variety of datasets but unable to derive desired particular responses independently through One-Shot Prompting. To get the intended effects, effective prompt engineering may entail experimenting with the prompts' phrasing, structure, and specificity as well as a give-and-forth method of communication with the LLM chatbot.

4.4.2 Prompting Process

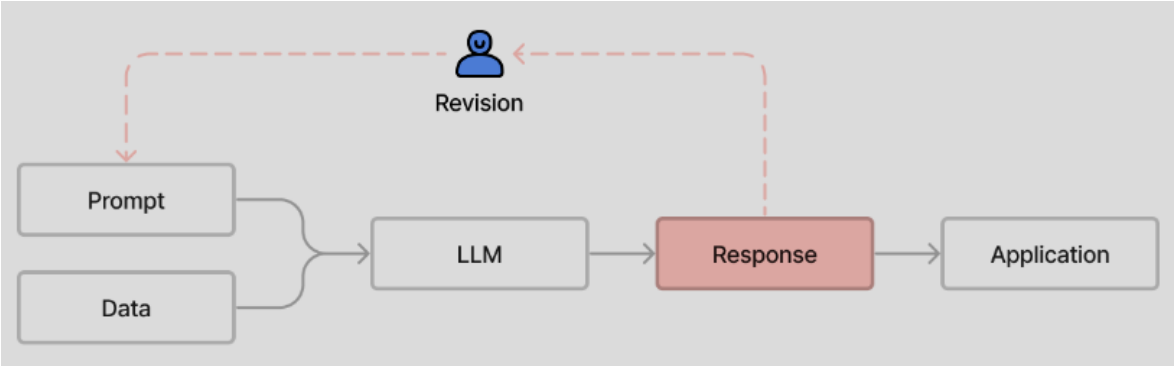

Figure 4.3: Basic Process for Prompt Engineering

As seen above in Figure 4.3, the prompting process aims to generate contextually relevant and humorous outputs, enhancing the chatbot's human-like interaction through a cyclic like sequence of steps. The process begins with the selection of a topic sentence, which sets the context for the joke. This sentence is crucial as it provides the foundation for the joke's theme. Utilising NLP techniques, the system extracts keywords from the topic sentence. These keywords are then linked with related words through wordplay to form the punchline. This creative step involves generating punch words and bridges to construct the joke.

A pre-trained neural network language model, fine-tuned on a dataset of TV show monologue jokes, is employed to complete the joke response. This model fills the gap between the topic sentence and the punchline, ensuring the generated joke is coherent and contextually relevant. An internal scoring method filters out jokes that do not meet a preset standard of quality. This step ensures that only the most humorous and relevant jokes are selected for output.

The Witscript system exemplifies a sophisticated approach to joke generation, combining expert comedy writing algorithms with advanced NLP and neural network models. This method not only enhances the chatbot's ability to generate humorous responses but also contributes to making the interaction more engaging and human-like.

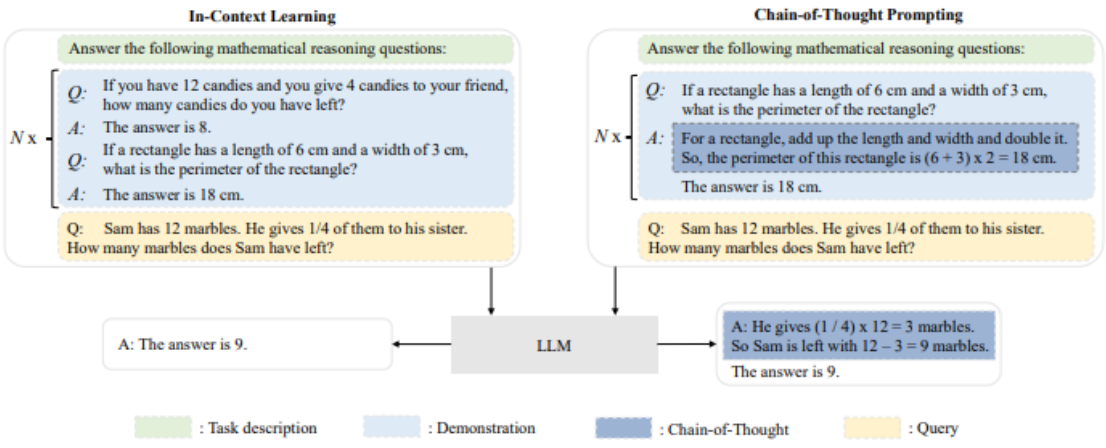

Figure 4.4: Different LLM prompting processes [52]

4.4.3 Pros of Prompt Engineering

1. **Task-Specific Guidance:** Prompt engineering allows for task-specific guidance to be provided to pre-trained language models (LLMs). This specificity can enhance model performance in particular domains or tasks, as seen in code intelligence tasks [50].

2. **Addressing Fine-Tuning Limitations:** Prompt engineering offers an alternative to traditional fine-tuning methods, addressing some of the limitations associated with fine-tuning. This includes the risk of overfitting to the fine-tuning dataset and the need for large amounts of task-specific labelled data [51].

3. **Flexibility and Adaptability:** With prompt engineering, users have more flexibility in shaping inputs to the model. This can be particularly beneficial when dealing with tasks that require nuanced or context-dependent instructions.

4. **Improved Context Sensitivity:** Prompt engineering allows researchers and users to inject context directly into the input, helping language models generate more contextually relevant outputs. This is especially valuable for tasks where context plays a crucial role, such as humour generation. [17]

5. **Knowledge Transfer:** Pre-trained models have general language understanding, and prompt engineering enables the transfer of task-specific knowledge to these models. This can lead to improved performance in specialised tasks without the need for extensive retraining, this also can wind up with the same dilemma of whether your Large Language Model is 'naturally' funny, or is it simply programmed to inmate humour. [48]

4.4.4 Cons of Prompt Engineering

1. **Expertise Requirement:** Effective prompt engineering often requires a deep understanding of both the task at hand and the underlying model architecture. This can be a barrier for users without the necessary expertise.

2. **Risk of Biases:** Crafting prompts requires careful consideration to avoid introducing biases or unintended influences on the model's output. Inappropriate or biased prompts can lead to undesirable results.

3. **Limited Generalisation:** While prompt engineering can enhance performance in specific tasks, there is a risk that models become too tailored to the provided prompts, limiting their ability to generalise to diverse inputs or tasks outside the prompt's scope.

4. **Manual Effort:** Crafting effective prompts may involve a significant manual effort, especially when dealing with complex tasks or nuanced instructions. This manual effort can be resource-intensive and time-consuming.

5. **Dynamic Task Requirements:** Prompt engineering may not be suitable for tasks with dynamic or evolving requirements, as updating prompts might be less straightforward than retraining a model through fine-tuning.

6. **Reliance on Pre-trained model:** While this intervention is more flexible and adaptable, it is entirely reliant on the Large Language Model being pre-trained well enough to generate jokes on any given prompt. An unseen prompt or joke could very well break the model or produce unsatisfactory results.

4.5 Intervention 2: Fine-Tuning

4.5.1 Overview:

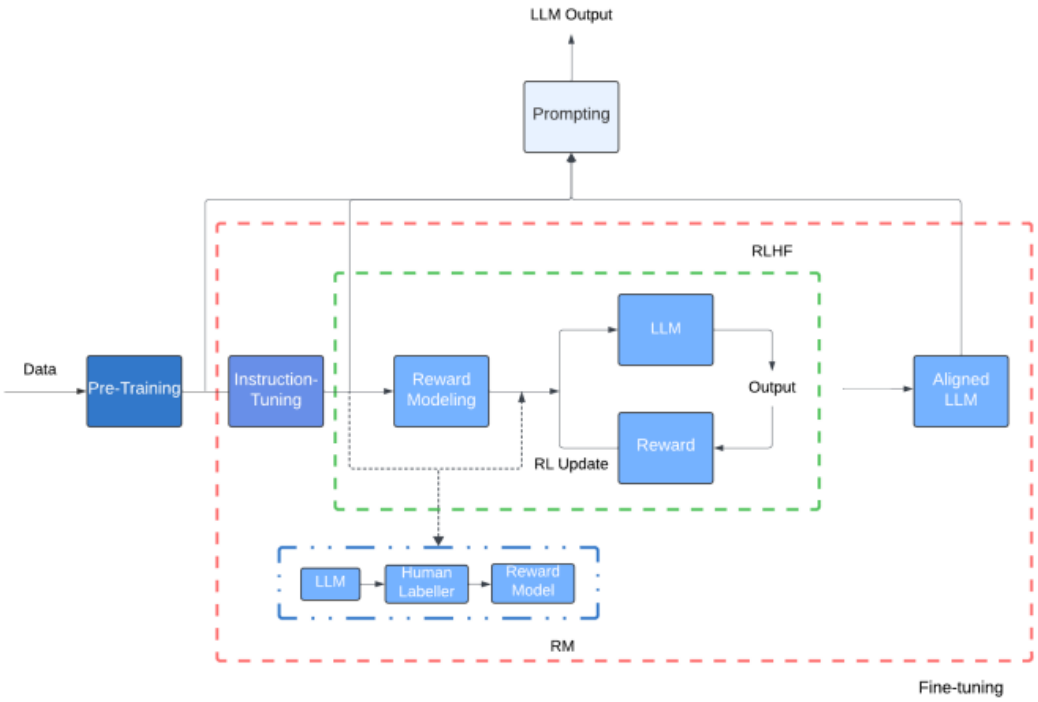

Figure 4.5: A basic flow diagram depicting various stages of LLMs from pre-training to prompting/utilisation [34]

Fine-tuning has been a conventional approach for enhancing the capabilities of pre-trained models. However, the effectiveness of fine-tuning is contingent on the availability of sufficient downstream data [51]. The exploration of alternative methods, such as prompt tuning, suggests a need to reconsider traditional fine-tuning approaches to maximise the potential of LLMs in humour generation.

This section provides an in-depth exploration of fine-tuning, examining its strengths and limitations in the context of humour synthesis. The goal is to understand how fine-tuning contributes to the improvement of LLMs' ability to generate creative and humorous content and to rationalise why this intervention is more useful than prompt engineering.

4.5.2 GPT-driven Humour Applications:

Already there are multiple applications which employ GPT to fine-tune a Large Laughter Model to classify, evaluate and create jokes [6, 7, 26, 29, 53, 54]. These applications all employ the pre-trained GPT-3 model which drives ChatGPT, also developed by OpenAI, to further train a Large Laughter Model to produce more refined jokes.

4.5.3 Strengths of Fine-Tuning

1. **Task-Specific Adaptation:** Fine-tuning enables the adaptation of pre-trained models to specific tasks, making them more proficient in generating content tailored to the requirements of the given task, such as sarcastic jokes [29].

2. **Utilization of Downstream Data:** The effectiveness of fine-tuning relies on the availability of sufficient downstream data. When such data is abundant and representative of the target task, fine-tuning can significantly enhance model performance.

3. **Domain Specialization:** Fine-tuning allows for domain-specific specialization, where the model can be fine-tuned on data from a particular domain, leading to improved performance in tasks related to that domain [26].

4.5.4 Limitations of Fine-Tuning

1. **Data Dependency:** Fine-tuning is highly dependent on the availability of task-specific labelled data. In scenarios where obtaining large amounts of labelled data is challenging, fine-tuning may not be as effective.

2. **Risk of Over fitting:** There is a risk of overfitting to the fine-tuning dataset, especially when the dataset is small. This can lead to a lack of generalisation to new and diverse inputs.

3. **Resource Intensiveness:** Fine-tuning can be resource-intensive, requiring significant computational power and time, especially when dealing with large language models and extensive datasets.

4.5.5 Prompt Engineering vs Fine-Tuning

The central question at hand pertains to the most effective approach for enhancing the humour synthesis capabilities of Large Language Models (LLMs): prompt engineering or fine-tuning. Investigating this, Goes et al.'s research [6] delves into GPT-4's ability to evaluate jokes using a novel many-shot prompting method. Additionally, they explore how different system descriptions impact the evaluation of jokes with LLMs [7].

In the realm of prompt engineering, the produced set of jokes tends to fall short of expectations, with punchlines being rather lacklustre, lacking the desired edgy punchiness. This limitation opens the door for fine-tuning to play a pivotal role. Considering that LLMs like GPT already possess the capability to evaluate joke quality, fine-tuning offers the potential to elevate these capabilities further. It allows the model to grasp the intricacies of joke composition and generate punchlines that align with the learned setups.

Fine-tuning facilitates a more nuanced approach, enabling the testing of the produced joke set against unseen test data. This addresses the challenges associated with original content creation by LLMs, ensuring that the refined Large Laughter Model is genuinely funny rather than being engineered to imitate comedy based on a finite training set. [48].

Furthermore, delving into the intricacies of fine-tuning, we find that this approach offers a unique opportunity to instil adaptability in LLMs, allowing them to evolve their understanding of humour in response to varying prompts and contexts. Unlike prompt engineering, which may struggle to capture the elusive nuances of comedic timing and style, fine-tuning empowers the model to refine its humour generation based on specific feedback. By exposing the model to a diverse range of

prompts during fine-tuning, it can learn to tailor its comedic output to different tones, ensuring a more versatile and context-aware humour synthesis.

However, it is crucial to acknowledge the challenges associated with fine-tuning, such as the risk of overfitting the training data and the need for a well-designed feedback mechanism. Striking the right balance between adapting to new patterns and preserving the model's originality is paramount. The iterative process of fine-tuning demands a thoughtful and dynamic approach to continually improve the model's humour synthesis without losing its intrinsic creativity.

With that being said, ensuring that the correct approach to fine-tune an LLM to produce a Large Laughter Model has been taken, fine-tuning greatly outshines the limited capabilities of prompt engineering. A simple solution is to employ vector embedding and a cosine similarity to ensure that the Large Laughter Model's training and test data are correctly split into a perfect balance, without over-training the model to the point of redundancy and the model reverting back to unoriginal content, keeping in line with the goal of creating a Large Laughter Model that is 'naturally' funny and not engineered to simply imitate comedy it has been trained on.

4.6 Project Management

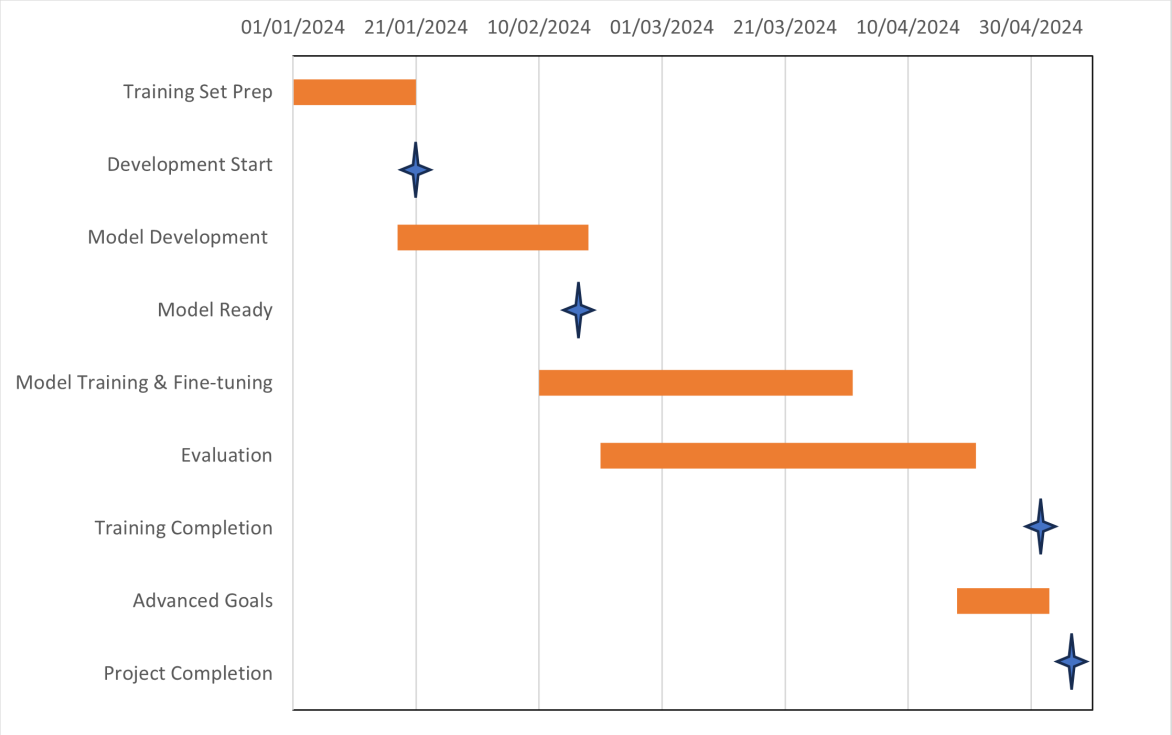

Figure 4.6: Gantt chart illustrating the project breakdown and milestones

Roughly half to a third of the time in the early spring semester will be used to work on and finish the project's core goals, allowing for the remaining time in the semester to be used to complete the advanced goals and the remainder of the report. In terms of the core goals, ideally by the week 2 mark, the training set should be correctly classified in terms of 'nsfw' or clean which will allow the remainder of the time allotted to the training of the LLM model to allow the late half of the spring semester entirely to fine-tuning and refining the model's humour capabilities.

As the advanced goals mainly consist of tweaking the code and training the LLM to produce a more refined set of jokes, I believe this amount of allocated time is adequate. The report will be written as the project reaches milestones and other key development points. Ideally, the project will be completed by the week 12 mark, allowing roughly three weeks of fine-tuning and tweaking the report and potentially addressing any adjustments/issues that arise in the code if required.

4.6.1 Work Structure Breakdown

Preparation of training sets:

- Correctly classify existing jokes within an 'NSFW'/'sfw' range to allow appropriate training data for the LLM to learn from.

Model Development:

- Make LLM group similar jokes together based on setups and punchlines.
- Identify a training and test set.

Model Training:

Develop a training set that includes partial jokes and prompts the model to generate the punchline or setup, evaluating its ability to grasp and extend humour contexts.

- Train the LLM to produce an apt punchline, given a set-up.
- Train the LLM to produce an apt set-up, given a punchline.
- Train the LLM to produce an original joke given a prompt, part of a joke or a joke.

Evaluation:

- Compare produced jokes to training sets based on topic, set-up and punchlines.
- Compare these jokes to a sample produced from prompt engineering
- Utilise vector embedding and cosine similarity algorithms to compare the originality and quality of jokes produced through fine-tuning against those generated using basic prompt engineering.

Advanced Goals Code Development:

- Introduce additional evaluation metrics and algorithms to comprehensively assess the generated jokes, considering factors like incongruity, punchiness, and overall effectiveness.
- These extensions will all be dependent on any issues or unique situations which arise within the time-frame when the evaluation phase begins to take over and the development phase comes to an end.

Chapter 5: Implementation

5.1 Dataset

5.1.1 Reddit_Joke_Dataset.xslx

Before development began, in the pursuit of developing a model capable of generating humorous content, it was imperative to select a comprehensive dataset that would act as the basis for the model's training. An Excel spreadsheet (see file **reddit_dataset.xlsx** within the project architecture as showcased in 1), comprises a collection of jokes, each with a unique structure and content.

```
jokes_df.info()
✓ 0.0s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34761 entries, 0 to 34760
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   Upvotes    34761 non-null  int64 
 1   Tags       454 non-null    object
 2   Setup      34761 non-null  object
 3   Punchline  34761 non-null  object
dtypes: int64(1), object(3)
memory usage: 1.1+ MB
```

Figure 5.1: Preliminary Dataset Analysis

Using a simple dataframe function, ***".info()"*** we are also able to determine some preliminary information about the dataset, with there being 454 marked examples to teach an ML model to classify which jokes were appropriate or not. However given the wider scope of the project being to teach an LLM to "think" for itself, especially with libraries such as **profanity-filter 1.3.3**[55] and **alt-profanity-check 1.4.2**[56], this proved a bit redundant, especially as an ML model runs the risk of false positives/false negatives due to the implicit vs explicit nature of humour.

The existence of 34,761 jokes altogether showcased great potential for an initial training dataset to create a wide array of combinations to train an LLM on.Originally this dataset contained a total of 34,761 jokes in total, split within their respective Setups, Punchlines along with their respective Upvotes, including the 454 marked with their profanity rating.

Using examples of both good and bad jokes from real life, this meticulously curated dataset from Reddit ensures a wide representation of joke types and styles. By utilising the dataset's richness and diversity, the model is able to learn to mimic the structure and style of jokes while abiding by moral guidelines that prevent the creation of offensive content.

	A	B	C	D
1	Upvotes	Tags	Setup	Punchline
2	30499	<clean>	How do you milk sheep?	With iPhone accessories.
3	26603	<nsfw>	Why are women and children evacuated first in a disaster?	So we can think about a solution in silence.
4	26514	<nsfw>	Why do riot police like to get to work early?	To beat the crowd.
5	19153	<nsfw>	What's the difference between everybody and bullets?	Everybody misses Harambe.
6	18000	<nsfw>	Why does Donald Trump take Xanax?	For Hispanic attacks.
7	15454		How many super saiyans does it take to change a light bulb?	Find out next time, on Dragon Ball Z!
8	14795	<clean>	What do you call a sad strawberry?	A blueberry
9	13703	<nsfw>	What's the difference between a hooker and jesus?	The look on their face when you're nailing them.
10	12975	<nsfw>	Damn girl are you a Rubik's cube?	Because fuck you, you stupid piece of shit.
11	12907	<nsfw>	Why was the baby in Africa crying?	It was having a mid-life crisis.
12	12733	<nsfw>	Why did the Mexican take Xanax?	For Hispanic attacks.
13	12467	<nsfw>	Why are redneck murders so hard to solve?	There's no dental records and all the DNA matches
14	12284	<nsfw>	What does Bill say to Hillary after sex?	Honey I'll be home in 20 minutes.
15	12263	<nsfw>	What do a cell phone and anal bleach have in common?	They both change your ring tone
16	12126	<nsfw>	Whats Saudi Arabia's highest rated sitcom?	How I bought your mother
17	12076	<nsfw>	What's the difference between a paycheck and a penis?	You don't have to beg your wife to blow your paycheck
18	11921	<nsfw>	Why didn't barbie ever get pregnant?	Because Ken always came in another box.
19	11890	<clean>	Why does Kylo Ren have no friends?	Because his whole life he's Ben Solo
20	11688	<nsfw>	What's 18 inches long and hangs in front of an asshole?	Donald Trump's tie.
21	11593	<nsfw>	What does a turtle and a pedophile have in common?	They both want to get there before the hare does.

Figure 5.2: Screenshot of the Reddit Joke Dataset

Looking at Figure 5.2, the XLSX file, consists of four columns:

1. The first column, Upvotes, quantifies the popularity of each joke on Reddit, providing a numerical value that reflects the joke's reception and relevance within the online community. This metric is crucial for understanding the joke's impact and relevance, serving as a key indicator of its quality and appeal.

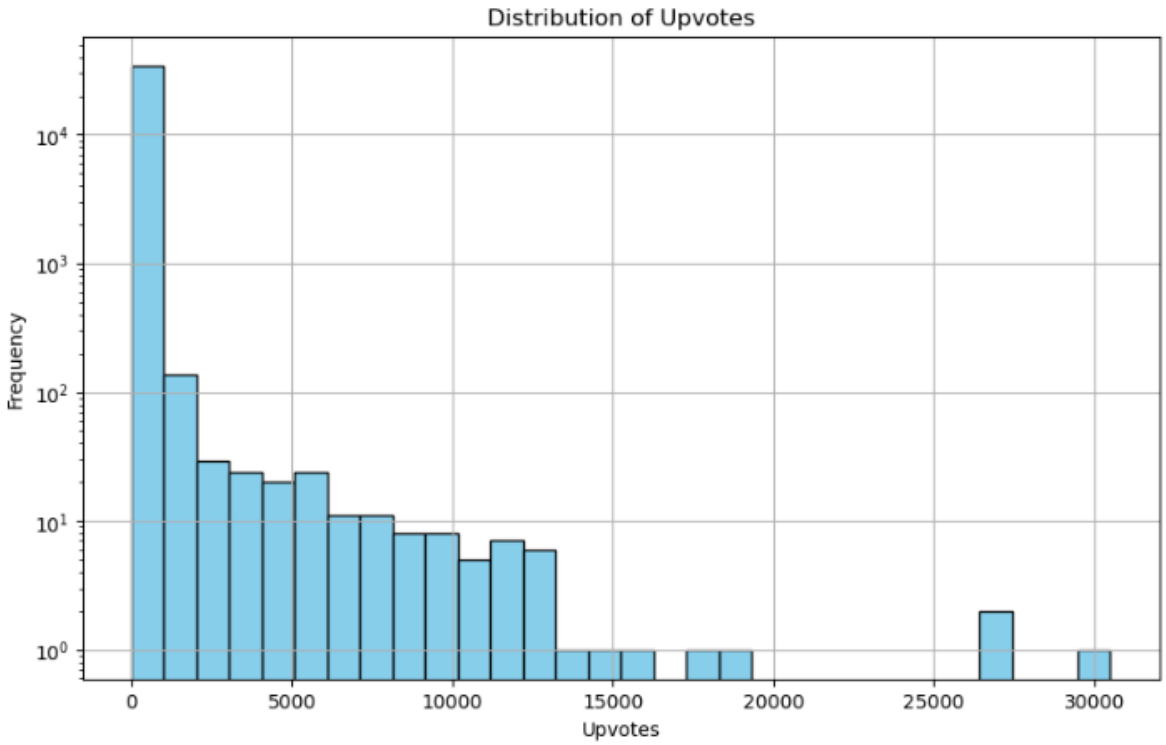

Figure 5.3: Up-vote distribution

As illustrated above, the distribution of upvotes, signified a possibility for a weighted training approach, which could've led to the model having a more nuanced and "human-like" manner

in learning what jokes are preferred over other to teach it more cultural/personal preferences, despite all being examples of good jokes and having no inherent value over the other.

2. The second column, Tags, categorises each joke based on its profanity rating. This categorisation is achieved through the use of three distinct tags: <clean>, <nsfw>, and no value. The <clean> tag indicates that the joke is considered appropriate and safe for all audiences, while the <nsfw> tag marks jokes that contain explicit content and are not suitable for all viewers. The absence of a tag suggests that the joke's profanity rating has not been pre-determined, allowing for a more nuanced analysis of its content.

3. Columns three and four, Setup and Punchline, contain the textual components of each joke. The Setup column provides the lead-in to the joke, setting the stage for the punchline. The Punchline column, as the culmination of the joke, delivers the humorous or unexpected conclusion. Together, these two columns form the complete joke, showcasing the structure and narrative style of the humour.

5.1.2 Feature Engineering

As outlined in Section 4.1 as well as the above in-depth view of our dataset, we must carefully select appropriate jokes to train our LLM on and remove offensive and inappropriate content, for example content which is racist, sexist, homophobic, etc.

Vector Embedding

Table 5.1: Similar Jokes

Setup	Punchline
Did you hear about the constipated math professor?	He worked his problem out with a pencil.
Did you hear about the constipated math teacher?	He had to work it out with a pencil.
Did you hear about the constipated math teacher's problem?	She worked it out with a #2 pencil.
Did you hear about the constipated mathematician?	He got a pencil and worked it out.
Did you hear about the constipated mathematician?	He needed a pencil to work it out.
Did you hear about the constipated mathematician?	He tried to work it out with a slide rule.
Did you hear about the constipated mathematician?	He worked his problem out with a pencil.

For example as shown in the above table 5.1, a simple change in wording from "you are" to "you're" creates a "new" joke from a programming language's viewpoint, yet to a human eye it is obvious that the joke is repeated, therefore finding jokes which were semantically/grammatically similar manually was not feasible with over 30 thousand entries.

The decision to use vector embedding and cosine similarities in the joke analysis is rooted in the need for a sophisticated method to measure similarity. Vector embedding map text into a high-dimensional space, capturing semantic meanings, while cosine similarity measures the angle between vectors, focusing solely on semantic similarity. This method allows for the identification of not just exact duplicates but also jokes conveying similar ideas or themes greatly reducing the chances of over-training the model.

Profanity Filtration

The integration of the python 'alt-profanity-check'[1] library was instrumental in ensuring the content's appropriateness and safety. This library, a robust and fast tool for detecting offensive

[1] https://pypi.org/project/alt-profanity-check/

language in strings, was utilised to filter out any content with a profanity rating of 0.5 and above. The reasoning for choosing 0.49 as my threshold is because, for example I found a joke with the setup "why did the chicken cross the road?" to have a rating of 0.2777 and its punchline "it was stapled to the child's face" having a rating of 0.3132, the punchline does raise a warning flag however without context, this could be a very harmless statement, perhaps its a smile that was "stapled to the child's face"? However when combined with the setup to provide context; "Why did the chicken cross the road? It was stapled to the child's face" had a rating of 0.496, which correctly assessed that separately the setup and punchline are harmless without being given context (separately there isn't anything inherently offensive nor inappropriate about this joke) yet when combined the joke does have an inherent theme of child harm.

Other jokes which were around this rating seemed to also be inherently safe, albeit not particularly funny while jokes above 0.5 seemed to be outright offensive, with an example of a joke with the rating of 0.5 being "What do you call a twelve faced shape made out of genitals? A dodickahedron." with the offence in said joke revolving around the element of genitalia.

False Positives

At this point, 22,414 jokes were filtered out due to their explicit and implicit humour, however upon manual inspection there seemed to be false positives which had leaked through due to slang, cultural terms or more "new-age" terms which had not been used to train any of the profanity library models yet.

To easily tackle this problem, a dataset of updated slang terms, bad words as well as the blacklist of words, terms and phrases from common public domains were added into a simple txt file and used to filter out an additional 14,808 jokes which upon more manual inspection, did indeed need filtration. An example of this would be the use of the phrase "2 girls, 1 cup" which references to a rather obscene video with themes of excrement and other adult themes, from a few years ago, which could lead to the fine-tuned models producing jokes which could be deemed as vastly inappropriate due to these themes, which must be vigorously not allowed to leak into the training data.

JSON File

For the training of the model, the next step would be to split the cleaned dataset into a 70-30 split for training and validation. At this point, the dataset contained about 7606 jokes, which was now split into a 70-30% split for training and validation, respectively.

As OpenAI require the training and validation file content to be similar to a conversation that you would ask the model to respond to at inference time in production. With their documentation stating that "each example in the dataset should be a conversation in the same format as our Chat Completions API, specifically a list of messages where each message has a role, content, and optional name. At least some of the training examples should directly target cases where the prompted model is not behaving as desired, and the provided assistant messages in the data should be the ideal responses you want the model to provide."[57]

```
[
  {"messages":
    [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."},
     {"role": "user", "content": "What's the capital of France?"},
     {"role": "assistant", "content": "Paris, as if everyone doesn't know that already."}
     ]
    },
```

```
  {"messages":
    [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."},
     {"role": "user", "content": "Who wrote 'Romeo and Juliet'?"},
     {"role": "assistant", "content": "Oh, just some guy named William Shakespeare."}
     ]
    },
  {"messages":
    [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."},
     {"role": "user", "content": "How far is the Moon from Earth?"},
     {"role": "assistant", "content": "Around 384,400 kilometers. Give or take a few"}
     ]
   }
  ...
]
```

Using the function "format_item_for_jsonl' the final cleaned dataset is converted into the respective training and validation file with the format of:

```
[
    {"role": "system",
     "content": "system prompt"},
    {"role": "user",
     "content": "setup: setup_content"},
    {"role": "assistant",
     "content": "punchline: punchline_content"}
]
```

The system prompt chosen was "Goofy is a joke writer for a standup comedian." as found in the various investigations; a longer system prompt lead to a more expensive as well as more inefficient API calls with GPT's models as well as the fine-tuned models giving more incoherent content as they had more tokens to process.

5.2 Model Training

This section goes into an overview of the fine-tuning process used to create 3 models:

1. LFT_Model_1 **(ft:gpt-3.5-turbo-1106:personal::906Y74tA)**

   - A fine-tuned model on a large dataset consisting of roughly 7000 jokes and a validation dataset of around 300 jokes., trained on 2 epochs.

   - LFT_Model_1 would serve as the benchmark model and the main LLM which will be used to analyse the humour capabilities of fine-tuning a GPT baseline model.

   - Additionally, this model would be also used to investigate the dangers of over-training a model, to showcase GPT's original problem when contrasted with a model without this limitation.

2. LFT_Model_2 **(ft:gpt-3.5-turbo-1106:personal:thesis-small:9DiVMMLB)**

   - LFT_Model_1 re-trained using the smaller dataset for SFT_Model to highlight the issues with overtraining an LLM

3. SFT_Model **(ft:job-tNdLv1GghEEhAeUs2UNjzaED)**

   - A fine-tuned model on a smaller dataset consisting of roughly 100 clean jokes within its training dataset and a validation dataset of around 30 jokes, trained on 5 epochs.

   - The hopes for this model would be that, perhaps an additional problem often overlooked would be LLMs relying too often on a large selection of data from its training, restricting its potential at creativity and "thinking" for itself.

The reasoning behind these three models was to investigate the effect of using a smaller/larger dataset and more/fewer epochs during training would have on the quality of the jokes produced by the models, as well as to pit the models against one another and compare a baseline model (**gpt-3.5-turbo-16k-0613**) against a fine-tuned model and the original human generated punchlines, with the aim and hopes that the fine-tuned model would outperform the baseline and hopefully be somewhat on par with the original human jokes.

5.2.1 API Functions Used

API function to create training file

```
training_file = client.files.create(
  file=open(training_file, "rb"),
  purpose="fine-tune"
)

print(training_file.id)
```

API function to create validation file

```
validation_file = client.files.create(
  file=open(validation_file, "rb"),
  purpose="fine-tune"
)

print(validation_file.id)
```

API function to start fine-tuning job

```
client.fine_tuning.jobs.create(
 training_file=training_file.id,
 validation_file=validation_file.id,
 model="gpt-3.5-turbo-1106",
 hyperparameters={
   "n_epochs": num_epochs
 }
)
```

API function to generate topics for joke:

```
response = client.chat.completions.create(
```

```
   model=model_5,
   messages=[{"role": "system", "content": "You are a joke writer for a comedian."},
             {"role": "user", "content": "From what you know about what makes a good
             joke, give me 10 topics to write jokes on"}]
)

print(response.choices[0].message.content)
```

API function to generate topics for joke:

```
response = client.chat.completions.create(
   model=model_5,
   messages=[{"role": "system", "content": "You are a joke writer for a comedian."},
             {"role": "user", "content": "From what you know about what makes a good
             joke, give me 10 topics to write jokes on"}]
)

print(response.choices[0].message.content)
```

API function to generate setups for joke:

```
response = client.chat.completions.create(
    model=model_5,
    messages=[{"role": "system", "content": "You are a joke writer for a comedian."},
            {"role": "user", "content": "From what you know about what makes a funny joke,
             give me 10 setups to a joke on the topic of" + topic}]
)

print(response.choices[0].message.content)
```

Function to generate Punchline:

```
# Define the number of punchlines per setup
num_punchlines = 5

# Iterate through each setup question
for setup_num in range(1, 11):
  # Print the current setup question
  print("Setup:", globals()["setup_" + str(setup_num)])

  # Generate five punchlines for the current setup
  for attempt in range(num_punchlines):
     # Call the GPT-3 API to generate responses using the fine-tuned model
     response = client.chat.completions.create(
        model=model_2_epochs,
        messages=[
            {"role": "system", "content": "You are a joke writer for a comedian."},
            {"role": "user", "content": "setup: " + globals()["setup_" + str(setup_num)]}
        ])
        # Print the generated punchline
        print(response.choices[0].message.content)
    # Add an empty line for separation between setups
    print()
```

5.3 Data Collection (pre-evaluation)

Data Collection was a rigorous task, manually obtaining 20 unique topics from jokes using a combination of Toplyn's prompt engineering algorithm, ChatGPT's chatbot[2] (to avoid a costly use of the baseline model's API calls, which serves a similar purpose as the chatbot).

The success of our approach will be measured through both quantitative and qualitative means. Quantitatively, we will assess the model's originality and quality of generated jokes using established metrics such as verbosity, profanity and coherency.
Qualitatively, audience feedback in the form of a test group will provide insights into the perceived humour of the generated content.

Additionally

- A randomly selected set of 12 setups from the original dataset (after checking they weren't inherently offensive nor inappropriate) were selected which was then used to obtain 24 punchlines from LFT_Model_1 as well as the baseline model for qualitative analysis.
- A randomly selected set of 12 further setups were selected from the collected data (after checking they weren't inherently offensive nor inappropriate) which were then used to obtain 12 further punchlines from SFT_Model to compare and contrast against LFT_Model_1 for another set of qualitative analysis.
- Finally, LFT_Model_2 would be run concurrently to contrast to the baseline model and investigate the impact of overtraining a model

Finally we have 2060 jokes in total, 1000 from LFT_Model_1 and 1000 from the baseline model, 24 from LFT_Model_1 and SFT_Model for the first set of qualitative analysis, an additional 24 from LFT_Model_2 for the second set and the final 12 for the over training analysis.

[2] https://chat.openai.com/

Chapter 6: Collected Jokes

This chapter gives an overview of the collected setups for each topic generated by SFT_Model. please refer to each respective file (labelled "topic".xlsx, for a full presentation of collected punchlines) 1.

Driving

1. "Did you hear about the car that was sent to therapy?"
2. "What makes a bad driver?"
3. "What makes a great driver?"
4. "What did the car tell its friend?"
5. "Why did the car drive across the road?"
6. "Why did the driver honk?"
7. "How did the driver get lost?"
8. "Why are car jokes not funny?"
9. "What's the best way to handle bad drivers?"
10. "How do you confuse a car?"

Books

1. "Why did the book go to therapy?"
2. "What did the book tell its friend?"
3. "What's a bookworm's favourite song?"
4. "How do books like to party?"
5. "Why did the author have an affair?"
6. "What kind of books do aliens like to read?"
7. "How do you make a good book?"
8. "What makes a bad book?"
9. "What did the book say to the librarian?"
10. "What's an author's favourite time of day?"

Anime

1. "Why did the anime fan bring a ladder to the convention?"
2. "What do anime villains and bad wifi have in common?"
3. "What's an anime character's favourite song?"
4. "Why do anime heroes never get a cold?"
5. "Why does Luffy from One Piece make a great pirate?"

6. "What did the Pokemon trainer tell their friend?"
7. "Why don't Attack On Titan characters use online dating?"
8. "Why was the One Piece fan bad at soccer?"
9. "What did the Naruto fan say at the job interview?"
10. "Why did the anime fan go to therapy?"

Gym/Fitness

1. "Why are bodybuilders bad at gambling?"
2. "Why did the bodybuilder go to therapy?"
3. "Why did the powerlifter get a ticket?"
4. "How do you know if someone does CrossFit?"
5. "Why did the gym close down?"
6. "What is an athlete's favourite song?"
7. "What do you call a gym full of lawyers?"
8. "Why did the yoga instructor get a divorce?"
9. "What do you call a fit apple?"
10. "What do you call a fit doctor?"

Supernatural

1. "Why couldn't the vampire go to the party?"
2. "Why do ghosts hate the rain?"
3. "Why did the ghost break up with its partner?"
4. "Why are vampires such bad liars?"
5. "What is a skeleton's favourite kind of music?"
6. "How do you know if a werewolf is lying to you?"
7. "How do you make a witch happy?"
8. "What do you call two witches living together?"
9. "Why did the werewolf go to the circus?"
10. "Why don't spirits ever get lost in the woods?"

Animals

1. "What do you call a sheep that's always quiet?"
2. "Why aren't fish jokes funny?"
3. "How do you catch a squirrel?"
4. "Why did the dog go to therapy?"
5. "Why are spiders great at art?"
6. "Why do bees always hum?"
7. "How do you make a cat bark?"
8. "What do you get when you cross a lion and a goat?"
9. "What do you call a pampered cow?"

10. "Why can't you make a bird joke?"

Travelling

1. "Why can't you make a joke about mountains?"
2. "What's the best travelling song?"
3. "What do you call a wandering chicken?"
4. "Where's the best place to visit?"
5. "What's the worst hotel?"
6. "How do you know if someone travels often?"
7. "Have you heard a joke about travelling?"
8. "Why did the pilot go to therapy?"
9. "Why did the tourist get kicked off the train?"
10. "How do you avoid getting lost on a trip?"

Random Jokes

1. "What do you call a nosy pepper?"
2. "How many ants does it take to change a lightbulb?"
3. "Did you know I'm scared of origami?"
4. "Do you know any jokes about paper?"
5. "Why did the scarecrow win an award?"
6. "A man walks into a bar. . . "
7. "Did you know I hate coffee?"
8. "Did you hear the joke about the chiropractor?"
9. "How do you organise a space party?"
10. "Why did the chicken cross the road?"

Science

1. "Why did the physicist break up with the biology major?"
2. "What do you do with a dead chemist?"
3. "What is the most important rule in chemistry?"
4. "Do you want to hear a joke about elements?"
5. "Why did the scientist go to therapy?"
6. "Want to hear a science joke?"
7. "What did the scientist tell their friend?"
8. "Why did the monkey become a scientist?"
9. "Why did the scientist have an affair?"
10. "What's an author's favourite time of day?"

Superheroes

1. "Why did Batman stop using the internet?"
2. "Why is Batman always calm?"
3. "What's Spider-Man's favorite day of the week?"
4. "Why did the Flash quit the Justice League?"
5. "Why did the Flash get a ticket?"
6. "How does a superhero get around?"
7. "Why was the superhero cold at night?"
8. "Why did Superman get a job at the newspaper?"
9. "Why did the superhero go to therapy?"
10. "What's a villain's worst nightmare?"

Pickuplines

1. "Are you a cat?"
2. "Can I follow you home?"
3. "Are you today?"
4. "I'm no scientist. . . "
5. "Are you a camera?"
6. "Are you shoes?"
7. "Are you a robot?"
8. "Do you like pictures?"
9. "What's your name?"
10. "Are you wi-fi?"

Music

1. "Why did the musicians go to jail?"
2. "What kind of music do cats listen to?"
3. "Why did the musicians get a ticket?"
4. "How many musicians does it take to change a light bulb?"
5. "Why did the rock band break up?"
6. "How do you fix a broken tuba?"
7. "Why did the guitar teacher go to jail?"
8. "Why did the musician break up with their partner?"
9. "Why did the iPod go to school?"
10. "What's a skeleton's favourite song?"

Football/Sports

1. "Why did the footballer go to therapy?"
2. "Why don't football players text you back?"

3. "Why was the football so good at making friends?"
4. "What's a football player's favorite kind of music?"
5. "Why did the footballer have an affair?"
6. "How do you know spiders are huge football fans?"
7. "Why are football stadiums the coolest places to be?"
8. "Why did the football coach go to the post office?"
9. "What did the football say to the punter?"
10. "Why did the football team go to the bakery?"

Food

---

1. "Why did the tomato turn red?"
2. "Why did the banana go to the doctor?"
3. "What do you call cheese that isn't yours?"
4. "Why don't eggs tell jokes?"
5. "What kind of nuts are always sick?"
6. "Why was the grape stressed?"
7. "Why is bread always unhappy?"
8. "Do you know any jokes about food?"
9. "Why did the chef have an affair?"
10. "What's Gordon Ramsey's favourite dish?"

Fishing

---

1. "Why don't fish like basketball?"
2. "Why don't fish ever get lost?"
3. "Why don't fisherman ever share?"
4. "What kind of music do fish listen to?"
5. "Why did the fish join a band?"
6. "What's a shark's favourite TV show?"
7. "Why don't fishermen play cards?"
8. "How do fish celebrate their birthdays?"
9. "What's a shark's favourite food?"
10. "How do you make an octopus laugh?"

Education/School

---

1. "Why do matheicians look sad?"
2. "Why was the student always sleepy in science class?"
3. "Why did the art student never have any money?"
4. "Why did the computer scientist hate coding?"
5. "Why did the apple go to school?"
6. "What did one pencil say to the other?"

7. "Why did the scarescrow do well in school?"
8. "Why did the teacher go to therapy?"
9. "Why did the farmer bring a ladder to school?"
10. "What do you call a jobless geography teacher?"

Movies

1. "Do you like Star Wars?"
2. "Why did the actor break-up with their co-star?"
3. "How do you make a bad movie?"
4. "What's the worst thing to say in a horror movie?"
5. "Why do actors need therapy?"
6. "Why did Chuck Norris quit acting?"
7. "Why did the actor have an affair?"
8. "How do you make a great movie?"
9. "Why don't scientists make movies?"
10. "What's a vampire's favourite movie?"

Space

1. "Why did the astronaut break up with his girlfriend?"
2. "What's a scientist's favorite planet?"
3. "Why can't astronauts make friends?"
4. "Did you know NASA spoke to aliens?"
5. "How do you make an alien laugh?"
6. "Why was the astronaut nervous?"
7. "Why did the alien go to the comedy club?"
8. "What did the alien tell its friend?"
9. "Why did the astronaut go to therapy?"
10. "Why did the Martian kill its friend?"

Law/Politics

1. "Why did the politician have an affair?"
2. "Why did the politician start dancing?"
3. "How many politicians does it take to change a lightbulb?"
4. "Why did the politician go to therapy?"
5. "What do you get when you cross a politician with a potatoe?"
6. "Why did the lawyer get a divorce?"
7. "Why do birds avoid politics?"
8. "What's the difference between a politician and a chicken?"
9. "Did you hear about the judge who went on holiday?"
10. "Why is politics boring?"

Technology

1. "Why did the computer catch a cold?"
2. "Why is computer dating a bad idea?"
3. "Why did the software developer go broke?"
4. "Why do programmers prefer dark mode?"
5. "Why was the JavaScript developer sad?"
6. "How many programmers does it take to change a light bulb?"
7. "Why was the phone wearing glasses?"
8. "Why did the phone go to therapy?"
9. "How does a robot apologize?"
10. "How does a computer get drunk?"

6.1 Data Format

	A	B	C	D	E	F	G	H	I
1	Setup	Set 1	Verbosity Set 1	Coherency Set 1	Profanity Set 1	Set 2	Verbosity Set 2	Coherency Set 2	Profanity Set 2
2	Did you hear about the car that	It had metal problems.	5	Funny	0.037323103	It had a lot of road rage.	8	Funny	0.118144944
3	Did you hear about the car that	Because it was EXHAUSTE	5	Funny	0.051862556	It stored a lot of baggage.	7	Funny	0.038601242
4	Did you hear about the car that	It was having an auto body	8	Funny	0.042660214	It had a lot of unresolved brake issues.	9	Passable	0.093612103
5	Did you hear about the car that	It had a lot of idle issues.	8	Funny	0.030647613	It had a lot of breakdowns	6	Funny	0.03540604
6	Did you hear about the car that	It had been a long road to r	9	Funny	0.032862669	It had a lot of transmission problems.	8	Passable	0.062369476
7	What makes a bad driver?	The highway to hell.	5	Passable	0.486246464	Texting while driving... or as I like to call it,	20	Passable	0.038481486
8	What makes a bad driver?	Breaking instead of accele	9	Incoherent	0.240223134	It's not the car's fault for having bad brake	23	Funny	0.065877072
9	What makes a bad driver?	Asking what his 5th wheel i	8	Funny	0.014673344	They treat stop signs like suggestions.	7	Funny	0.004861404
10	What makes a bad driver?	Alcohol.	2	Funny	0.025162482	They think "SPEED LIMIT" is a challenge,	14	Funny	0.013873365
11	What makes a bad driver?	Nothing. They're just bad.	7	Passable	0.057517553	They think the left lane is only for taking p	12	Funny	0.011616355
12	What makes a great driver?	Ask Vin Diesel.	4	Funny	0.013256075	Knowing that the left lane is for passing, r	13	Passable	0.014814997
13	What makes a great driver?	No witnesses.	3	Funny	0.007797847	A blindfold and a "Thelma and Louise" D	14	Passable	0.052752294
14	What makes a great driver?	Alcohol.	2	Funny	0.012438163	Knowing that the pedal on the right is not	12	Passable	0.028664331
15	What makes a great driver?	A cop in the passenger sea	7	Funny	0.056506908	Being able to parallel park without hitting	10	Passable	0.067260508
16	What makes a great driver?	Red and blue lights in the re	9	Funny	0.024319817	The ability to merge without causing a tra	12	Passable	0.028656834
17	What did the car tell its friend?	I have metal problems.	5	Funny	0.035043842	I'm tire-d of this traffic.	7	Funny	0.027293122
18	What did the car tell its friend?	I've had a road of ups and d	10	Funny	0.042224737	I brake for no-apparent reason.	6	Funny	0.136068087
19	What did the car tell its friend?	I'm tire-d of you.	6	Funny	0.024439451	I've got a lot of mileage on me but I'm still r	15	Funny	0.024418266
20	What did the car tell its friend?	Nothing, it just beeped!	6	Funny	0.029906054	Tire's no use worrying about things!	8	Funny	0.031553682
21	What did the car tell its friend?	I'm on the Highway to Hell!	8	Passable	0.449261018	I'm quite exhuasted	4	Funny	0.029323402
22	Why did the car drive across the	Because its tires were on th	9	Funny	0.68965634	To get to the other side of the traffic jam!	11	Funny	0.375421252
23	Why did the car drive across the	The green light told him to g	8	Funny	0.056892401	To get to the mechanic on the other side	10	Funny	0.352877103
24	Why did the car drive across the	Because the driver was a c	7	Passable	0.435334195	To get to the mechanic's office and ask f	18	Incoherent	0.057323707
25	Why did the car drive across the	To escape the highway to h	7	Incoherent	0.51506072	To get to the other side of the DMV.	10	Passable	0.352877103
26	Why did the car drive across the	To get to the other road.	7	Passable	0.369870976	To get to the mechanic on the other side	10	Funny	0.352877103
27	Why did the driver honk?	To tell someone to hurry up	7	Passable	0.052412722	He wanted to wake up the pedestrians wl	19	Incoherent	0.16821606
28	Why did the driver honk?	Because he had anger ste	7	Funny	0.097151129	Because he wanted to hear his own horr	10	Passable	0.028489297
29	Why did the driver honk?	Because he stepped on th	7	Funny	0.025491253	To get to the other side of the traffic jam!	11	Funny	0.090840249
30	Why did the driver honk?	To scare the duck out of its	10	Incoherent	0.050031678	Because he wanted to trumpet his achie	10	Incoherent	0.017685454
31	Why did the driver honk?	To get to the other side. Ha	15	Incoherent	0.259911857	Because he wanted to trumpet his own h	9	Incoherent	0.015053421
32	How did the driver get lost?	He took a wrong turn.	6	Funny	0.081481791	He was following his GPS, but it turned ou	21	Incoherent	0.066623767
33	How did the driver get lost?	He was on the highway to h	7	Funny	0.621250074	He refused to take directions from his GP	34	Incoherent	0.021147179
34	How did the driver get lost?	He took a "dead end."	8	Passable	0.121809369	He was following the directions on his toa	9	Funny	0.079364845
35	How did the driver get lost?	He lost his Waze.	5	Funny	0.145210444	He was relying on his pet snail's sense of	12	Funny	0.012442188
36	How did the driver get lost?	I don't know, he was followi	10	Passable	0.083506447	He took directions from his GPS... and en	17	Passable	0.07235982
37	Why are car jokes not funny?	They always get exhausted	5	Funny	0.109888954	Because they always drive people away.	7	Funny	0.28212417
38	Why are car jokes not funny?	They don't work for everyon	7	Funny	0.086976578	Because they always drive into the groun	8	Funny	0.119171027
39	Why are car jokes not funny?	They require maintenance	3	Funny	0.039134858	Because you're driving them into the gro	9	Funny	0.03843619
40	Why are car jokes not funny?	Because they are tire-ing.	5	Funny	0.477968115	Because they are tire-ing.	5	Funny	0.477968115
41	Why are car jokes not funny?	Because they're driving pe	7	Funny	0.105322984	Because they're all exhausted	5	Funny	0.109888954
42	What's the best way to handle b	With an automatic transmis	5	Passable	0.043913932	Avoid them by becoming an astronaut - t	14	Funny	0.032101396
43	What's the best way to handle b	Alcohol.	2	Funny	0.017859847	Avoid them like the plague, unless of cou	22	Incoherent	0.033513475
44	What's the best way to handle b	You steal the headlights.	5	Incoherent	0.040861014	Just remember, they're not worth the insu	12	Funny	0.011579822
45	What's the best way to handle b	A soapbox with Velcro.	5	Incoherent	0.033486368	Just don't be one of them.	8	Funny	0.027239348
46	What's the best way to handle b	By the ears!	4	Incoherent	0.033101387	Become a good driver yourself and leave	24	Passable	0.036367237
47	How do you confuse a car?	You steal the headlights.	5	Incoherent	0.039184698	Show it a stick shift and tell it to drive in re	13	Incoherent	0.123709052
48	How do you confuse a car?	Put it in reverse and let it sm	9	Incoherent	0.06917424	Put it in a roundabout and watch it go in c	20	Passable	0.005205184
49	How do you confuse a car?	Paint the roof and tell it a go	10	Incoherent	0.02734017	Put it in reverse and ask where it wants to	12	Passable	0.010068748
50	How do you confuse a car?	The Highway to Hell	4	Passable	0.393359345	Take it to a roundabout and tell it to go str	12	Funny	0.031216657
51	How do you confuse a car?	Paint the doors on backwa	6	Incoherent	0.011935216	You take it to the mechanic and tell them	15	Incoherent	0.015595889

Figure 6.1: Analysis Example 1: Driving Jokes

This is a sample of what each file looked like after data collection and quantitative analysis was complete. Please refer to code repository for a detailed overview on each file.

Chapter 7: **Analysis**

This chapter provides a thorough examination of the research topic, weaving together both quantitative and qualitative elements. The first section transitions into a detailed exploration of the quantitative analysis, discussing the statistical significance of the findings and their broader implications, with subsequent sections shifting focus to the qualitative aspects, detailing the methodologies used to uncover and analyse themes, patterns, and insights, and finally concludes with the overtraining analysis.

7.1 Quantitative Analysis

For the quantitative analysis, a rigorous methodology was employed to ensure the accuracy and reliability of the data. The profanity checker, in conjunction with the Natural Language Toolkit[1] (NLTK) library [58], was utilized to analyze the length of each joke, encompassing both the setup and the punchline. This approach allowed for a comprehensive evaluation of the jokes' structure and content, providing a quantifiable measure of their complexity and humour potential.

The profanity checker used for the data filtration process was once again used, to analyse how profane each model was despite having been only trained on "clean" data. The hopes was that despite a "clean" dataset, the fine-tuned models would still be able to use suggestive wordplays and edgier "safe" content as trained from the reddit dataset.

Additionally, the dataset was manually tagged with one of three values, <Incoherent>, <Passable>, <Funny> in each of the respective 'Coherency' columns, with a joke being marked as <Incoherent> if it either made no sense or was simply a poor attempt at a joke, <Passable> if the joke was simply not funny but a genuine attempt was made at creating it and <Funny> if a joke well well constructed, comedic and made sense.

To avoid bias, these values were then validated by asking friends and family to review and give their opinions, with the majority ruling in favour for a value.

The steps used to carry out this experiment were as follows:

1. Preliminary analysis of the datasets
2. Feature Engineering: - Manually tagging for coherency values - Handling repeated/similar jokes via vector s and cosine similarities - Applying NLTK and Alt-Profanity-Check
3. Data Analysis using GPT (Data Analyser Model via ChatBot)
4. Data Analysis & Visualisations - Verbosity: measured from 0-100, using NLTK libary - Profanity: measured from 0 - 1 using Alt-Profanity-Check
5. Statistical & Probability Analysis

[1] https://www.nltk.org/

Following the initial analysis, standard data analysis techniques were applied to the collected data. This involved calculating the mean, median, and standard deviation of the joke lengths, allowing for a deeper understanding of the distribution of joke lengths within the dataset. Additionally, visualization techniques were employed to graphically represent the data, facilitating a clearer understanding of the trends and patterns within the joke lengths.

7.1.1 Type of Humour

Using ChatGPT to analyse each of the setups/punchlines obtained, I found that the best sets of prompts were ones which were more open-ended and less specific, for example "why did the car go to therapy" originally felt as though was a rather niche subject, however LPT_Model_1 creatively came up with a wide array of car and road related puns.

As well as that, when asked to analyse the punchlines obtained, GPT-4's data analyser model (through the ChatBot interface) suggested that the type of humour generated by LPT_Model_1 was more creative, full of puns and double entendres whereas on the topics such as politics, for example "why did the politician have an affair", "what is a writer's favourite time of day?" the responses gotten were more "literal" than creative.

However there were cases as highlighted in some of the case studies, there was instances of the guardrails breaking and the LLM's true capabilities peaking through, for example for the topic of travelling with the setup: "What's the worst hotel?", the other responses gotten were satisfactory but safe, however response 4/5 from LPT_Model_1 was "Auschwitz" which is a clear example of its guardrails breaking and the model giving a very edgy content with dark humour.

7.1.2 Insights

Straight off the bat, the easiest conclusions I found were that the fine-tuned model gave far more nuances to its punchlines than the baseline model. The fine-tuned model actually tried to be creative albeit failing at it at times, whereas the baseline model as well as LPT_Model_2 kept giving the same sets of jokes repeatedly.

Additionally, I found that music, science and were both fine-tuned models as well as GPT's best topics. When given back to GPT to analyse via API calls, a possible suggestion I got was because due to these topics and their respective setups, open-endedness, "less thinking" was involved as a wide array of punchlines were possible. however when it came to more niche, or topical prompts such as law/politics, random jokes or even pickup lines, GPT struggled to give comedic responses while LPT_Model_1 gave creative albeit not comedic responses.

Drawing on the extensive research conducted within the literature review, a possible explanation could be due to a lack of training data/examples for it to draw upon as well as the railguards from OpenAI moderating the models' outputs ensuring that more topical or "edgy" content isn't given, content that one would rate above PG-13 "edgy" that is.

7.1.3 Results:

Coherency, verbosity, and the presence of profanity are factors that can influence the perceived quality of a joke. Coherent jokes are more likely to be understood and appreciated by audiences, while verbosity and profanity may detract from the humor's effectiveness.

Understanding the relationship between these factors and joke quality is crucial for improving joke generation algorithms and enhancing the user experience. By investigating the impact of verbosity and profanity on joke coherency, we aim to provide insights into the underlying mechanisms of humor generation and inform the development of more sophisticated computational humor systems.

After careful evaluation (please refer to analysis.ipynb file within code repository) the following values were calculated:

Metric Averages:

Average Coherency for LFT_Model_1: 0.8548895899053628%
Average Verbosity for LFT_Model_1:: 6.437434279705573%
Average Profanity for LFT_Model_1:: 0.1002201421264598%
Average Coherency for Baseline Model: 0.7975814931650894%
Average Verbosity for Baseline Model: 10.360672975814932%
Average Profanity for Baseline Model: 0.09228614036605852%

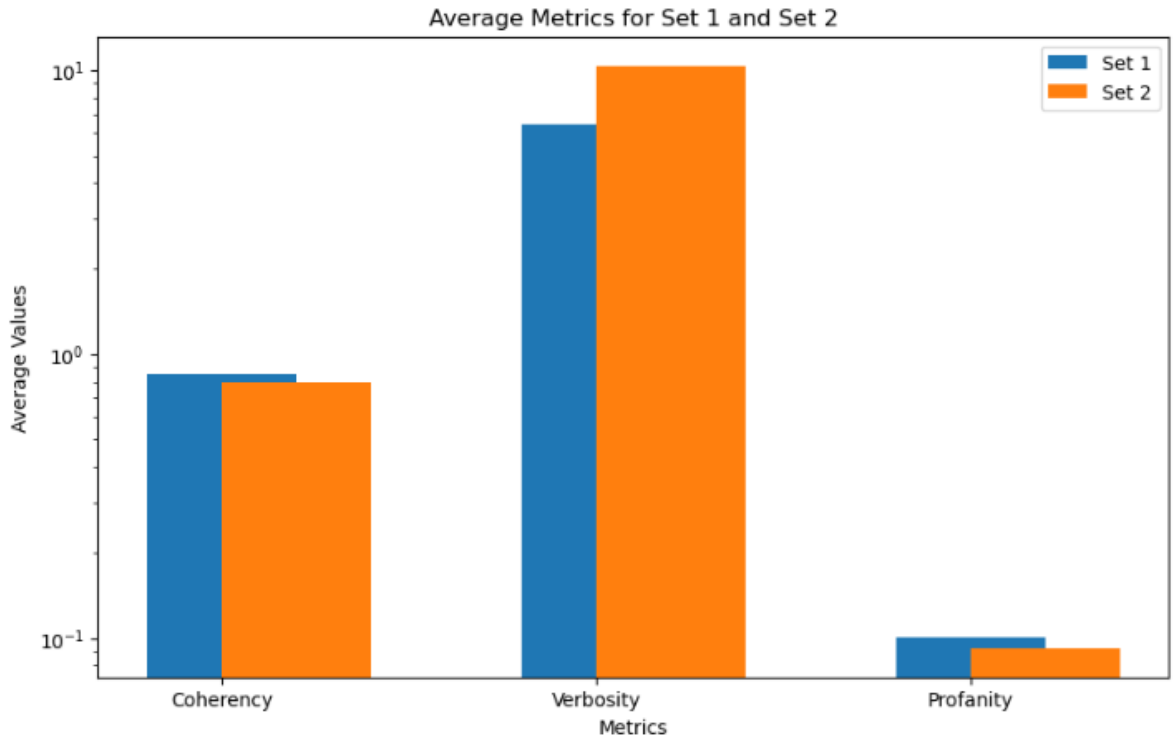

Figure 7.1: Metric Value Distributions

Metric Mean, Median, Mode

Table 7.1: **LFT_Model_1 Statistics**

	Mean	Median
Mode		
Coherency	0.85489	1.00000
1.000000		
Verbosity	6.56000	7.00000
5.000000		
Profanity	0.10022	0.04302
0.182336		

Table 7.2: **Baseline Model Statistics**

	Mean	Median
Mode		
Coherency	0.797581	1.000000
1.000000		
Verbosity	12.240000	11.000000
12.000000		
Profanity	0.092286	0.046379
0.013291		

Table 7.3: **Correlation for LFT_Model_1**

	Coherency	Verbosity	Profanity
Coherency	1.000000	-0.280804	0.002460
Verbosity	-0.280804	1.000000	0.164819
Profanity	0.002460	0.164819	1.000000

Table 7.4: **Correlation for Baseline Model**

	Coherency	Verbosity	Profanity
Coherency	1.000000	-0.545897	-0.076849
Verbosity	-0.545897	1.000000	-0.244292
Profanity	-0.076849	-0.244292	1.000000

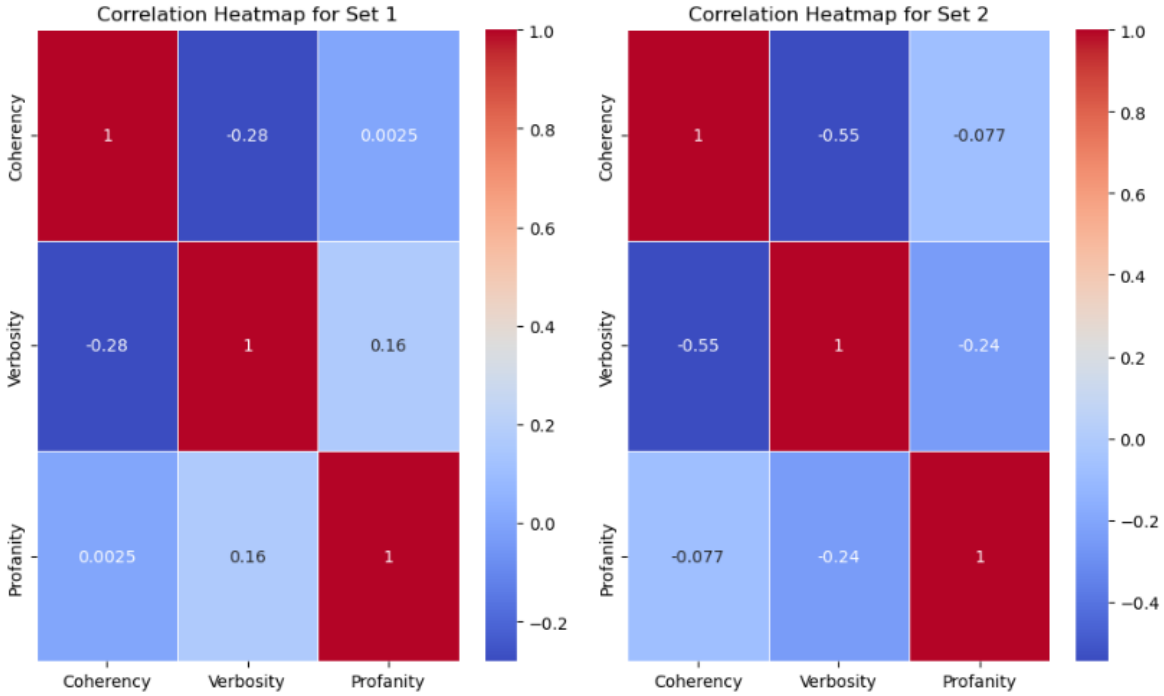

Figure 7.2: Metric Correlation Heatmap

Summary: The statistical analysis reveals that the LFT_Model_1 generally exhibits slightly higher coherency and lower verbosity compared to the Baseline Model, while profanity levels remain similar between the two. Both models demonstrate high average coherency, with most jokes perceived as coherent. In terms of correlations, the LFT_Model_1 shows a weak negative correlation between coherency and verbosity, and no significant correlation between coherency and

profanity. Conversely, the Baseline Model exhibits a moderate negative correlation between coherency and verbosity. These findings are visually confirmed by the correlation heatmap. Overall, while both models produce coherent jokes, the Baseline Model's higher verbosity may have a more pronounced impact on coherency. Profanity does not seem to significantly affect coherency in either model, suggesting potential areas for optimizing joke generation models to prioritize coherency and humour quality.

Statistical Analysis

The statistical analysis was conducted to investigate the relationship between various linguistic factors, specifically profanity and verbosity, and the perceived coherency of jokes. The primary hypothesis under scrutiny was whether coherency is influenced by the presence of profanity and the verbosity of jokes.

It was hypothesized that an increase in profanity within jokes may lead to a decrease in perceived coherency, as profanity could potentially disrupt the flow of the joke or detract from its overall coherence. Similarly, the hypothesis regarding verbosity posited that longer jokes, characterized by higher verbosity, might be perceived as less coherent due to their potential to introduce unnecessary complexity or tangential elements.

Furthermore, the analysis aimed to assess whether jokes generated by the LFT_Model_1 exhibited superior performance in terms of humour compared to jokes generated by the baseline model. This hypothesis stemmed from the assumption that different joke generation models might produce varying levels of humour, with one model potentially outperforming the other in eliciting laughter or amusement from the audience.

By statistically examining the relationships between profanity, verbosity, and coherency, as well as comparing the humour outcomes between the LFT_Model_1 and the baseline model, this analysis sought to provide empirical insights into the linguistic factors influencing joke perception and generation.

LFT_Model_1: The linear regression model for LFT_Model_1 is:

$$Coherency_{LFT\_Model\_1} = \beta_0 + \beta_1 \cdot Verbosity_{LFT\_Model\_1} + \beta_2 \cdot Profanity_{LFT\_Model\_1} + \epsilon_{LFT\_Model\_1}$$

where:

- $Coherency_{LFT\_Model\_1}$ is the coherence score for jokes in LFT_Model_1 ,
- $Verbosity_{LFT\_Model\_1}$ is the verbosity score for jokes in LFT_Model_1 ,
- $Profanity_{LFT\_Model\_1}$ is the profanity score for jokes in LFT_Model_1 ,
- β_0, β_1, and β_2 are the regression coefficients,
- $\epsilon_{LFT\_Model\_1}$ is the error term.

The regression analysis suggests that neither verbosity nor profanity significantly predicts coherency in LFT_Model_1 .

Baseline Model: Similarly, the linear regression model for the Baseline Model is:

$$Coherency_{BaselineModel} = \beta_0' + \beta_1' \cdot Verbosity_{BaselineModel} + \beta_2' \cdot Profanity_{BaselineModel} + \epsilon_{BaselineModel}$$

A significant negative coefficient for β_1' suggests that verbosity significantly predicts lower coherency in the Baseline Model.

Correlation Analysis

- Correlation coefficient between Coherency and Verbosity:
  - LFT_Model_1: $r_{LFT\_Model\_1}(Coherency, Verbosity) = -0.2808$
  - Baseline Model $r_{BaselineModel}(Coherency, Verbosity) = -0.5459$
- Correlation coefficient between Coherency and Profanity:
  - LFT_Model_1 : $r_{LFT\_Model\_1}(Coherency, Profanity) = 0.0025$
  - Baseline Model: $r_{BaselineModel}(Coherency, Profanity) = -0.0768$

The negative correlation between coherency and verbosity suggests that longer jokes tend to be perceived as less coherent, particularly in the jokes obtained from the Baseline Model.

Chi-square Test Results

- Chi-square statistic: $\chi^2 = 18.2675$
- Degrees of freedom: $df = 1$
- p-value: $p < 0.05$

There is a significant difference in the proportions of jokes marked as "Funny" between LFT_Model_1 and the Baseline Model and we can safely conclude that LFT_Model_1 outperforms the Baseline Model at generating funnier jokes as there is a statsically strong case for it as shown above.

7.1.4 Conclusion

The statistical analysis provides insights into the relationships between verbosity, profanity, and coherency in jokes from both sets. While verbosity significantly predicts coherency for punchlines for the Baseline Model, neither verbosity nor profanity significantly predicts coherency in LFT_Model_1, indicating more creativity along with the conclusion that the negative correlation between coherency and verbosity suggests that longer jokes tend to be perceived as less coherent. Additionally, the chi-square test confirms a significant difference in the proportions of funny jokes between LFT_Model_1 and the baseline model.

7.2 Qualitative Analysis

For both qualitative experiments, a meticulous approach was employed to ensure the validity and reliability of the findings. Initially, a carefully curated set of 12 setups was extracted from the set of jokes that were deemed funny through quantative analysis as well as ensuring that they were neither inherently offensive nor inappropriate. These setups were then utilized to generate 24 punchlines from LFT_Model_1 and the baseline model, facilitating a comprehensive qualitative analysis. Subsequently, an additional set of 12 setups was selected from the collected data, again ensuring they met the criteria for appropriateness. These setups were then used to generate 12 further punchlines from SFT_Model, allowing for a comparative analysis against LFT_Model_1.

To ensure the integrity of the experiments, two Microsoft Forms were created, with the order of questions and options randomly shuffled to achieve true blind testing. This methodology was instrumental in minimising bias and ensuring that the responses were genuinely reflective of the participants' preferences. The responses were collected from a diverse group of over 114 individuals, including students from UCD and other colleges, as well as 20 professional computer scientists, thereby ensuring a broad representation of respondents.

Finally, respondents were simply presented with a random setup and given, depending on the experiment, either 4 or 5 options:

- Neutral - No model provided a punchline that was better/worse than the other(s)
- Neutral - No model produced a satisfactory punchline
- Model choice - Respondents could selected their preferance for a certain mdoel, if one was found to be funnier over the others.

In addition to the structured survey, word-of-mouth feedback was also collected to gain further insights into the respondents' thoughts. Interestingly, it was observed that respondents with a STEM background took more time to analyze and digest the punchlines, rarely opting for the neutral option. In contrast, respondents outside of STEM fields seemed to prefer shorter punchlines or chose that neither option was funnier. A rough estimate indicated that approximately 60% of the respondents had a background in STEM, while the remaining 40% did not.

7.2.1 Experiment 1: Human vs ChatGPT vs Fine-tuned Model Performances

Experiment Setup:

Verifying that Fine-Tuning a model can outperform ChatGPT's baseline and be on par with human humour[2] 1

Setup	Question
Setup 1	Why was six afraid of 7?
Setup 2	Why did the chicken cross the road?
Setup 3	Did it hurt when you fell from Heaven?
Setup 4	Did you blow bubbles as a kid?
Setup 5	What's brown and sticky?
Setup 6	Is your refrigerator running?
Setup 7	Are you russian?
Setup 8	How do you know when you have bad acne?
Setup 9	What's the slowest vehicle on the road?
Setup 10	Is there any Indian left?
Setup 11	Why does Eminem serve weak coffee?
Setup 12	Why did Anakin Skywalker join the dark side?

Table 7.5: Setups

Experiment Dataset:

Setup	GPT - Baseline	Reddit	Finetuned Model
1	Because seven "ate" nine!	Because 7 is a registered six offender	Because 7 is a "prime" suspect
2	To get away from all of the bad egg yolks in its life.	To fight Peter Griffin	To escape KFC
3	Because I heard they have strict immigration laws up there.	Because it really looks like it did	I can tell.
4	Well, nowadays the only bubbles I blow are in my champagne.	Well, he gets out of jail next month.	I heard he's looking for you.
5	A stick!	The contents of my pants	Brown paint
6	Because if it is, you better go catch it!	Because it smells like feet	For presidency? Really?
7	Because you're always Putin a smile on my face.	No, I'm not in a hurry.	Because you're always Stalin in bed.
8	When your pillowcase is more colorful than your wardrobe.	When the blind try to read your face.	When Proactive sends you a thank you card for their #1 customer
9	The driver in front of me during rush hour.	The short bus	A hearse
10	Yes, I haven't finished my plate yet.	No, there's naan left.	Yes, I think I hear some curry-ing in the kitchen!
11	Because he's afraid of the "Real Slim (Bean)dy	You only get one shot	He likes it with a lot of vanilla ice
12	He needed some space from Obi-Wan.	Because he needed a hand.	He got burnt one too many times by the Jedi

Table 7.6: Punchlines per model

Results:

Setup	Reddit %	LFT_Model %	GPT %	Not funny	Nothing is better/worse	Total
1	54	20	23	4	10	111
2	23	32	14	33	9	111
3	35	19	37	14	6	111
4	29	33	18	23	8	111
5	40	30	12	22	7	111
6	12	45	24	19	11	111
7	31	44	26	5	5	111
8	65	12	15	8	11	111
9	11	40	17	29	14	111
10	69	15	8	13	6	111
11	70	11	13	12	5	111
12	31	18	16	31	15	111
Total:	470	319	223	213	107	1332

Table 7.7: Comparison of Responses

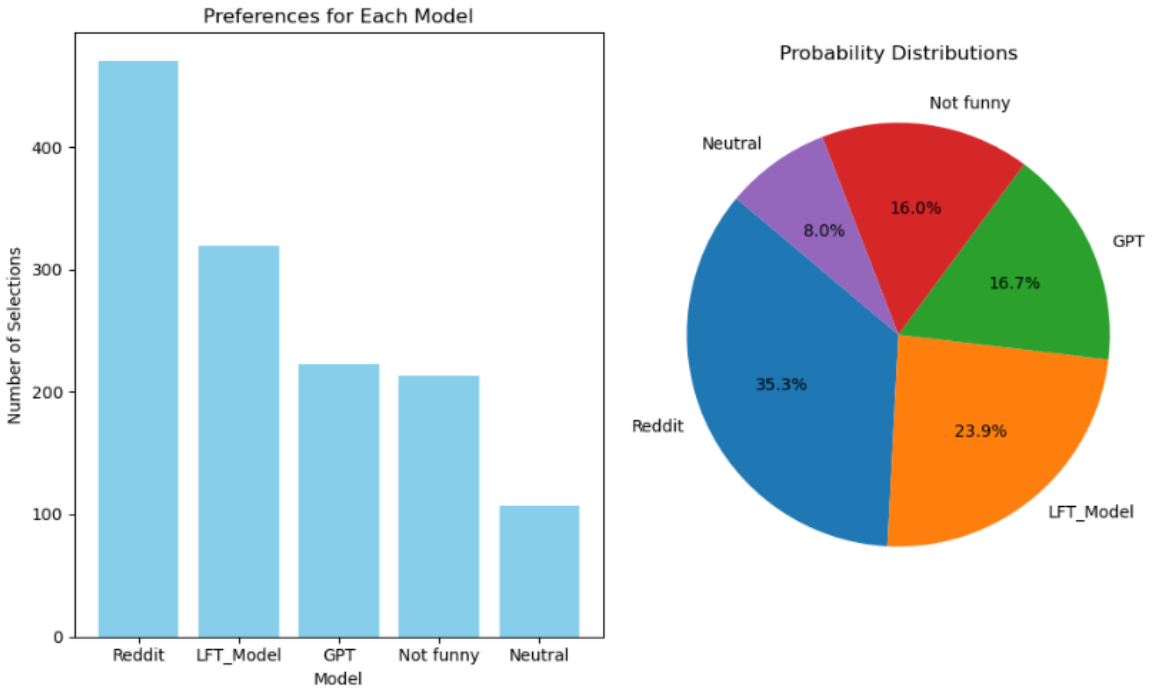

Figure 7.3: Results for Experiment 1

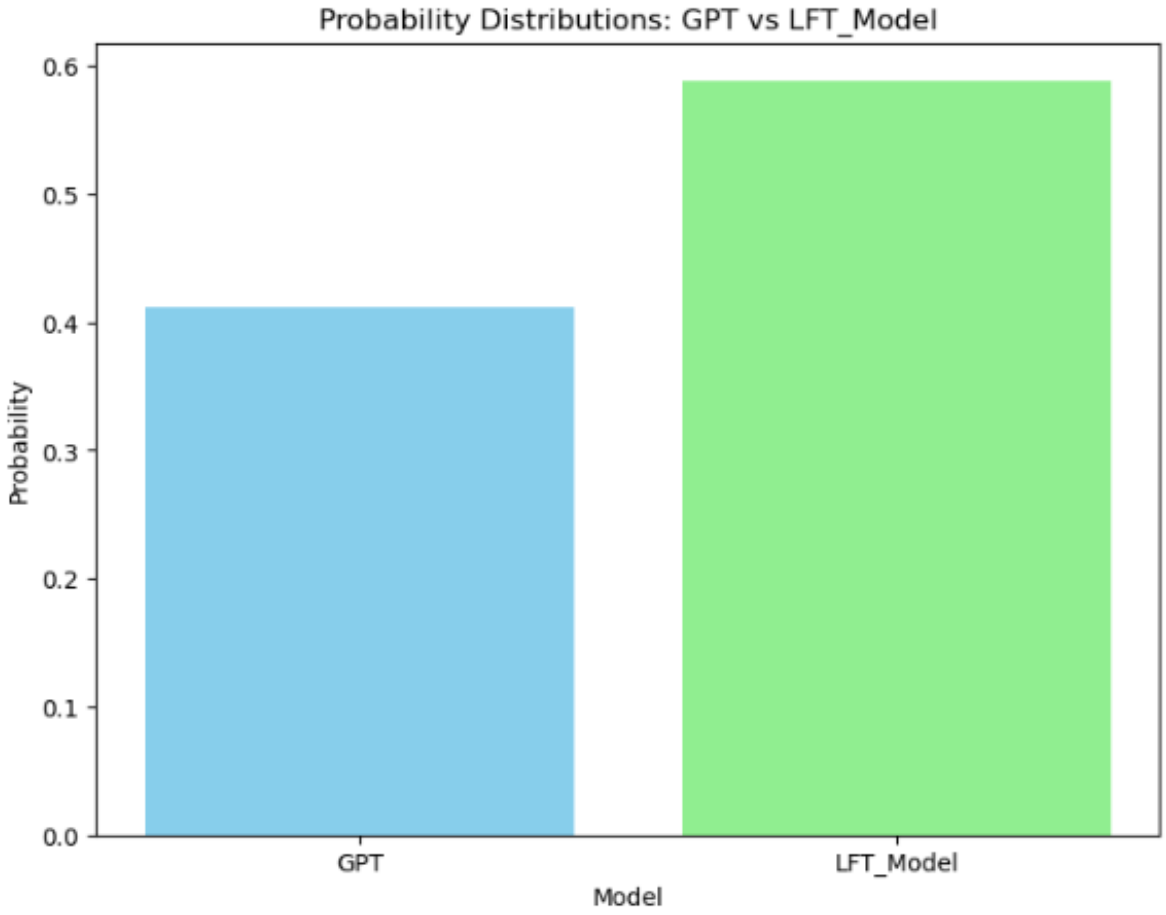

Figure 7.4: Probability Distributions for Experiment 1

To determine whether the observed difference in preferences between the LFT_Model and GPT is statistically significant, we perform a binomial test. Let n be the total number of selections (542), x be the number of selections of the LFT_Model (319), and p_0 be the null hypothesis probability of preference for the LFT_Model, which is set to 0.5 (no difference).

The test statistic for the binomial test is given by:

$$Z = \frac{x - np_0}{\sqrt{np_0(1-p_0)}}$$

The p-value is calculated using the cumulative distribution function of the standard normal distribution:

$$p - value = P(Z > z)$$

where z is the calculated test statistic.

Substituting the values, we find:

$$Z = \frac{319 - 542 \times 0.5}{\sqrt{542 \times 0.5 \times (1-0.5)}} \approx \frac{-7}{\sqrt{135.5}} \approx -2.1519464398057077e-05$$

Using a standard normal distribution table, we find that the p-value is approximately 0.0000215.

Since the p-value is less than the chosen significance level (e.g., 0.05), we reject the null hypothesis and conclude that there is strong evidence to suggest a statistically significant difference in preferences between the LFT_Model and GPT, and we can confidently say that LFT_Model was preffered over the GPT baseline model, and subsequently outperformed the baseline model.

7.2.2 Experiment 2: Dataset size/Epoch Model Performances

Experiment Setup:

Analysing effects of a large vs small dataset and more/less epochs on the fine-tuned model[3] 1

Setup	Question
Setup 1	Did you hear about the car that was sent to therapy?
Setup 2	Why did the actor break up with their partner?
Setup 3	Why did the rock band break up?
Setup 4	Why did the art student never have any money?
Setup 5	What's a scientist's favorite planet?
Setup 6	Why don't football players ever text you back?
Setup 7	What makes a bad driver?
Setup 8	Did you know I'm scared of origami?
Setup 9	What do you call two witches living together?
Setup 10	What did one book say to the other book?
Setup 11	Why are bodybuilders bad gamblers?
Setup 12	Why was Batman always calm?

Table 7.8: Setups

Experiment Dataset:

Setup 1	2epoch	5epoch
Setup 1	It had metal health issues	It had a lot of breakdowns
Setup 2	They had no chemistry	They couldn't handle the drama
Setup 3	They became sedimental	They took each other for granite
Setup 4	She were always doing something sketchy	She had no Monet
Setup 5	Plut-oh-wait...	Definitely not Uranus...
Setup 6	They prefer the huddle	They tend to step on their receivers
Setup 7	Alcohol	Ask Paul Walker
Setup 8	I get very easily creased.	I tend to fold under pressure
Setup 9	Broom-mates	What's your sisters names again?
Setup 10	Don't worry, I've got you covered	Boy do I have a story for you!
Setup 11	They're afraid of losing their gains	Because of their muscle memory
Setup 12	Nothing gets under his skin	Because Dick wasn't 'Robin' anymore

Table 7.9: Jokes and Punchlines

Results:

Setup	2 epoch Model %	5 Epoch Model %	Not funny	Neither funnier/worse	Total
1	16	55	4	4	79
2	7	50	15	7	79
3	15	47	9	8	79
4	26	34	6	13	79
5	30	31	11	7	79
6	21	24	26	8	79
7	11	41	17	10	79
8	10	54	11	4	79
9	34	38	4	3	79
10	24	40	8	7	79
11	52	15	10	2	79
12	11	30	33	5	79
Total:	257	459	154	78	948

Table 7.10: Epoch Model Comparison

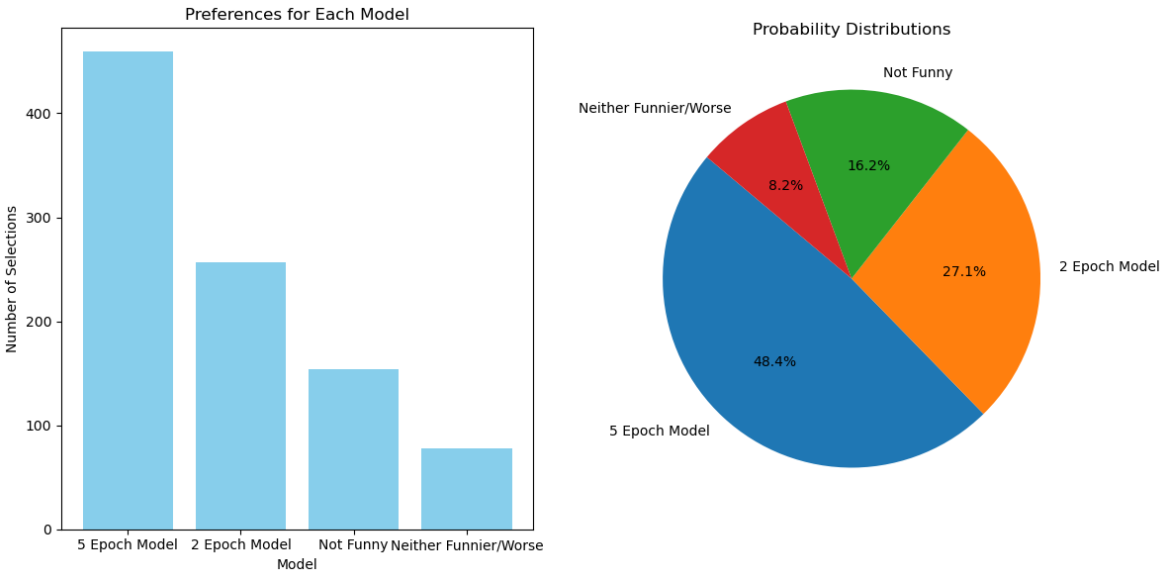

Figure 7.5: Results for Experiment 2

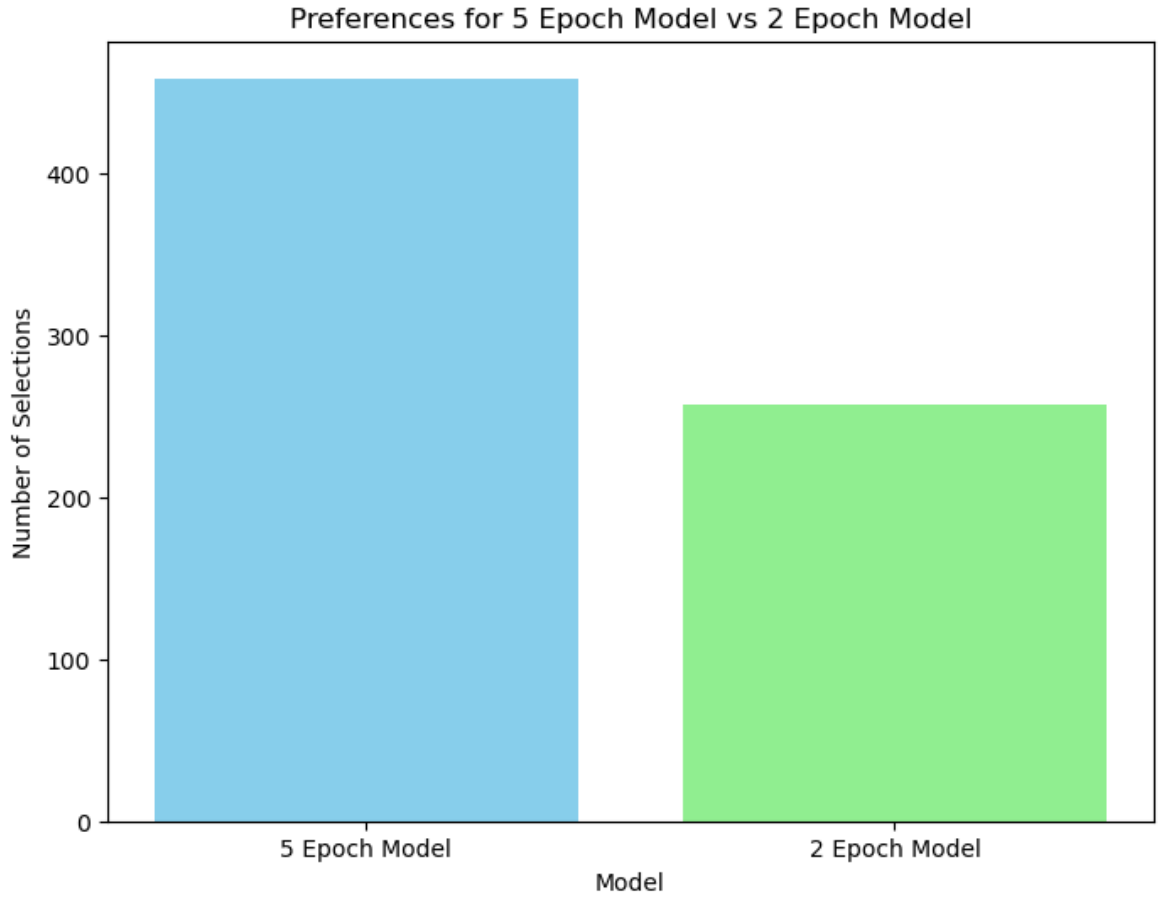

Figure 7.6: Probability Distributions for Experiment 2

To determine whether the observed difference in preferences between the 5epoch model and 2epoch model is statistically significant, we perform a binomial test. Let n be the total number of selections (716), x be the number of selections of the 5epoch model (459), and p_0 be the null hypothesis probability of preference for the 5epoch model, which is set to 0.5 (no difference).

The test statistic for the binomial test is given by:

$$Z = \frac{x - np_0}{\sqrt{np_0(1-p_0)}}$$

The p-value is calculated using the cumulative distribution function of the standard normal distribution:

$$p - value = P(Z > z)$$

where z is the calculated test statistic.

Substituting the values, we find:

$$Z = \frac{459 - 716 \times 0.5}{\sqrt{716 \times 0.5 \times (1 - 0.5)}} \approx \frac{-257}{\sqrt{179}} \approx \ 1.9904947789421935e - 14$$

Using a standard normal distribution table or software, we find that the p-value is extremely small (close to 0).

Since the p-value is less than the chosen significance level (e.g., 0.05), we reject the null hypothesis and conclude that there is strong evidence to suggest a statistically significant difference in preferences between the 5epoch model and 2epoch model, and we can confidently say that 5epoch model was preferred over the 2epoch model , and subsequently outperformed the 5epoch model.

7.3 Overtraining

The overtrained model yielded less than satisfactory results, in line with the intended purpose of testing its limitations. The model consistently lacked nuances in its jokes and often resorted to repeating the same jokes, indicating a lack of diversity and creativity. These limitations underscore the importance of appropriate model training and validation techniques to ensure the production of novel and engaging content.

For example, the randomly acquired setup "Why did Anakin leave the Jedi?" provides a fairly open-endedness, allowing a model to lean towards punchlines about him becoming Darth Vader, the fall of the Jedi, his need for power along with many other nuances, which can be seen in the fine-tuned modes:

In this case, the baseline model consistently produced punchlines with more wit and humour compared to LFT_Model_2. The punchlines generated by LFT_Model_2 lacked the creativity and coherence found in those produced by the baseline model. This stark contrast underscores the potential risks of overtraining a model, as it can lead to a significant deterioration in the quality and originality of generated content.

Table 7.11: Setup: **"Why did Anakin leave the Jedi?"**

Model	**Punchlines**
baseline_model	He wanted to avoid paying the high light-saber maintenance fees! He needed to vent his frustration and anger... but the Dark Side had a better benefits packa He just couldn't handle the lightsaber development anymore. He was tired of being a Skywalker-walker. He wanted to explore his dark side career opportunities.
LFT_Model_1	He thought it was a Sithy organization He just couldn't take their puns. He thought they were holding him back. He couldn't take his master's sarcasm. He thought it was an April Fool's joke.
SFT_Model	He thought it was a Sithy organization He just couldn't take their puns. He thought they were holding him back. He couldn't take his master's sarcasm. He thought it was an April Fool's joke.
LFT_Model_2	He just wasn't cutting it. He just wasn't cutting it. He just wasn't cut out for it. He just couldn't take any more Sith. He got burned out.

Chapter 8: **Conclusions**

This final chapter focuses on presenting the conclusions expanding from the background research to design and implementation, up to the summary of results. The following sections is a recap of the work made that is used to establish conclusions of the project, critical analysis, lessons learned and future extensions to the project.

8.1 Summary

This project undertook a comprehensive study of humour, focusing on various interventions to study and analyse the limitations as well as the current comedic capabilities of current Large Language Models such as ChatGPT with the end goal of creating a "Large Laughter Model" capable of producing original, comedic jokes. The report progresses from a literature review to a comparative analysis of fine-tuning and prompt engineering. It begins by reviewing relevant literature, with an emphasis on the issues and limitations on current LLMs as well as a focus on solving these issues by addressing humour as a computational problem rather than a linguistic one.

Following the literature review, the report details the explicit high-level implementations of the three models which were fine-tuned from the process of cleaning the dataset and formatting a JSONL file to the collection and analysis of the gathered jokes.

Through both quantitative and qualitative analysis as well as performance analysis, it's evident that ChatGPT does have the capability to produce a "Large Laughter Model" which can be on par with human capabilities as well as outperforming its baseline chatbot model through the method of fine-tuning. As found by re-training LFT_Model_1 to create LFT_Model_2 using the smaller dataset used to create SFT_Model to investigate the issues of over-training, the current baseline model has been over trained on large datasets as the produced jokes were of the same quality, with both models tending to repeat jokes without nuances, while the LFT_Model_1 and SFT_Model models vastly outperformed them, showing great nuances within creativity, more punchy jokes as well as edgier jokes without being inappropriate.

It is interesting to note that Thorp's notion of ChatGPT being "fun but not an author" [43] proved true as while trying to obtain topics and setups for jokes, which admittedly is difficult for even a trained professional joke writer to come up with on the fly, the only model which gave consistent nuances in its replies was the SFT_Model, as the vast majority of the responses generated weren't original nor coherent. Additionally, as investigated in the work by Jentzsch and Kersting [1] , I found repeated jokes and topics to be present within my collected dataset, with a emphasis on "why did x/y/z go to therapy/cheat on their partner?" to be a recurring theme.

In summary, the fine-tuned models showed great promise, albeit not 100% satisfactory, and vastly outperformed the prompt-engineered method as well as being almost equally preferred by a test audience when presented with a blind sample of jokes.

8.2 Future Work

This project has a couple of limitations that could be addressed by future endeavours in this field:

- One primary concern is the personality of the LLM model; while the delivery and punchiness of the punchlines are respectable, there is room for improvement. OpenAI's models such as GPT4, once fine-tuning becomes a possibility, could potentially exhibit a higher ability to incorporate edgier or more topical themes into jokes, balancing between politically correct humour and more lacklustre or verbose jokes.
- Consequently, once GPT4 is available for fine-tuning, there is an opportunity to extend the evaluation metrics as investigated in the study conducted by F. Goes et al [6]. Implementing a joke evaluator using a fine-tuned GPT-4 model could assess and compare the evaluatory abilities of an LLM and humans regarding the ability to analyze and identify funny jokes compared to bad ones.
- As mentioned above in 5.1.1, incorporating upvotes as weights to produce more "topical/relevant" jokes preferred by the larger population could be invaluable.
- Finally, when made available through OpenAI, investigating the enabling of fine-tuned LLMs to create more topical jokes based on current events, with moderation for edgier content, could be a worthwhile endeavour to fully understand their capabilities.

These enhancements could lead to more nuanced and engaging joke generation systems that better resonate with users and reflect the evolving nature of humour.

My proposition is to use a weighted trained LLM and incorporate prompt-engineering interventions to allow a "chat" with the fine-tuned model in order to teach it more nuances and a more free flowing sense of what kind of jokes are funny, to learn a sense of "delivery" as well as to experiment with the ability of an LLM to incorporate more edgier elements into a politically correct joke once made possible by OpenAI

Bibliography

1. Jentzsch, S. & Kersting, K. *ChatGPT is fun, but it is not funny! Humor is still challenging Large Language Models* 2023. arXiv: 2306.04563 [cs.AI]. https://aclanthology.org/2023.wassa-1.29.pdf.
2. Gupta, A. *Survey on Performance of Fine-tuned model, ChatGPT and Orginal Human Dataset* Verifying that Fine-Tuning a model can outperform ChatGPT's baseline and be on par with human humour. 2024. https://forms.office.com/Pages/DesignPageV2.aspx?subpage=design&FormId=icUOQmao0EqaV-YEng07wIXTZFxHYlxPnYscdi5GuTpUQkJGRDhBVU4zWlJUTUNH
Token=4937d63827344750b271adef33c601e5.
3. Gupta, A. *Survey on Large and Small Epoch Models' Performances* Analysising effects of a large vs small dataset and more/less epochs on the fine-tuned model. 2024. https://forms.office.com/Pages/DesignPageV2.aspx?subpage=design&FormId=icUOQmao0EqaV-YEng07wIXTZFxHYlxPnYscdi5GuTpUN1VYOUI5SFlVTEdLRTVRSDBDR0xVT0M2OS4u&Token=5740787a1cd34fb1b4ccc971397288c9.
4. Kanuck, S. Humor, Ethics, and Dignity: Being Human in the Age of Artificial Intelligence. *Ethics & International Affairs* **33,** 3–12. https://sci-hub.se/10.1017/s0892679418000928 (2019).
5. Veale, T. *Does Not Compute! Does Not Compute! The Hows and Whys of Giving AIs a Sense of Humour* in *Proceedings of the 14th Conference on Creativity and Cognition* (Association for Computing Machinery, Venice, Italy, 2022), 1. ISBN: 9781450393270. https://doi.org/10.1145/3527927.3534960.
6. Fabricio Goes, Z. Z., Sawicki, P., Grześ, M., Brown, D. & Volpe, M. *Is GPT-4 good enough to evaluate jokes?* in *14th International Conference for Computational Creativity* (Waterloo, Canada, May 2023). https://kar.kent.ac.uk/101552/.
7. Goes, F., Zhou, Z., Sawicki, P., Grześ, M. & Brown, D. *Crowd score: a method for the evaluation of jokes using Large Language Model AI voters as judges* Dec. 2022. https://kar.kent.ac.uk/101553/.
8. Hempelmann, C., Raskin, V. & Triezenberg, K. *Computer, Tell Me a Joke ... but Please Make it Funny: Computational Humor with Ontological Semantics.* in. **13** (Jan. 2006), 746–751.
9. Paulos, J. A. *Mathematics and humor* (University of Chicago Press, 2008).
10. Ravi, S. *et al.* Small But Funny: A Feedback-Driven Approach to Humor Distillation. arXiv: 2402.18113 [cs.CL]. https://arxiv.org/abs/2402.18113 (2024).
11. Ritchie, G. *Prospects for Computational Humour* in *Proceedings of 7th IEEE international workshop on robot and human communication* (1998), 283–291.
12. Cundall, M. K. & Macagno, F. What Makes a Joke Bad: Enthymemes and the Pragmatics of Humor. *The Philosophy of Humor Yearbook* **4,** 111–129. https://doi.org/10.1515/phhumyb-2023-0005 (2023).
13. Dean, G. & Hammock, L. *Why Are Jokes Funny? Take-aways from Greg Dean's Step by Step Stand-Up Comedy* https://www.marblejar.net/2018/01/why-are-jokes-funny-take-aways-from.html. Accessed: 2024-04-28. 2018.
14. Toplyn, J. *Witscript: A System for Generating Improvised Jokes in a Conversation* 2023. arXiv: 2302.02008 [cs.CL].
15. Toplyn, J. *Witscript 2: A System for Generating Improvised Jokes Without Wordplay* 2023. arXiv: 2302.03036 [cs.CL].

16. Toplyn, J. *Witscript 3: A Hybrid AI System for Improvising Jokes in a Conversation* 2023. arXiv: 2301.02695 [cs.CL].
17. Chen, Y., Shi, B. & Si, M. Prompt to GPT-3: Step-by-Step Thinking Instructions for Humor Generation. *arXiv preprint arXiv:2306.13195* (2023).
18. Smith, J. & Johnson, J. Prompt Engineering for Humor Generation: A Study on Joe Toplyn's Techniques. *Journal of AI and Natural Language Processing* (2022).
19. Lee, A. & Kim, B. Witscript: A Language Model for Humor Generation. *Journal of AI and Natural Language Processing* (2023).
20. Lee, A. & Kim, B. The Role of Witscript in Enhancing AI-Generated Humor. *Journal of AI and Natural Language Processing* (2023).
21. Toplyn, J. US11080485B2. https://patents.google.com/patent/US11080485B2/en (2021).
22. Toplyn, J. US11080485B2. https://patents.google.com/patent/US11080485B2/en (2020).
23. Toplyn, J. US11080485B2. https://patents.google.com/patent/US11080485B2/en (2019).
24. Fthilton, N. *Grok, AI and the humour test: would Douglas Adams find your LLM funny?* https://nickfthilton.medium.com/grok-ai-and-the-humour-test-would-douglas-adams-find-your-llm-funny-68c1b8721fcb. Accessed: 2024-03-27. 2023.
25. Attardo, S. Linguistic Theories of Humor. *Language* **72** (Mar. 1996).
26. Chen, P.-Y. & Soo, V.-W. *Humor Recognition Using Deep Learning* in (Jan. 2018), 113–117.
27. Toplyn, J. E. US20200227032. Filed on March 26, 2020; Assigned to Twenty Lane Media, LLC. https://uspto.report/patent/app/20200227032#D00011(2020).
28. Turano, B. & Strapparava, C. *Making People Laugh like a Pro: Analysing Humor Through Stand-Up Comedy* in *Proceedings of the Thirteenth Language Resources and Evaluation Conference* (European Language Resources Association, Marseille, France, June 2022), 5206–5211. https://aclanthology.org/2022.lrec-1.558.
29. Khodak, M., Saunshi, N. & Vodrahalli, K. A Large Self-Annotated Corpus for Sarcasm (Apr. 2017).
30. u/ProllyDie. *Graph of Laughter in Stand-Up Comedy* Example of a joke for comedic timing delivery using a graph. https://www.reddit.com/r/StandUpComedy/comments/16gq9ah/graph/.
31. Veale, T. The Educational Potential of AI-Generated Humor. *Journal of Humor Studies* (2021).
32. Ray, P. P. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems* **3,** 121–154. ISSN: 2667-3452. https://www.sciencedirect.com/science/article/pii/S266734522300024X (2023).
33. Cain, W. Prompting Change: Exploring Prompt Engineering in Large Language Model AI and Its Potential to Transform Education. *TechTrends* **68,** 47–57. https://doi.org/10.1007/s11528-023-00896-0 (2024).
34. Naveed, H. *et al. A Comprehensive Overview of Large Language Models* 2024. arXiv: 2307.06435 [cs.CL].
35. Joshi, I., Budhiraja, R., Akolekar, H. D., Challa, J. S. & Kumar, D. *"It's not like Jarvis, but it's pretty close!" – Examining ChatGPT's Usage among Undergraduate Students in Computer Science* 2023. arXiv: 2311.09651 [cs.HC].
36. Gorenz, D. & Schwarz, N. How funny is ChatGPT? A comparison of human-and AI-produced jokes (2024).

37. Barattieri di San Pietro, C., Frau, F., Mangiaterra, V. & Bambini, V. The pragmatic profile of ChatGPT: Assessing the communicative skills of a conversational agent. *Sistemi intelligenti* **35,** 379–400 (2023).

38. Gómez-Rodríguez, C. & Williams, P. A confederacy of models: A comprehensive evaluation of LLMs on creative writing. *arXiv preprint arXiv:2310.08433* (2023).

39. Holl, C. The content intelligence: an argument against the lethality of artificial intelligence. *Discov Artif Intell* **4.** https://doi.org/10.1007/s44163-024-00112-9 (2024).

40. Technica, A. *Researchers discover that ChatGPT prefers repeating 25 jokes over and over* https://arstechnica.com/information-technology/2023/06/researchers-discover-that-chatgpt-prefers-repeating-25-jokes-over-and-over/. Accessed: 2024-03-27. 2023.

41. Li, C., Chen, M., Wang, J., Sitaram, S. & Xie, X. CultureLLM: Incorporating Cultural Differences into Large Language Models. arXiv: 2402.10946 [cs.CL]. https://arxiv.org/abs/2402.10946 (2024).

42. Popova, O. & Dadić, P. *Does AI Have a Sense of Humor? CLEF 2023 JOKER Tasks 1, 2 and 3: Using BLOOM, GPT, SimpleT5, and More for Pun Detection, Location, Interpretation and Translation* in *Automatic Wordplay Analysis (JOKER)* (2023), 1888–1908. https://ceur-ws.org/Vol-3497/.

43. Thorp, H. H. ChatGPT is fun, but not an author. *Science* **379,** 313–313. eprint: https://www.science.org/doi/pdf/10.1126/science.adg7879. https://www.science.org/doi/abs/10.1126/science.adg7879 (2023).

44. Koco'n, J. *et al.* ChatGPT: Jack of all trades, master of none. *Information Fusion* **99,** 101861. https://api.semanticscholar.org/CorpusID:257050407 (2023).

45. Veale, T. The Creativity of ChatGPT: A Critical Review. *Journal of Humor Studies* (2021).

46. Lin, Z. Techniques for supercharging academic writing with generative AI. *Nature Biomedical Engineering* **2157.** https://doi.org/10.1038/s41551-024-01185-8 (2024).

47. Weller, O., Fulda, N. & Seppi, K. *Can humor prediction datasets be used for humor generation? humorous headline generation via style transfer* in *Proceedings of the Second Workshop on Figurative Language Processing* (2020), 186–191.

48. Veale, T. *Your Wit is My Command* ISBN: 9780262045995 (THE MIT PRESS, 2021).

49. Toplyn, J. The Structure of Humor: A Review of Joe Toplyn's Algorithm. *Journal of AI and Natural Language Processing* (2021).

50. Wang, C. *et al. No More Fine-Tuning? An Experimental Evaluation of Prompt Tuning in Code Intelligence* in *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Association for Computing Machinery, <conf-loc>, <city>Singapore</city>, <country>Singapore</country>, </conf-loc>, 2022), 382–394. ISBN: 9781450394130. https://doi.org/10.1145/3540250.3549113.

51. Junprung, E. *Exploring the Intersection of Large Language Models and Agent-Based Modeling via Prompt Engineering* 2023. arXiv: 2308.07411 [cs.AI].

52. Zhao, W. X. *et al. A Survey of Large Language Models* 2023. arXiv: 2303.18223 [cs.CL].

53. Elizondo, R. *Jokes at Scale Using GPT-2 and AWS* https://medium.com/dataroots/jokes-at-scale-using-gpt2-and-aws-3ef488361a34.

54. Petrović, S. & Matthews, D. Unsupervised joke generation from big data. **2,** 228–232 (Jan. 2013).

55. Beigh, A. & Sergeev, I. *profanity-filter* https://pypi.org/project/profanity-filter/. Version 1.3.3, released on April 30, 2020. Supports English and Russian profanity filtering. Includes deep analysis, multilingual support, and extensibility for adding new languages. Licensed under GNU General Public License v3 or later (GPLv3+). 2020.

56. Mistriotis, D. *alt-profanity-check* https://pypi.org/project/alt-profanity-check/. Version 1.4.2. 2024.

57. OpenAI. *Fine-tuning Guide* Accessed: 2024-04-22. 2024. https://platform.openai.com/docs/guides/fine-tuning/preparing-your-dataset.

58. Bird, S., Loper, E. & Klein, E. *Natural Language Processing with Python* (O'Reilly Media Inc., 2009).