

# Automatic Generation and Detection of Motivational-Interviewing-Style Reflections for Smoking Cessation Therapeutic Conversations Using Transformer-based Language Models

by

Imtihan Ahmed

A thesis submitted in conformity with the requirements  
for the degree of Master of Applied Science  
Graduate Department of Electrical and Computer Engineering  
University of Toronto

© Copyright by Imtihan Ahmed 2022

# Automatic Generation and Detection of Motivational-Interviewing-Style Reflections for Smoking Cessation Therapeutic Conversations Using Transformer-based Language Models

Imtihan Ahmed

Master of Applied Science

Graduate Department of Electrical and Computer Engineering  
University of Toronto

2022

## Abstract

If conversational agents can take on a therapeutic role, they may provide a scalable way to help many more people suffering from addictions, where and when they need it. Motivational Interviewing (MI) is a validated therapy for behaviour change that has been applied to many behaviours, including helping smokers to quit. A key technique in MI is to ask open-ended questions concerning a negative behaviour, and then provide a reflection of the response. Reflections can be simple restatements of the response, or more complex inferences drawing on prior statements or general knowledge. This thesis presents a method of generating simple and complex reflections with GPT-2 and GPT-3, transformer-based generative language models, from open-ended prompts and free-form responses. The models are used both in few-shot mode and are fine-tuned. We show that the few-shot approach can produce very promising simple and complex reflections even in low-data environments and is significantly improved by fine-tuning. However, some of the generated reflections are poor – either off-topic or contradictory to the therapeutic goal. To address these reflections, we developed a classifier that can determine if the reflection meets an acceptable level of quality and is used to filter poor quality generated reflections. The classifier is trained using labels provided by an MI expert, on a corpus of questions, responses, and

reflections obtained from trials of a previous smoking cessation chatbot. It achieved an accuracy of 81%, a sensitivity of 90%, and a specificity of 71%. We show that with just the few-shot learning method, GPT-2 produces acceptable reflections with a 54% hit rate, and GPT-3 improves that to 89%. Combining GPT-2 with the classifier as a filter, the system achieves a 73% hit rate of the time. The fine-tuned GPT-2 language natively achieves an 80% hit rate, and 81% with the classifier as a filter.

## Acknowledgments

I want to thank my supervisor Prof. Jonathan Rose for his guidance and support over the last two years as I worked on this thesis. Our weekly discussions have helped me grow as a researcher, and more importantly, as a person. I also want to thank Dr. Peter Selby and the rest of the CAMH team involved in this project for their valuable expertise on Motivational Interviewing, without which this project would not be possible. I am grateful for my peers, and the summer students with whom I was able to work with, for their intellectual support and for the much necessary coffee runs.

I would also like to thank my life partner Emma Mannan, friends, and my parents for their continued contribution to my existence.

# Table of Contents

Acknowledgments.....	iv
Table of Contents .....	v
List of Tables .....	vii
List of Figures .....	viii
Chapter 1 Introduction .....	1
1.1 Automating Mental Health .....	1
1.2 Focus and Goals .....	3
1.3 Contributions.....	3
1.4 Organization.....	3
Chapter 2 Background and Related Work .....	5
2.1 Reflections and MI.....	5
2.2 NLP and Language Models .....	6
2.2.1 Generative Models .....	6
2.2.2 Text Sequence Classification Models.....	10
2.3 Related Work on Chatbots for MI .....	12
Chapter 3 Dataset and Therapeutic Context .....	14
3.1 Overview of Required Data .....	14
3.2 Source Dataset .....	14
3.3 Enhancing Dataset for Few-Shot Reflection Generation & Classification.....	15
Chapter 4 Few-Shot Reflection Generation Methodology .....	17
4.1 Few-Shot Learning With GPT-2 and GPT-3 .....	17
4.1.1 Choosing the Number of Primers .....	19
4.1.2 Exploring Decoding Parameters for Generation.....	20
4.2 Fine-Tuning GPT-2 For Reflection Generation.....	22
Chapter 5 Automatically Classifying Reflections.....	24

5.1 Reflection Quality Classifier (RQC) Description .....	24
5.2 Training the RQC.....	25
5.3 Automatic Reflection Labeling with the RQC.....	27
Chapter 6 Generating Reflections .....	28
6.1 Generation Set-up .....	28
6.2 Reflection Generation Results .....	29
6.2.1 Few-Shot Reflection Generation.....	29
6.2.2 GPT-2 Fine-tuned Reflection Generation.....	30
6.2.3 Generating Reflections with GPT-2 followed by an RQC Filter.....	31
6.3 Qualitative Discussion of Reflection Generated Reflections.....	34
Chapter 7 Deploying a Reflection Generation System .....	40
7.1 Reflection Generation Microservice .....	40
7.2 Reflection Generation Microservice .....	43
Chapter 8 Conclusions, Limitations, and Future Work .....	45
8.1 Conclusions.....	45
8.2 Limitations .....	45
8.3 Future Work .....	46
Bibliography .....	47

## List of Tables

Table 2.1: Examples of Simple and Complex Reflections .....	6
Table 2.2: GPT-2 Decoding Strategies .....	10
Table 5.1: RQC Model Accuracy, Sensitivity and Specificity Pre- and Post- Fine-Tuning .....	26
Table 5.2: RQC BERT-base MSRP+Reflection Fine-tuned Result Counts .....	27
Table 5.3: Inter-rater Correlation Between Hand-Labeling and RQC.....	27
Table 6.1: Reflection Hit Rate (for 100 trials) by Model .....	29
Table 6.2: Reflection Hit Rate (for 100 trials) for Fine-Tuned Models.....	31
Table 6.3: GPT-2 Reflection Generation Rate with and without Filtering.....	33
Table 6.4: Acceptable and Better Reflection Examples .....	35
Table 6.5: Not Acceptable Reflection Examples .....	39

## List of Figures

Figure 2.1: GPT-2 Model Architecture.....	7
Figure 2.2: GPT-2 Model Architecture.....	7
Figure 4.1: Example Few-Shot Input Sequence .....	19
Figure 4.2: Reflection Generation Hit Rate for Different Number of Primers for GPT-2 .....	20
Figure 4.3: Reflection Hit Rate at Different Decoding Parameter Combinations for GPT-2.....	21
Figure 4.4: Example GPT-2 Fine-Tuning Datapoint .....	22
Figure 7.1: Reflection Generation API Schema .....	42
Figure 7.2: Example Reflection Generation Application .....	43



# Chapter 1

## Introduction

### 1.1 Automating Mental Health

Mental health disorders are the most underserved of all medical fields, and there are insufficient practitioners to serve the world-wide need [1,2,3]. Therapeutic conversations [4,5,6] form an important part of the treatments available for mental health disorders. If aspects of these conversations could be automated and shown to have therapeutic impact, they could provide far greater access to care than is currently possible by being delivered through any internet portal.

An automated conversational agent that can mimic a psychotherapist is likely far beyond current capabilities, as this task seems equivalent to building a General AI [7,8]. However, some widely used (and validated) therapies have a structure that may lend themselves to automation using natural language processing (NLP) techniques. This structure, together with recent advances in language models that have exhibited an ability to carry a conversation [9,10,11,12] may provide a pathway to creating an effective therapeutic conversation.

This thesis focuses on a specific therapy called Motivational Interviewing (MI) [6] and addresses a specific condition: addiction to smoking. We narrow the focus to the generation of a specific kind of conversational response within MI, called a reflection, that is broadly used as a part of MI therapy but also has application in many other kinds of talk therapy [13,14]. An MI-style conversational reflection has two main types: a simple reflection, which is a re-statement of something that a patient says and is used to encourage the patient to keep talking. A complex reflection makes an inference or provides an insight by connecting the patient's recent words either to something relevant and useful conveyed by the patient in the past, or to a helpful insight from general knowledge. The purpose of reflections in a smoking addiction MI conversation is to help the patient continue to contemplate their smoking behaviour, and to engage in self-reflection. This kind of contemplation can lead to the resolution of the ambivalence that many smokers have towards the smoking habit and help move them towards their decision to quit [6,15]. The automatic generation of appropriate reflections is key to this process, and we believe, could be useful in the many other therapies that also require the patient to engage in self-reflection [16,13].

There have been several previous attempts to create automated MI conversations [17,18,19,20,21,22,23,24]. However, in most cases the reflections that are presented to the subjects are pre-written by humans, and the choice of reflection is based on either an automated classification of the subject's input [17,24] or through topic modeling based on keywords [21]. Using pre-written responses makes the bot behave predictably but makes it difficult to reflect on specific ideas the subject brings up in the conversation. These reflections are often shallow and broadly applicable, reducing their therapeutic effect compared to more personalized reflections. In this work we seek a way to create reflections that are generated directly from what the subjects say, rather than human-scripted reflections. This method might enable automated MI systems to have multi-turn conversations with responses that are unique to each person, rather than selecting from a limited list of scripted and therefore more generic responses. This work also demonstrates a method to deploy such a reflection generation system using off-the-shelf tools from cloud providers for use in downstream applications (for example an MI chatbot web application) by deploying the system using AWS [25]. Overall, this work most closely relates to Shen et. al's [26] work on automatically generating reflections, which we describe in more detail in Chapter 2.

MI conversations begin with open-ended questions from a counselor which produce free-form responses from patients. The goal of this work is to generate an appropriate reflection, automatically, given an input that is the question together with the response. We are partly motivated by recent progress in language models has exhibited remarkable conversational fluency in the generation of text [8,27,9,11,10,12]. The few-shot approach to language tasks that was suggested in [9] may allow such models to be used to produce good reflections when conditioned with only a few examples of good reflections. In this work, we show that both OpenAI's GPT-2 [12] and GPT-3 [9] language models produce good simple and complex reflections in an MI-style smoking cessation conversation. However, some of the reflections are inappropriate, motivating the development of a classifier that can select relevant and good quality reflections. The classifier is trained and validated with data that is labelled by an expert MI counselor. We measure the difference in quality of reflections generated by GPT-2 and GPT-3, using both hand labels and the classifier. We show that GPT-3 is more likely to produce a good reflection, but that with the use of the classifier as a filter the GPT-2 model can be significantly improved and require less computational power. We also show that using a fine-tuned GPT-2 model significantly improves the reflection generation rate.

## 1.2 Focus and Goals

The objective of this work is to explore if it is possible to replicate specific therapist behaviour and skills used during a counseling session with a patient in an automated way. Though many validated therapeutic styles target behaviour change, this work focuses on replicating the skill of reflective listening, a major component of Motivational Interview-style counseling, in the context of smoking cessation. With this focus in mind, our goal is to determine if it is possible to use modern language models to generate an acceptable MI reflection that follows a patient's response to a prompting question in an MI conversation.

## 1.3 Contributions

To achieve the goal stated above, this work makes the following contributions:

- Demonstrates a way to use GPT-2 [12] and GPT-3 [9] based language models and their few-shot learning properties to generate acceptable MI reflections in a low-data setting.
- Demonstrates a method to detect and filter low-quality generated reflections using a fine-tuned BERT classifier with 81% accuracy, 0.90 sensitivity, and 0.71 specificity.
- Builds a system combining a GPT-2-based reflection generator and the reflection filter to build a system that produces acceptable reflections 73% of the time.
- Demonstrate that with enough training data in a specific domain, a fine-tuned GPT-2 based reflection generator can produce acceptable reflections 80% of the time in the same domain.
- Demonstrates how a reflection generation system can be deployed using off-the-shelf cloud infrastructure through AWS.

## 1.4 Organization

The remainder of this thesis is organized as follows: Chapter 2 reviews relevant background on the Motivational Interviewing approach, the language models used to generate and classify reflections, and related work on automated MI conversations. Chapter 3 describes the dataset that is used to condition/train the generator and classifier. Chapter 4 describes the reflection generation methodology, and Chapter 5 describes the training and performance of the classifier. Chapter 6 presents the performance of the GPT-2 model and the GPT-3 model on reflection generation using few-shot learning, the GPT-2 model together with the classifier as a filter, and the GPT-2 model

fine-tuned with MI conversation data. This chapter also presents a qualitative analysis of some of the generated outputs for each model. Chapter 7 demonstrates how the reflection generation system was deployed to be used with downstream MI-based web applications. Chapter 8 discusses the limitations of the presented work, concludes, and gives an overview of future directions for this project.

## Chapter 2 Background and Related Work

The first Section of this chapter provides relevant background on the Motivational Interviewing (MI) therapy approach. The second Section is a survey on the NLP landscape relevant to this work. The final Section reviews relevant prior work on the automation of MI.

### 2.1 Reflections and MI

MI [6] is a widely-used counseling approach to help people move towards positive behavior change. It has been used to target many health behaviours and has been shown to be a successful therapeutic tool for many types of behaviour change [28], including smoking [15]. A key goal of the approach is to move a person away from their ambivalence about changing a behavior, a state which leaves them unmotivated to change. MI counselors use a structured conversation that encourages a patient to contemplate the roots of the behaviour and guides them to overcome this ambivalence.

A key skill used by MI counselors is called reflective listening [6,13,16]. This includes careful listening to a patient and responding with words that both reflect what is said, and guides the patient towards continued exploration of their thoughts and feelings about change. These responses, called reflections, can be simple or complex. A simple reflection repeats or rephrases the patient's words to convey understanding. A complex reflection makes a relevant and helpful inference based either on previous statements by the patient or on the meaning within a patient's words. A good reflection invites the patient to continue their own internal exploration about the behaviour. The thesis of MI is that doing so will help the patient come to their own conclusion to make the change in behaviour. Table 2.1 shows three examples of simple and complex reflections that might be given in response to a patient statement.

<b>Patient Statement</b>	<b>Simple Reflection</b>	<b>Complex Reflection</b>
I don't think I have a problem with smoking, I don't smoke that much, and I can control my urges when I want.	You don't believe smoking is a problem for you.	You don't believe there is a problem with your smoking habits, and you don't see these sessions are helpful.

I really want to quit, but I get headaches if I don't have a few each day.	You get headaches if you don't smoke.	You're frustrated that you're not able to stop.
I can't just suddenly stop smoking; it would make my friends feel awkward when we hang out and they all smoke.	You're worried about ruining the mood when you hang out with friends.	It sounds like you're afraid that you won't be able to socialize with your friends without smoking together.

**Table 2.1: Examples of Simple and Complex Reflections**

An MI counselor requires significant dedicated training in MI which would typically be in addition to training in a social science or medical discipline. Counselors are expensive, requiring a full-time salary but can help only a limited number of people [29]. If an automated and effective counselor could be created, it would allow far greater scale of access to help at much lower cost. The automatic generation of good quality reflections is a first step towards this goal.

## 2.2 NLP and Language Models

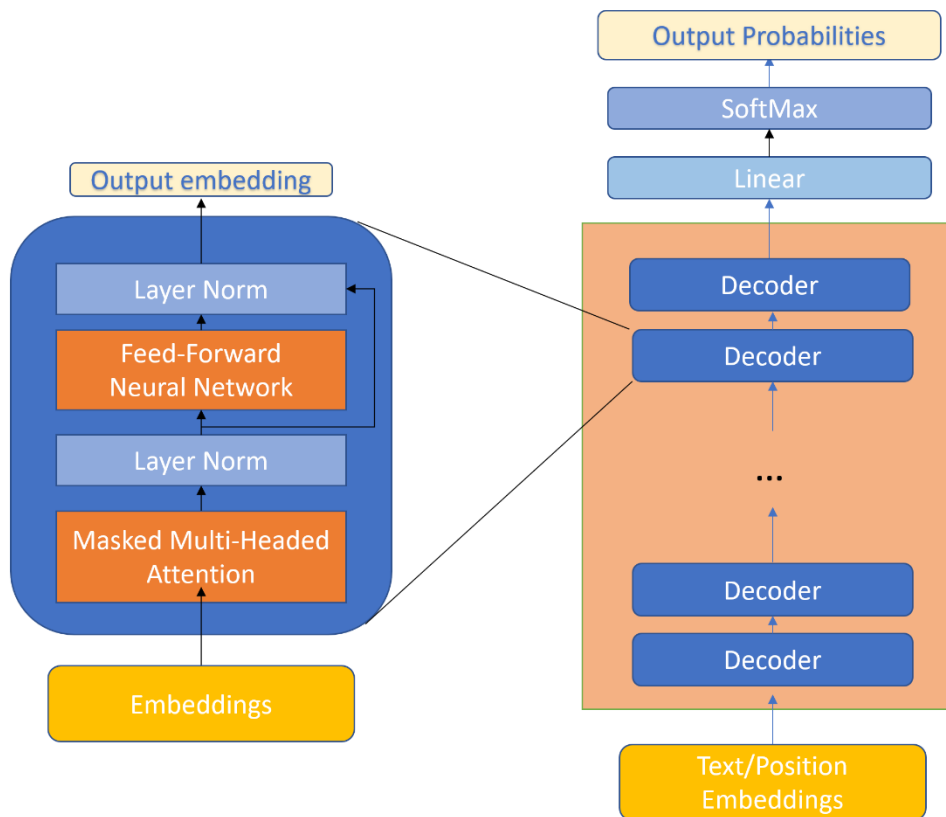
The field of Natural Language Processing (NLP) concerns the research and application of computational techniques to represent and manipulate natural language text and speech [30]. Recently, significant emphasis has been placed on using neural networks to build NLP systems for language representation [31,32], classification [33,34,35,36], and dialogue generation [10,37,38,39,40]. Of these, language models based on the Transformer architecture [41] have had the most success in generating human-like speech [37,42,43] and in text classification [44].

### 2.2.1 Generative Models

This work focuses on a specific type of language model known as an *autoregressive language model* (LM), which generate text by taking a sequence of tokens (that represent words) as an input and producing subsequent tokens/words [9,12]. The training objective of an autoregressive LM optimizes the likelihood of the next token given a series of input tokens. Autoregressive LMs also allow fine-tuning of the LM itself on downstream tasks, a feature essential in domains that have sparse training data, such as the conversation therapy arena in this work.

Fine-tuning for autoregressive LMs is a form of transfer-learning that allows LMs that have been pre-trained on one objective to be further trained on a separate objective to improve performance in a specific domain. In this work, finetuning is done on autoregressive LMs to improve their performance on MI-style dialogue containing a prompt, response, and reflections by further training the models with example dialogue.

One family of autoregressive languages models that have been very successful in various language modelling tasks is OpenAI’s GPT-2 (Generative Pre-trained Transformer 2) [12], and its more recent larger variant, GPT-3 [9]. GPT-2 employs the multi-headed self-attention decoder Transformer block [45,46], which is a slightly modified version of the decoder block described by Vaswani et al [41]. A key feature of self-attention is the efficient encoding of long-range



**Figure 2.1: GPT-2 Model Architecture**

dependencies in text, compared to earlier text-generation models based on recurrent neural networks [35,36]. The attention layers do this because they have visibility over a large context window of the input. Figure 2.1 illustrates the Decoder layer and how a stack of decoders are used to form GPT-2, adapted from [46,45,12].

The GPT-2 model [12] has 1.5 billion parameters and was pre-trained using the WebText [47] language corpus internal to OpenAI, comprised of 40GB of internet text. The dataset was curated with an emphasis on document quality, only including websites that are outbound links from Reddit with significant user engagement. It comprises text from over eight million documents and does not include text from websites that are present in other popular textual datasets such as Wikipedia.

The GPT-3 model [9] has 175 billion parameters and is trained on 499GB of text data from multiple datasets: Common Crawl from 2016-2019 [48], WebText, Books1 and Books2 (collection of books and movie script text data) [49], and English-language Wikipedia [50].

In this work, we make use of the pre-trained GPT-2 and GPT-3 models to generate reflections using *few-shot* learning [9]. In [9,10] the term ‘few-shot learning’ is defined differently than the original notion [51,38] of having few training examples that fine-tune the model parameters. Here, rather, the input to the model (as opposed to the training samples) are several complete examples of the text generation task itself. Each example consists of the *context* and desired *completion* (which is what the generated text should look like). After the examples, the model is given a final context, and from which it generates a completion. The presentation of the examples to the model input is referred to as ‘priming’ or ‘conditioning’ the model.

The examples that form the input to the GPT-2 and GPT-3 model are first converted from their text form to an embedding, using a tokenizer. Our work uses the tokenizers provided by the HuggingFace library [52]. Then the inputs are passed through a series of Decoder layers, as shown in Figure 2.1. The exact number of Decoder layers is determined by the size of the model used. The standard GPT-2 model has 12 Decoder blocks; the GPT-2 XL model used in this work has 48. Once the input has passed through the decoder blocks, it is passed through a linear and softmax layer, which produces a vector with the output probability for each token in the vocabulary. A decoding strategy is then used to select the next token from this distribution, the methods for which are described in Table 2.2. The token is then appended as part of the input and the entire process repeats until an ‘end\_of\_sentence’ token is reached, at which point the generation is complete and the tokens can be converted back into text form.

GPT-2 and GPT-3 are fixed-length models, meaning the amount of conditioning presented to these models is limited by the maximum size of the input available to the model. The maximum size of



the inputs to GPT-2 is 1024 tokens, and for GPT-3 it is 2048 tokens. In general, larger/more conditioning generates better completions [9]. Adjustments to the GPT-2 architecture have been proposed to address this limitation by turning it into a model with unbounded context by implementing a recurrence mechanism [53]. There are other models such as the Transformer-XL [27] and XLNet [54] that can handle much larger context windows, however these models require fine-tuning to perform well on domain-specific tasks.

Other factors that influence the quality of the generated text in any autoregressive language model is the choice of decoding strategy, which is the algorithm used to select the next word to be generated given the probabilities across the vocabulary [9]. There are significant differences in the quality of the text generated between different decoding strategies, which are described in Table 2.2 along with the parameters that affect the strategy. A greedy approach will select the most word with the highest probability, given all the previous words but typically does not produce a high probability final text, with multiple words. More complex search-based methods such as beamforming [55], are better, but require significantly more computation due to many more invocations of the model inference. It has also been shown that the volatility inherent in human dialogue is best mimicked with a sampling technique such as top-k, or top-p (nucleus) sampling [55]. GPT-2 has a choice between greedy decoding (with no parameters), search-based decoding with beam search (which uses the number of beams as the parameter), and sampling decoding methods (which uses a combination of top\_k and top\_p sampling). A repetition penalty [56] is also used to reduce the likelihood of a previously input or generated token to be repeated.

<b>Decoding Method</b>	<b>Description</b>	<b>Decoding Parameters</b>
Greedy	At each step, select the highest probability token to be part of the output. Repeat until stopping token reached.	None
Beam-Search	At each step, keep track of the num_beams highest probability sequences. Repeat until stopping token reached for each beam, then return the highest likelihood sequence.	num_beams: The number of sequences of tokens to track as candidate completion sequences.

Sampling	At each step, sample from either the top_k number of tokens or from the number of tokens corresponding to the top_p probability. Repeat until stopping token reached.	<p>Temperature: adjusts the differences in probabilities between the most likely and least likely tokens. Temperature of 0 forces the selection of the most likely word (greedy) while an infinite temperature will degenerate the distribution to the uniform distribution and make every token equal probability.</p> <p>top_k: the number of highest probability tokens to sample from.</p> <p>top_p: threshold for the sum of probabilities of the highest probability tokens to consider when sampling.</p>
----------	---	--

**Table 2.2: GPT-2 Decoding Strategies**

### 2.2.2 Text Sequence Classification Models

Also relevant to this work is the use of a second model to classify generated text to filter out low-quality generation. A previous system that took this approach is the DialoGPT system [40]. It uses a maximum mutual scoring function to rank generated text to filter bland and generic outputs. In Chapter 5 we describe a simpler approach using a fine-tuned text classifier as a secondary model to filter out low quality reflections.

Text sequence classifiers come in two flavours: rule-based and machine learning-based methods. Rule-based models make use of hand-crafted features that are then used in a classification model to classify text sequences. One popular method of deriving hand-crafted features is using the bag of words (BoW), combined with a simple classifier (Naïve Bayes, Support Vector Machines, hidden Markov Models) [44]. Although rule-based methods had early success with text sequence classification, they required extensive domain knowledge and feature engineering. The features were also static; thus, they could not be improved upon by using the growing number of text corpora that are available [44].

The machine learning-based methods address many of the weaknesses of rule-based methods, with the caveat that large text corpora training data are a required for good performance. These model types are used to derive continuous low-dimensional representation of features using vectors, commonly referred to as embeddings, that can be used in downstream tasks for text sequence classification. Early machine learning-based methods, such as latent semantic analysis (LSA) [31], suffered from low parameter sizes and small training corpora, thus they did not perform as well as rule-based methods [44]. More recent methods used significantly larger models and corpora with billions of words and perform significantly better than rule-based methods [44]. Embedding models such as Word2Vec [32] and ELMo [57] have been very successfully used for text-sequence classification tasks using Long-Short Term Memory (LSTM) models [34,33]. Although these models performed well generally, they needed to be fully trained from scratch for each task, reducing these models' utility in low-data domains.

A newer model architecture based on the transformer architecture [41] has been shown to be significantly more powerful for text sequence classification tasks than its predecessors. The Bidirectional Encoder Representation from Transformers (BERT) [11] model uses Masked Language Modeling (MLM) as a pretraining step to learn its representation of words. MLM works by masking a subset of the input, and then having the model predict those masked words. In addition, BERT combines this task with the task of predicting if a second sentence logically or naturally follows from the first. This combined training objective allows BERT to encode relationships between tokens and their positions in a text sequence and to learn entailment [44] relationships from text sequences.

Once the pre-training step is complete, the BERT model can be augmented by appending various neural network layers to its outputs. The augmented model can then be further fine-tuned on a small dataset for specific natural language understanding (NLU) tasks, including classifying sequences of texts based on specific characteristics. For example, BERT can be fine-tuned to detect if question-answer pairs make sense. It can also do general textual entailment between two sequences of texts [44]. In Chapter 5 we use a fine-tuned BERT model to classify reflections, as a filter on the generated reflections.

## 2.3 Related Work on Chatbots for MI

There has been a small number of previous chatbots that used Motivational Interviewing approaches for behaviour change. That prior work was in several domains, including stress management [21], sexual health education [39], and smoking cessation [17]. There are also two other studies of the use of MI chatbots, that are clinically-oriented and so do not provide details on the design and engineering of the system itself [23,58].

Park et al. designed a qualitative case study for a chatbot that conducts a brief MI session with university students [21]. The researchers constructed pre-written responses in four MI categories of counselor statements: giving information, asking questions, providing reflections, and MI-adherent statements. The responses were designed to be generic enough that they could be used in many conversational contexts. The chatbot responded to statements by selecting from the set of responses based on specific keywords present in the statements. The researchers found that the students responded well to the chatbot's open-ended questions but did not relate strongly to the reflective and affirming statements, since these statements did not directly address what the students had said. The participants felt the chatbot could be improved if it were able to give more contextual replies and informational support. The present work addresses this issue directly by generating a unique reflection from the preceding question and response.

A second prior effort where MI chatbots have had some success is providing sexual health information through Facebook [39]. This system is designed to address the 'giving information' component of the MI conversation, specifically providing information about HIV/AIDS. It uses a question-answer corpus to train a response classifier to rank potential responses to a user's questions and presents the most likely response while accounting for repetitiveness. Here the responses are also pre-written, but because there is a larger set to select from, it more likely to be responsive. The context of providing information is also more straightforward than that of making reflections, the subject of the present work.

A more complete automated MI chatbot targeting a specific behaviour was created by Almusharraf et al. [17,59]. The researchers designed a fully automatic MI-style chatbot capable of conducting a conversation relating to an individual's reasons for smoking. Their system followed a special form of MI known as the *running head start* technique [6]. The chatbot identified a person's reasons for and against smoking by asking open-ended questions and classifying their responses

into one of twenty-one categories. The system was evaluated with 121 participants that produced over 6568 responses to the chatbot. Though a significant portion of participants (35%) found their interactions with the chatbot helpful in getting them to think about their smoking, there were some weaknesses also identified. The conversation was specific to the running head start method and does not generalize beyond that; it also cannot respond to participant statements outside of the scope given by the smoking reasons. In addition, the structure of the responses was often felt by the participants to be repetitive, as they were also scripted and had a repetitive structure.

In the most relevant prior work to the present one, Shen et al [26] explored how generative models can be used to create reflections that could be used in actual MI conversations. Their work focuses on trying to reproduce reflections from pre-existing clinical conversations. The goal was to see how closely a GPT-2 based language model can reproduce the actual reflections given by the therapists and counselors from those conversation sessions. The authors demonstrate that it is possible to produce compelling free-form reflections in a therapeutic context using a transformer-based generative language model. They showed that it performs better than standard seq2seq-type models. Their approach is to first domain-adapt a GPT-2 based model on transcripts of videos from Alexander Street Press and then to fine-tune the model using an MI counseling dataset [60]. The authors then evaluate the GPT-2 model's ability to generate reflections given a subset of dialogue history. They also explored how adding more context, by adding reflections from similar conversations as part of the input, changes the quality of the generated reflections. The authors then evaluated the reflections produced by GPT-2 on how similar they were to the actual reflections from the context conversations, as well as manually labeling the generated reflections on the quality of the grammar, reflection-likeness, and relevance. They found that the GPT-2 based models perform better than the baseline seq2seq models on the generation task. Adding similar reflections to the context slightly improves the similarity of the generated reflections to the ground truth, though not in all cases. Their qualitative analysis also show that the generated reflections were relevant, high quality (in terms of grammar and structure) and had high reflection-likeness.

All four of these studies cited a need for the chatbot to provide more free-form, contextual responses to the user's statements or queries, described below in Chapter 6.

## Chapter 3

# Dataset and Therapeutic Context

This chapter describes the data set used in this work. The first section gives an overview of the type of data required to achieve the goal of this work. The next section describes the source dataset. The third section details how this dataset was enhanced to provide training data for few-shot reflection generation and reflection classification.

### 3.1 Overview of Required Data

The goal of this work is to automatically generate a reflection, in the context of behaviour change therapy for smoking cessation. The input to this process is an open-ended question/prompt, together with the response from a patient. The output is a reflection as described in Section 2.1. We have also noted the need for a classifier that can detect good quality reflections, ultimately for use in filtering the generations, but also useful for measuring the quality of the unfiltered generation process.

For both tasks, a dataset is required that contains prompts, responses, and reflections. Ideally these would come from real counselors and their patients engaged in MI counseling sessions, but such counseling transcripts are protected by medical privacy laws and regulations. Fortunately, we have access to data from our previous study, in which we prototyped a different MI-oriented chatbot [59,17]. It provides us with 204 conversations to mine for appropriate prompt-response pairs. The next Section describes this dataset and the original experiment from which the data was collected in more detail.

### 3.2 Source Dataset

We collected transcripts from the deployment of a previous MI-oriented chatbot that explored an individual's reasons for smoking [59,17]. The transcripts contain open-ended questions from the chatbot asking for the participant's reasons for and against smoking. The bot then classified each reason into one of 21 categories, and then selected a reflection based on that classification from a human-authored set. Subsequently, the chatbot asks the participant to recall situations where the reason was in play – for example, 'recall a time when you were stressed.' It asked the participants

to describe a time that this reason caused them to smoke. In addition, they were asked to describe a time that they did not smoke with that reason in play. Finally, the participant was asked to say what enabled them to resist smoking in the case that they smoked, compared to the time that they did not smoke. This last question was a key part of the conversation, as the intention was to evoke contemplation within the participant, on what gave them the ability to resist the addiction [59,17].

From each transcript in this dataset created by Almusharraf et al. [59,17], we extracted pairs of utterances. The first utterance was the question asked by the chatbot itself, with the second utterance being the response from the participant. We then manually reviewed this collection of utterance pairs in two steps. The first step was to discard any utterance pairs where the first utterance was not an open-ended question (where the possible answers were limited to specific responses – for example ‘yes’ or ‘no’). Secondly, we reviewed the remaining utterance pairs based on whether the participant responses could (and should) be reflected upon, discarding any utterance pairs that do not warrant a reflection.

The resulting dataset consists of 1665 utterance pairs where the first utterance is the prompting question, the second being the participant response. However, this dataset does not have the (required) reflections that are needed for few-shot generation (as described in Chapter 2 and 4, which requires a completion) and classifier training (described in Chapter 5). The prior study did not have the capability of generating a free-form reflection. In the next Section we describe how this dataset was modified and enhanced for both the few-shot reflection generation task and for the classifier training task.

### 3.3 Enhancing Dataset for Few-Shot Reflection Generation & Classification

The dataset described above contains utterance pairs of prompts and responses. To create high quality conditioning examples that can be used in the few-shot reflection generation task, we first manually constructed a set of high-quality reflections for each prompt-response pair in the dataset, to form the requisite completions.

The creation of good-quality reflections requires expertise in the field of motivational interviewing, which we gained in two ways: first, the authors undertook a training course in MI, and also read and reviewed a fundamental text in the field [6]. We then created 20 reflections,

manually, to go along with 20 of the prompt-responses. Using the few-shot priming approach we generated 369 reflections (based on 123 different prompt-response pairs, reflected three times each) using the few-shot approach described in Chapter 4.

Each of these 369 reflections were then labelled by an MI expert practitioner at the Centre for Addiction and Mental Health in Toronto. The MI expert also provided adjustments to poor quality generated reflections to improve the dataset.

The reflections were given a binary label with ‘0’ meaning that the reflection is not appropriate to the prompt and response, and ‘1’ meaning that the reflection is consistent with MI. The reflections and the corresponding prompts and responses that received a label of 1 were exclusively used as the set from which examples were drawn for the few-shot reflection generation task. This dataset will be referred to as the priming set.

Using the priming set, we were able to generate and label many more prompt-response-reflection triplets that were of good quality over multiple experiments. Due to the generative nature of the task and the fact that the overall dataset was collected over many reflection generation experiments with varying decoding parameter choices (described in Chapter 4), we were able to produce multiple distinct reflections for each prompt-response pair. These reflections were then labelled by non-expert (student) labellers and left us with a total of 3301 prompt-response-reflection triplets, 39% of which were positive and 61% were negative. This set of prompt-response-reflection-label collection was then used as the training data for the reflection classification task, described in Chapter 5.



## Chapter 4

# Few-Shot Reflection Generation Methodology

Modern transformer architectures are pre-trained on large corpora of text, so they are already capable of generating grammatically correct text given a seed text. The seed text can be either an example text showing how to form a completion, or a written description of what the completion should be [9,11,42,10]. In this Chapter, we describe how transformer models can be used to generate text that is MI consistent, in a domain with low data availability. We employ two approaches: few-shot learning (which does not require model training) and fine-tuning (which does require model training).

As described in Chapter 2, a reflection is a response to a prior open-ended question (the prompt) and the response from a patient. The reflection generation task is to produce a reflection given the prompt and response. We describe the reflection generation process by first describing the few-shot learning methodology using GPT-2 and GPT-3 (introduced in Chapter 2) in the first Section. The first subsection explores how the number of conditioning examples affects the quality of the generated reflections while the second subsection describes how different decoding parameters affects the quality of generated outputs, and which set of parameters were used during generation for the remainder of this work. The second Section describes the fine-tuning steps used with GPT-2 to attempt to improve its native reflection generation rate.

### 4.1 Few-Shot Learning With GPT-2 and GPT-3

We first employ the few-shot learning method (described in Section 2.2.1) using the GPT-2 [12] and GPT-3 [9] models for the reflection generation task. The GPT-2 pretrained models are easily accessible from the Huggingface library [52]. We were able to access the GPT-3 pretrained models from OpenAI due to a generous program from OpenAI that enabled several projects at the University of Toronto to access and run the large models, but within a fixed budget. As such, we decided to explore the various parameters and conditions needed to create good reflections mostly with the GPT-2 model given the constrained access to GPT-3.

The context of the input is the prompt and the response whereas the desired completion is the reflection. The few-shot learning method [9] requires several examples of the context and

completion to be prepended to the specific input context for which a completion is desired. Since the GPT-2 and GPT-3 models have finite context windows, the maximum number of example context-completions is limited. For GPT-2 the approximate example limit is eight, and for GPT-3 is it roughly 12. These numbers are determined by dividing the context window size by the average size of context-completions in the dataset. However, the exact number of examples may vary because the size of the examples themselves vary. So, in the case the input token was greater than the context size limit (which did happen occasionally) the input was truncated from the start until the input was equal to the maximum number of tokens.

The prepended examples are drawn from the priming set described in Section 3.2. According to Brown et al. [9], the examples and the final context must be constructed in a specific way, with delineators between the elements of the context and completion. We chose to prepend the prompt with the identifier ‘Prompt: ’, and on the next line we prepend the response with ‘Response: ’, followed by another line where the reflection is prepended with ‘Reflection: ’. To distinguish between each example, they are separated with a blank line. The final prompt has one line for the prompt, and a second line for the response. The response line is appended with a newline character. In the case of GPT-2, the entire input is then passed through the pre-trained GPT-2 tokenizer from the HuggingFace library, based on byte-level byte-pair-encoding [52]. The generator then follows the examples to produce a completion, which is usually (but not always) of the form ‘Reflection: {some text}’. If the generator fails to produce output in this form, we reject the output and record it as a blank string. Figure 4.1 presents an example of the full few-shot generation input with three example context-completion priming pairs and a final context without the completion.

```

Prompt: Please describe a time where you were worried about the smell of
cigarettes and you didn't smoke
Response: When I was going to my doctor. I refrained from smoking on the way so
she wouldn't smell it and be reminded to lecture me about smoking.
Reflection: You are afraid of being reminded by your doctor to talk about
smoking.

Prompt: Think back to the time when you were able to prevent yourself from
smoking. What made it different from when you did smoke?
Response: Because of the effect it was having on my daughter
Reflection: You're worried how smoking affects your daughter.

Prompt: Please describe a time where you had a compelling feeling to smoke but
didn't end up smoking
Response: I was compelled to have a cigarette before an interview but refrained

```

**Reflection:** You felt like you could do without cigarettes.

**Prompt:** Please describe a time where you smoked because you needed to feel good

**Response:** When I am feeling stressed out at work.

**Figure 4.1: Example Few-Shot Input Sequence**

### 4.1.1 Choosing the Number of Primers

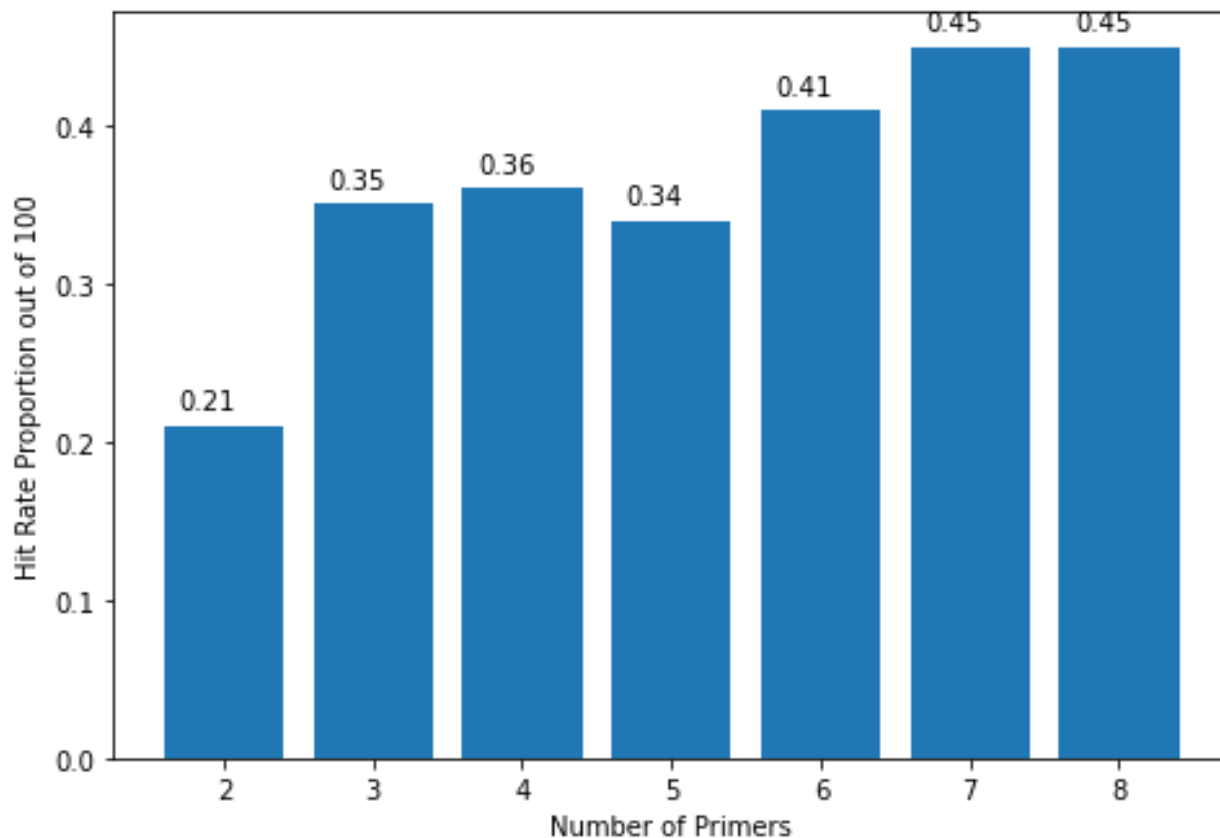
A key parameter of the few-shot generation method is the number of context-completion examples to use to condition/prime the models. To determine this number, the following experiment was performed: For 100 randomly selected prompt-response pairs, reflections were separately generated using seven different numbers of primers, from 2 to 8. The model used was GPT2-XL from the HuggingFace model library [52]. The primers were resampled from the primer set for each generated completion. Top-k sampling with temperature was chosen for the decoding strategy of this experiment, with a temperature of 0.4, top-k of 100, and top-p of 1.0. The high value of top-k and top-p of 1.0 ensures that a sufficient number of high probability words are considered for the completions, while a temperature of 0.4 ensures some creativity in the generated responses without letting the text become too random [55]. Other combinations of decoding values are explored in the next Section.

In some cases, the input size was larger than the context window of GPT2-XL, which is 1024 tokens. To be able to use these larger inputs to produce reflections, the input was truncated during tokenization, removing the input text from the beginning until the input size was reduced to 1024 tokens. In the case that the input was smaller than the context window size, the end of the input was padded with ‘end-of-sentence’ tokens.

Every reflection generated by this process was hand-labeled according to the binary labeling criteria described in Section 3.3. We will refer to the fraction of reflections that were acceptable as the hit rate.

Figure 4.2 is a bar chart of the hit rate plotted against the number of priming examples (which range from 2 to 8.). These results suggest that, as the number of primers is increased, the likelihood that GPT2-XL can produce acceptable reflections reaches a plateau with seven and eight primers.

We will use eight primers for the remainder of the experiments with GPT-2 in this work. We can observe that using less than three priming examples significantly reduces the likelihood of producing an acceptable reflection.

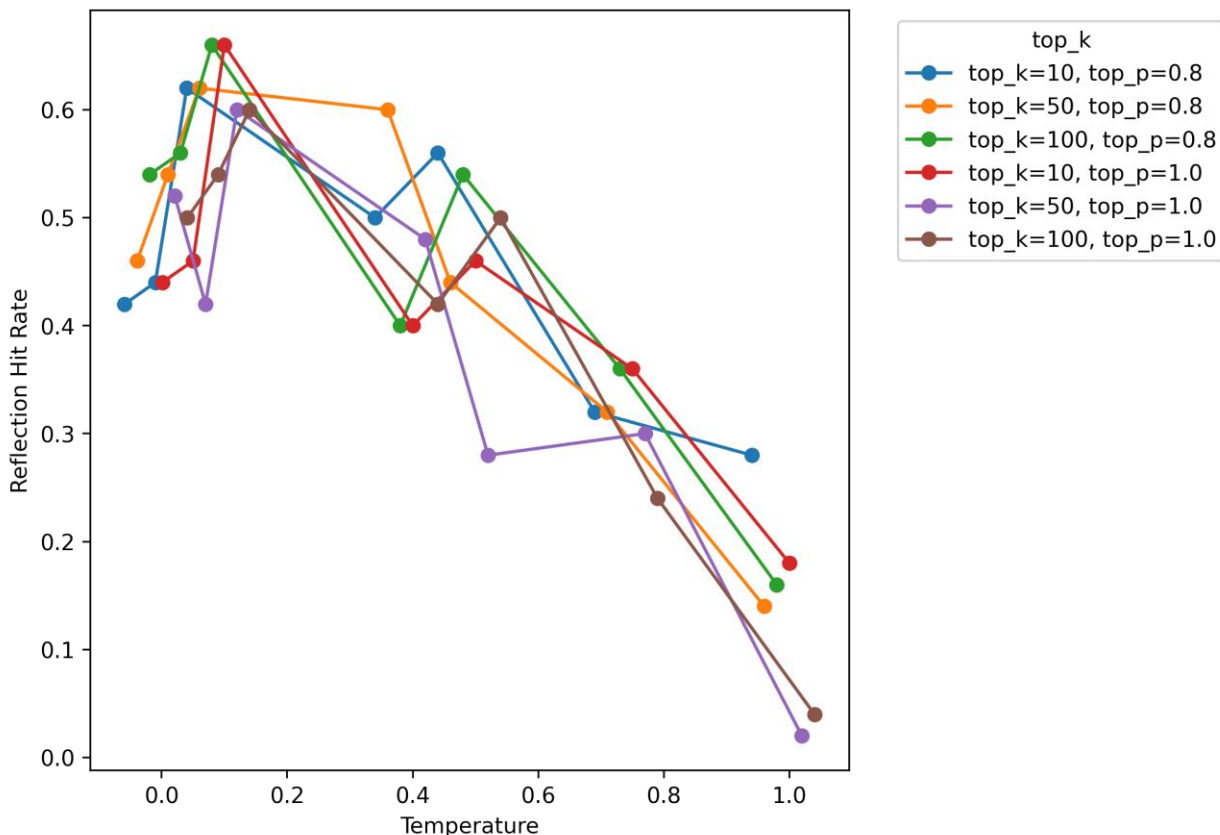


**Figure 4.2: Reflection Generation Hit Rate for Different Number of Primers for GPT-2**

#### 4.1.2 Exploring Decoding Parameters for Generation

A second factor that has significant effect on the quality of the generated output from GPT-2 is the decoding strategy, as described in Section 2.2.1. We focus on finding good decoding parameters, determined by finding the hit rate on a test sample, for sampling-based decoding as they have been shown to be more effective than search or greedy strategies for dialogue generation [55]. We explored different combination of values for temperature, top\_k, and top\_p. To select the best decoding parameters for these experiments and datasets, we first produced 50 reflections using each combination of parameters, and then manually labeled the generated reflections to determine the hit-rate. The results are presented in Figure 4.3, which gives several curves for different combinations of parameters, with temperature on the x-axis. The hit rate is somewhat noisy as a

function of temperature (because there are only 50 examples per point), but the trend is quite clear: a low temperature achieves the highest hit rate. For the remainder of this work, we chose the following decoder parameter values: temperature = 0.1, top\_k = 100, and top\_p = 0.8, which had the highest hit rate of 0.66.



**Figure 4.3: Reflection Hit Rate at Different Decoding Parameter Combinations for GPT-2**

GPT-3 has many more options for decoding parameters (than GPT-2) that impact the quality of the generated output. The only GPT-3 parameters/options that were the same as the decoding parameters of GPT-2 are temperature, top\_p, and max\_tokens. The other decoding parameters relevant to generating a single completion were kept at their default values (as recommended by OpenAI) which are: presence\_penalty = 0, frequency\_penalty = 0, best\_of = 1, logit\_bias = null. A full description of these parameters and others can be found in the OpenAI API documentation page [61]. When modifying temperature and top\_p, the OpenAI API documentation suggests only modifying either one or the other but not both when trying to find good values for decoding parameters for GPT-3, so we chose to vary the temperature but keep top\_p at its default 1.0 value.

Ten reflections were then generated with eight primers for temperature values of 0.1, 0.4, 0.5, 0.75, and 1.0. The reflections were then manually reviewed for coherence and repetition. We observed that the temperature value of 0.4 provided the least repetitive and most coherent reflections and so this value was used to generate reflections for the remainder of the experiments when using GPT-3.

## 4.2 Fine-Tuning GPT-2 For Reflection Generation

Fine-tuning of large language models has the potential to improve their performance on specific tasks. In this Section, we describe the method used to fine-tuning a pre-trained GPT2-XL model from the HuggingFace Library [52].

A sample of 1700 prompts, responses, and reflections with label ‘1’ from the dataset described in Section 3.3 was used as the training data to fine-tune GPT-2. To prepare the data for training, each example prompt, response, and reflection were concatenated together into a single string in much the same way that priming examples are constructed in Section 4.1. An example of a datapoint is provided below in Figure 4.4. The same GPT-2 pre-trained tokenizer described in Section 4.1 is also used to tokenize the examples in this training set.

<p><b>Prompt:</b> Please describe a time where you were worried about the smell of cigarettes and you didn't smoke</p> <p><b>Response:</b> If I went to a formal meeting</p> <p><b>Reflection:</b> Your anxiety over whether someone will notice how badly your clothes smell made you decide against smoking</p>
---

**Figure 4.4: Example GPT-2 Fine-Tuning Datapoint**

The fine-tuning objective of GPT-2 is the same as the pre-training step, which is to model the standard language model task [12,62]. For our example, the objective is to model the conditional probability of a token from the token vocabulary to be the next token, given a sequence of input tokens as the input. The Transformers library from Huggingface [52] provides a Trainer class that abstracts this training objective, and is used for our fine-tuning task.

Model training was done for three epochs, with a batch size of one and two gradient accumulation steps. The learning rate was set to  $3 \times 10^{-6}$ , using the version of the Adam optimizer [63] implemented by DeepSpeed [64]. DeepSpeed helps us train the GPT-2 XL model by allowing us to use mixed precision to train the model while staying within the constraints of the VRAM available in the system. FP16 was enabled within DeepSpeed, with the default ranges for mixed

precision parameters, which lets DeepSpeed automatically determine where and by how much to reduce precision based on the constraints of the hardware.

The fine-tuning was done on Amazon SageMaker [65] in an 'ml.p3.2xlarge' machine instance which contains hardware optimized for lightweight model training jobs running on Amazon Linux 2. The training code was written in Python 3.6 and hosted on a Jupyter Notebook [66] instance. SageMaker's built-in version of the Transformers 4.6 and HuggingFace Library [52] was used, alongside PyTorch version 1.7 and DeepSpeed [64] version 0.4.4.

Chapter 6 reports on the performance of this fine-tuned GPT2-XL model at generating reflections and compares it to the non-fine-tuned versions.

## Chapter 5

# Automatically Classifying Reflections

As discussed in Chapter 4, current transformer-based models are not able to generate acceptable reflections all the time, and so there is a need to identify when an unacceptable reflection has been produced. This can be done by training a classifier that takes the prompt, response, and generated reflection as input and produces an acceptable/unacceptable label as output. Such a classifier can also be used as a tool to rapidly label many samples that have the prompt, response, reflection format. This chapter describes one way to build this classifier, which we call a Reflection Quality Classifier (RQC), that can distinguish a high-quality reflection from a poor one.

The first Section introduces the inputs and outputs of the RQC as well as the base model we will use, followed by the second Section which discusses the RQC training process. Finally, the third Section discusses the performance of the RQC as a labeling tool.

### 5.1 Reflection Quality Classifier (RQC) Description

The RQC is a classifier which will take two strings as inputs: a context followed by a completion and produce a binary label that represents whether the completion is an acceptable reflection to the context. The output is 0 if the reflection is not acceptable and 1 if it is acceptable. This binary labeling scheme was chosen to reduce the noise with amateur labelers. More granular labeling schemes requires supervision from an expert MI practitioner to produce precise labels, and we had limited access to expert resources for labeling. The inputs to the RQC are built by concatenating a prompt, a newline character, and a response as the context and using the corresponding reflection as the completion.

The RQC is trained by finetuning a pretrained uncased BERT Base model from the HuggingFace library [52]. BERT is used as the underlying model since it has been shown to be successful at text sequence classification tasks when finetuned with a small dataset. The input text is tokenized using the corresponding pretrained BertTokenizer from HuggingFace tokenizers [52], based on the WordPiece tokenization algorithm [67].



## 5.2 Training the RQC

The model was fine-tuned in two stages: the first step used the Microsoft Research Paraphrase Corpus (MSRP) [68], which captures a text *entailment* [69] relationship between a pair of texts. Each pair has a binary label which indicates whether the second text is a paraphrase of the first, or not. This is similar to the relationship between the MI prompt/response and reflection for *simple* reflections. Using the MSRP dataset increased the total amount of data that the model can be trained on, without increasing the burden of collecting and labelling the difficult-to-collect MI-style conversations.

The MSRP dataset comes pre-split into a 4077 training samples and 1726 test samples. We further subdivided the training samples into 80% training and a 20% validation set.

The model was further fine-tuned with the dataset described in Section 3.3, containing prompts, responses, reflections, and a binary label for the acceptability of the reflection. Since the dataset is unbalanced, a sample of 1075 positive and 1075 negative examples were randomly selected for the training task, giving a total of 2150 samples. We then shuffle and split this dataset into 80% training, 10% validation, and 10% test sets.

For both fine-tuning steps training was done using the `tf.Keras` functionality within the Tensorflow library [70]. An initial learning rate of  $3 \times 10^{-5}$  was used over 20 epochs, with early stopping enabled if the validation accuracy does not change over 3 epochs. The Adam optimizer was used due to its ability to adapt the learning rate over the training period, and it has also been shown to work well with large parameter models [63]. The loss function was the sparse categorical cross entropy loss which is recommended by the Tensorflow library when there are two label classes and they are provided as integers, and not one-hot encoded [70].

Table 5.1 reports the accuracy, sensitivity, and specificity of the model at each stage of the fine-tuning process, across the 220-sample test set described in Section 3.2. Each row of the table gives the accuracy impact with different levels of finetuning: first without any finetuning (the row labeled ‘Bert-base No Finetuning’) and when the two different datasets are used to fine-tune BERT separately (‘BERT-base MSRP Fine-tuned’ and ‘BERT-base Reflection Fine-tuned’), and when they are used together (‘BERT-base MSRP+Reflection Fine-tuned’).

Table 5.1 shows that the base pure BERT MODEL without finetuning is no better than an uninformative model (at 50% accuracy), but that both training datasets improve this when used to train the model separately and do even better when used together. This approach shows that even in the absence of a large dataset in a target domain, we can still achieve improved classification performance by drawing on datasets designed for similar tasks in different target domains.

Model	Accuracy on Test Set	Sensitivity	Specificity
BERT-base No Finetuning	50%	0.07	0.95
BERT-base MSRP Fine-tuned	55%	0.13	0.97
BERT-base Reflection Fine-tuned	73%	0.82	0.59
BERT-base MSRP+Reflection Fine-tuned	81%	0.90	0.71

**Table 5.1: RQC Model Accuracy, Sensitivity and Specificity Pre- and Post- Fine-Tuning**

Since one role of the RQC is to filter generative outputs, it is important to know how likely it is that the RQC will produce a false positive classification, and thus fail in its filtering function. For the 220-sample test set used above, Table 5.2 gives the count of true positives, false positives, true negatives, and false negatives for the final BERT-base MSRP+Reflection fine-tuned model. From these we can calculate the specificity, or true negative rate, which is 71% for the final model, meaning that the chance that an unacceptable reflection will be classified as acceptable is 29%. The false negatives are less important in the filtering application as the cost of a false negative is the time and effort required to generate another reflection. However, we must be careful to select a model for the RQC that also has a good sensitivity, as practically speaking, we need an RQC filter that eventually lets a positive class through. Table 5.1 shows that the sensitivity improves with more of the fine-tuning steps included, while specificity decreases. The best balance of specificity and sensitivity is achieved with BERT-base MSRP+Reflection Fine-tuned, and this model is used as the RQC in the following chapters.

When used as a reflection dataset labeling tool, the RQC’s performance in labeling both the positive and negative classes are important. For this case the accuracy and F1 scores will give us an idea of the RQC’s performance as a labeling aid. The overall accuracy of the RQC is 81%, and the F1 score is 83%.

Class Labels	Predicted Positive	Predicted Negative
Actual Positive	103 (True Positive, TP)	12 (False Negative, FN)
Actual Negative	30 (False Positive, FP)	75 (True Negative, TN)

**Table 5.2: RQC BERT-base MSRP+Reflection Fine-tuned Result Counts**

### 5.3 Automatic Reflection Labeling with the RQC

A secondary use of the RQC is as an automatic reflection labeling aid to help parse and label large collections of MI transcripts to identify utterances with reflections. This has application as both an aid for new MI practitioners evaluating MI conversations, or to help build large, labeled corpora of MI utterances with labeled reflections.

We evaluate the performance of the RQC as a labeling aid by using it to label 300 reflections produced natively by GPT-2 and GPT-3 models (these are all the reflections generated by the models for the testing done in the next chapter). We then compare the RQC labels to the hand-labeling, using the Cohen-Kappa Interrater correlation [71]. It measures inter-rater reliability between labelers who are labeling qualitative items. It considers the likelihood that two labelers agreed on the label of a sample randomly, which makes this correlation values more conservative than simple percentage agreement between the two labelers. McHugh [72] reports that a Cohen-Kappa score of 0 implies no agreement between the two labelers, 0.1-0.20 implies slight agreement, 0.21-0.40 implies fair agreement, 0.41-0.60 as moderate agreement, 0.61-0.80 as substantial agreement, and 0.81-1.0 as almost perfect agreement. The agreement between the hand-labeling and the RQC labeling is shown in Table 5.3 with both the Cohen-Kappa correlation as well as the percentage of matching labels.

Cohen-Kappa Correlation	Percentage Agreement (%)
0.33	75

**Table 5.3: Inter-rater Correlation Between Hand-Labeling and RQC**

The RQC agrees with the human hand-labeling 75% of the time, with a Cohen-Kappa score of 0.33. The Cohen-Kappa score indicates ‘fair agreement’ between the hand labeling and the RQC in this dataset.

## Chapter 6

# Generating Reflections

This chapter describes the different methods used to generate reflections and reports on the performance of each method. The first Section begins by describing the models and the parameters used for each model during generation. The second Section describes the quantitative performance of the models which first reports the performance of generating reflections using GPT-2, with and without using the RQC as a filter. Then, the performance of generating reflections with GPT-3 is described, using both eight and 12 primers. We then report how a GPT-2 model, fine-tuned using the method described in Section 4.2, performs against the previous models. In the third section we provide a qualitative discussion of a sample of acceptable and unacceptable generated reflections from each model.

### 6.1 Generation Set-up

We first present the conditions and parameters used for the models during the few-shot generation process: First, the output sequence was limited to a maximum output of 200 tokens for both the GPT-2 and GPT-3 based models, which causes generation to stop when this limit is reached (generation also stops when an end-of-sentence token is generated). This limitation on the output length was placed to prevent a model from generating extremely long sequences of text, which does occur occasionally. We know that a very long output would almost always have little value as a reflection, as in MI, it is good practice to let a patient speak more than the therapist [6]. Since we know that typical participant statements are roughly 14 words and the longest response recorded was 73 words, an acceptable reflection should be in this range, which corresponds roughly to 200 tokens.

The primers used in the few-shot generation were randomly selected from the priming set described in Section 3.2, and are separated from the test set used to measure performance. The generation is launched by appending the primers with the input prompt and response, for which a completion was generated.

When generating reflections with the GPT-2-XL model, eight primers were used (as described in Section 4.1.1) when generating using few-shot priming. The decoding parameters were set to the best-performing combination that was explored in Section 4.1.2, which are as follows: temperature = 0.1, top\_k = 100, top\_p = 0.8. The repetition penalty was set to 1.3.

The finetuned model uses the same GPT-2-XL model, which is further trained according to the steps described in Section 4.2. To explore how finetuning the model changes the quality of the output, generation on the test set was done using the same decoding parameters as the model without finetuning. The fine-tuned model generation was done without adding any primers to the context. Another set of reflections were generated with a higher temperature of 0.7 to understand how the fine-tuned model behaves when outputs are allowed to be more creative.

Reflections generated with the GPT-3 model were done using both eight and 12 primers. The reflections generated with eight primers provides a direct way to compare the few-shot reflection generation performance of GPT-3 and GPT-2, as eight was the maximum allowed for GPT-2. The 12-primer method was used to explore how well GPT-3 can generate reflections when taking advantage of its much larger context window by adding more priming examples. The GPT-3 engine used for these experiments were run using the DaVinci engine from the OpenAI API [73]. The decoding parameters were set as described in Section 4.1.2.

## 6.2 Reflection Generation Results

### 6.2.1 Few-Shot Reflection Generation

A set of 100 prompt-response pairs from the dataset described in Section 3.1 was used to evaluate the generative models on their ability to produce acceptable reflections. When selecting the prompt-response pairs, we ensured that the same prompt-response pair did not exist as part of the priming set described in Section 3.2. This ensures that the model isn't given an unfairly helpful priming set.

Model Type	Human-Label Hit Rate
GPT-2 - 8 primers	0.54
GPT-3 - 8 primers	0.89
GPT-3 - 12 primers	0.74

**Table 6.1: Reflection Hit Rate (for 100 trials) by Model**

Table 6.1 gives the human-label hit rate when generating reflections using the GPT-2 model with eight primers, and with the GPT-3 model using both eight and 12 primers. The GPT-2 model achieves a 54% hit rate, which shows that with just a little priming, the language models succeed quite often. It seems sufficient to form the basis of a system that includes the output filtering function, as discussed in Chapter 5.

Table 6.1 also shows that both versions of the generation from the GPT-3 model outperform the GPT-2 model, achieving 74% and 89% hit rates. This comes at the price of the GPT-3 model being more difficult to use at the present time, due to its very large size and computational requirements. It can currently only be run on large-scale hardware. By contrast, the GPT-2 based models can be run locally on consumer-grade hardware, and smaller versions of it can even be run on lightweight machines.

Table 6.1 also shows a surprising performance difference between the GPT-3 model with 8 primers and the GPT-3 model with 12 primers, with the 12-primer version performing worse. This result highlights the importance of tuning input and decoding parameters for each model to the target task, as performance can vary significantly (and not in an intuitive way) based on the parameter values.

### 6.2.2 GPT-2 Fine-tuned Reflection Generation

Here we present the results of using the fine-tuned GPT-2 model. Taking the same set of 100 prompt-response pairs from the previous section, we use the fine-tuned GPT-2 model to generate reflections, which is shown in Table 6.2. The first row reports the human labeled reflection hit-rate of the GPT-2 model without finetuning from the previous section. The next row gives the performance of the fine-tuned model with all decoding parameters identical to the first model and shown in Section 4.1 but without using few-shot priming. This is referred to as ‘GPT-2-Fine-tuned-Native’. We also wanted to evaluate how well the finetuned model performs when the decoding encourages more creative outputs by generating 100 reflections with the decoding parameter temperature value set to 0.7, shown in ‘GPT-2-Fine-tuned – Native - 0.7 Temp’.

<b>Model Type</b>	<b>Human-Label Hit Rate</b>	<b>Percentage Increase vs. Few-Shot</b>
GPT-2 - 8 primers	0.54	-
GPT-2-Fine-tuned – Native - 0.1 Temp	0.80	26%

GPT-2-Fine-tuned – Native - 0.7 Temp	0.66	12%
---	------	-----

**Table 6.2: Reflection Hit Rate (for 100 trials) for Fine-Tuned Models**

Table 6.2 indicates a 26% increase in hit rate when generating reflections on the fine-tuned model compared to the model that uses few-shot learning with the same decoding parameters. With the temperature increased to 0.7, the fine-tuned model still improves on the few-shot generation performance by 12%. We notice that the more creative fine-tuned model failed more often than the fine-tuned model with default parameters. This behaviour reflects the few-shot learning performance, which as we see in Figure 4.3, also performs poorly as temperature is increased. The fine-tuned GPT-2 achieves significantly improved performance over its few-shot only counterparts and is reasonably close to the performance of GPT-3’s reflection generation (given in Table 6.1), but with an order of magnitude lower computer memory and processing requirements. This lower hardware requirement can make a fine-tuned GPT-2 model a useful alternative to GPT-3 where hardware access is limited.

### 6.2.3 Generating Reflections with GPT-2 followed by an RQC Filter

Here we show the results of a system using the GPT-2 reflection generator, in combination with the RQC as a filter, to reduce the number of unacceptable reflections presented. In this system the RQC is responsible for providing a label as to whether a reflection is acceptable given the context. It takes as input the same prompt and response given to the model, as well as the generated completion (which is the reflection produced by GPT-2). If the RQC deemed the reflection as not acceptable, the generation system was then re-invoked to generate a new candidate reflection for the model.

This filtering setup assumes that we can stimulate the generative model to produce diverse and unique reflections while keeping the input context (the prompt and response) the same when producing multiple completions. So, even if the first completion produced is not an acceptable reflection, a second or third attempt at generation may produce a completion that is an acceptable reflection. We believe this is possible as the generation is simply selecting from a series of output probabilities after we pass the inputs to the model. There are many ways to stimulate a GPT-2 based model to produce different completions for the same context, and we use three techniques here. The first technique manipulates the input itself and is used when generating reflections using

few-shot priming. The strategy is to resample primers from the primer set for each generation attempt. Since the chosen primers between each run will be different, it will likely lead to differences in the generation at the output stage as the output probabilities will be different. The second strategy is used when we are not using few-shot priming, instead we perturb the sampling at the decoding stage, so for each generation a different random seed is set so that the sampling changes for each attempt. This method is used when generating with the fine-tuned model at low temperature, such as the 0.1 temperature version reported in Table 6.3. The final strategy is to simply change the decoding parameters to make the output distributions more uniform, which can be done by using a high temperature value. A higher temperature value tends to flatten the probability differences between the highest and lowest probability words, making it more likely to see different sequences when sampling. The performance of the fine-tuned model with temperature at 0.7 is also reported in Table 6.3

We place a limit of five attempts to generate an acceptable reflection, to limit computational cost and avoid a possible infinite loop. If no generated reflection was deemed acceptable by the RQC, the fifth reflection generated is passed through as the output.

It is possible to approximate the hit rate of the combined generation and RQC filtering process. Under the (inexact) assumption that there is no limit of 5 filtering operations (and the generation/reflection loop would continue until successful), the approximate hit rate can be determined based on the generator’s native hit rate (as shown in the GPT\_2 row of Table 6.1) and the true negative rate (the specificity) of the RQC as follows:

$$\text{CombinedHitRate} = 1 - (1 - \text{NativeHitRate})(1 - \text{RQC\_Specificity}) \quad (1)$$

The *NativeHitRate* when using the GPT-2 model is 0.54 (from Table 6.1), while the *RQC\_Specificity* for the RQC model with BERT-base MSRP+Reflection Fine-tuned is 0.71 (as given in Table 5.1). Using these two values we obtain an estimated *CombinedHitRate* for the few-shot GPT-2 model of 0.87. We can also compute an approximate hit rate for the ‘GPT-2-Fine-tuned – Native’ model: From Table 6.2 its *NativeHitRate* is 0.80, and so its *CombinedHitRate* of is approximated at 0.94.

Table 6.3 gives the results of the filtered generation process for both the few-shot and fine-tuned models. Here the RQC filtering loop was limited to a maximum of five cycles for each input, and if no reflection was deemed acceptable by the RQC, then the 5<sup>th</sup> reflection is the one used. The



same 100 examples comprising the test set for Table 6.1 and 6.2 are used as input. For the few-shot GPT-2 model, the hand-labelled reflection generation hit rate was improved from 0.54 to 0.73 when using the RQC filtering. This is a significant improvement on the hit rate and makes the use of a reflection filter a practical way to improve performance.

The filtered hit-rates achieved with the few-shot model is less than the estimated value of 0.87, perhaps in part due to the 5 filter-cycle limit. GPT-2 w. RQC Filtering loop failed to pass the filter after 5 attempts for 3 of the test samples. If those three poor quality results had been acceptable, the hit rate would increase to 0.77. With filtering, a few-shot GPT-2 model approaches the performance of the fine-tuned GPT-2 model. The fine-tuned GPT-2 model does not improve significantly with the filtering. In cases where fine-tuning a GPT-2 model is infeasible, one can alternatively train a classifier, which is often smaller and requires less computing power, to achieve similar end-user reflection hit-rate when using the filtering strategy.

Table 6.3 shows that applying the filtered generation process with GPT-2-Fine-tuned – Native from Table 6.2, did not significantly improve the hit-rate of the native model. The hit rate increased from 0.80 to 0.81. This is likely due to the low temperature, which makes it less likely for the outputs over multiple tries to be significantly different to each other as less likely tokens are selected more rarely when sampling. Using a higher temperature value with the fine-tuned model did show improvement from the native hit-rate shown in Table 6.2 by going from 0.66 to 0.75 but did not achieve the higher hit-rates of the standard finetuned model since the baseline hit-rate without filtering was significantly lower at this temperature.

<b>Model/Process</b>	<b>Human Label Reflection Hit Rate</b>
GPT-2	0.54
GPT-2 w. RQC Filtering loop	0.73
GPT-2-Fine-tuned – Native - 0.7 Temp w. RQC Filtering Loop	0.75
GPT-2 Finetuned – Native – 0.1 Temp w. RQC Filtering Loop	0.81

**Table 6.3: GPT-2 Reflection Generation Rate with and without Filtering**

## 6.3 Qualitative Discussion of Reflection Generated Reflections

The quantitative results above indicate that we can achieve some success generating MI-consistent reflections. Here we review specific examples of the generated outputs qualitatively, to provide a sense of what is promising about the results, and what still needs work.

Table 6.4 gives a set of ten acceptable (by human labeling) reflections that were generated from the models in Section 6.2, when producing the data for Table 6.1, 6.2, and 6.3. Each row of the table provides the input prompt and response, as well as the generated reflection, and which specific model produced it.

One can informally observe that the reflections in Table 6.4 do read as if a human created them and could be appropriate statements that a therapist could use. It can also be observed that the model appears robust when dealing with grammatically incorrect inputs - Examples 1, 5, and 7 have spelling errors in the responses, but the reflections are still produced with correct grammar and address the responses appropriately.

#	Prompt	Response	Reflection	Model
1	Think back to the time when you were able to prevent yourself from smoking. What made it different from when you did smoke?	i dont want to embarse my kids in front of their friends	You didn't want them to see how much you struggled with this problem.	GPT-2
2	I want you to describe a time or situation where you experienced the reason "Relationship Problem" and it did not lead you to smoke	I refused her request to stop at the gas station to buy cigarettes on the way to work. I didn't want one and wanted to demonstrate it to her.	You are making a conscious decision not to smoke.	GPT-3 - 8 Primers
3	Let me see if I understand, smoking is helpful because it provides a useful distraction for you. Are there other things that are good about smoking? If so, please tell me	I'm able to meet new people	You enjoy meeting new people when you smoke.	GPT-3 - 8 Primers
4	Think back to the time when you were able to prevent yourself from smoking. What made it	I was very stressed at work	You've been working long hours lately.	GPT-2 with RQC Filter

	different from when you did smoke?			
5	Think back to the time when you were able to prevent yourself from smoking. What made it different from when you did smoke?	I felt like I was affecting my chances at having children	You're worried about the impact on your fertility.	GPT-3 - 8 Primers
6	Let me see if I understand, smoking is helpful because it provides a useful distraction for you. Are there other things that are good about smoking? If so, please tell me	I'm able to meet new people	You met someone interesting.	GPT-2 with RQC Filter
7	Think back to the time when you were able to prevent yourself from smoking. What made it different from when you did smoke?	I was feeling good about myself and I had more determination.	You felt better than before because of how much effort you put into stopping	GPT-2-Fine-tuned - Native
8	Think back to the time when you were able to prevent yourself from smoking. What made it different from when you did smoke?	For this last instance I felt more in control	You feel like things are going better now than before	GPT-2-Fine-tuned – Native
9	Please describe a time where you had a compelling feeling to smoke and actually smoked	when i am stressed at work	You felt like smoking would help relieve your stress.	GPT-2-Fine-tuned – Native - 0.7 Temp
10	Please describe a time where you were worried about the smell of cigarettes and you didn't smoke	If I went to a formal meeting	The smells are too strong for most people.	GPT-2-Fine-tuned – Native - 0.7 Temp

**Table 6.4: Acceptable and Better Reflection Examples**

Although all examples in Table 6.4 are acceptable MI-consistent reflections, there is still a variation in the level of each reflection, with a mix of simple and complex reflections. Recall that complex reflections not only follow from the response but has an insight or a suggestion to move the conversation forward by drawing on ideas and experiences that have not been spoken by the patient. Consider Example 1, where the patient expresses the desire to not embarrass their children by smoking in front of the children's friends. The reflection suggests a deeper struggle the patient may be facing about their smoking. This kind of reflection often encourages a patient to contemplate further in a specific direction related to their previous response, which is one of the key goals of MI-based therapy.

Reflections 2,4, 6, and 10 in Table 6.4 also draw from an idea that was not present in the patient's response to help the patient contemplate the prompt further. Consider Example 2, which connects the idea of making a "conscious decision not to smoke" to the dynamic of an interaction during driving that is about buying cigarettes. Similarly, Example 4 makes a guess that the person was working "long hours" recently. A good therapist will make a guess like this to keep the patient contemplating cause and effect about their habit.

Reflections 7 through 10 in Table 6.4 also provide examples of complex reflections. These examples focus more on extrapolating the participant's feelings about their smoking, which is a different form of complex reflection that also encourages a participant to contemplate why they smoke.

Sometimes a reflection does not need to be complex to have a therapeutic effect [6]. A simple restatement of the response, in different words, is sufficient to build rapport with a patient and move the conversation forward. Reflections 3 and 5 in Table 6.4 are simple restatements of a patient's response.

These examples demonstrate that within the context of these prompts and responses, few-shot and fine-tuned generation with GPT-2 and (few-shot only) GPT-3 can produce reflections consistent with the ones a therapist may use in an MI session. The language sounds natural, follows many of the principles of MI, and fulfills the goal of moving the conversation forward.

However, not every reflection produced by these models can fulfill the therapeutic goal, and some examples of the unacceptable cases are shown in Table 6.5. There are multiple ways the models fail to generate acceptable reflections, but there were two categories of failures that were more common: The first category is statements that are coherent and follow from the context prompt and response but fail to adhere to MI principles. For example, reflection 1 in Table 6.5 responds to the patient's statement about having important bills to pay by stating that they would choose to buy cigarettes rather than paying bills. This misrepresents the patient's intent and is directly counter-productive to the smoking-cessation goal. Arguably, it is phrased in an accusatory tone, something that therapists are trained to avoid as much as possible as it produces discord in the patient-therapist relationship.

Reflection 3 in Table 6.5 dismisses the patient’s response about their attempt to resist smoking, stating rather unhelpfully that the patient needs to stop smoking completely to quit. This kind of reflection falls into the ‘expert trap’ [6] category, where the therapist offers direct guidance instead of letting the patient draw their own conclusions on their journey.

The second common category of failure occurs when the generated completion does not follow from the context prompt and response, and thus cannot be considered a reflection. This kind of failure occurs in reflections 2 and 4-10 in Table 6.5. Reflections 2 and 4 have the same context prompt and response with the reflection being generated by two different models, but both models failed to generate an acceptable reflection. In both instances, the generated reflections focused on addressing stress whereas the patient response was about not being able to remember an example where they experienced stress without it leading them to smoke.

Reflection 5 in Table 6.5 is a reflection that appears to focus on a specific word (‘flavour’) in the *prompt* and ignores the essence of the patient’s response. Reflection 6 fails to capture the context of the response and makes a leap about the patient’s boyfriend for which there is no evidence. Examples 7-10 also fail to capture the context of the response in their reflections, and produce reflections with logical leaps that cannot reasonably occur given the context of the response.

Beyond these two categories of failure, there are some rare instances where the models would produce incoherent reflections or just produce a blank reflection. These types of failure case were not present in the reflections produced from this test set but were observed with a specific combination of decoding parameters.

The failure cases show that the reflection generation is still imperfect and there is significant room to improve. One positive aspect of these failure cases is that they are less about actual grammar or language issues, and more about the nuance of human conversation, suggesting that if the models had more exposure to human experiences (through more diverse training data) some of the failure cases may be addressable. Including a filter like the RQC can help identify these failure cases, although as we see from the ‘RQC Label’ column in Table 6.5, it is only able to detect some of the failures and thus has room to be improve.

#	Prompt	Response	Reflection	Model	RQC Label
1	Think back to the time when you were able to prevent yourself from	The difference was that I was	When you have a choice between spending money	GPT-2 with RQC Filter	0

	smoking. What made it different from when you did smoke?	paying important bills	on cigarettes vs other things, you will choose cigarettes over those other things.		
2	Please describe a time where you experienced stress and didn't end up smoking	i cant really its been so long	Stress causes anxiety which leads to depression.	GPT-2 with RQC Filter (Case where 5-cycle limit was hit)	0
3	Think back to the time when you were able to prevent yourself from smoking. What made it different from when you did smoke?	it feels good to know i can resist if i try	The only way to stop smoking is to quit completely	GPT-2	0
4	Please describe a time where you experienced stress and didn't end up smoking	i cant really its been so long	You feel stressed about smoking.	GPT-3 - 8 Primers	1
5	Please describe a time where you dreaded tasting the flavour of smoking a cigarette and you didn't end up smoking	plenty of times just sat at home, especially when i already dont feel well	you know that there are many flavours out there	GPT-2 with RQC Filter	1
6	Please describe a situation where you thought about being a good role model and that cause you not to smoke	on my first date with my current boyfriend	You found someone who smokes and he/she doesn't judge you	GPT-2	1
7	Ok, so you associate Smell as something negative about smoking Please describe a time where you were worried about the smell of cigarettes but ended up smoking	it was after the news from my grandmother	Your mother's death had an impact on you that lasted for years.	GPT-2-Fine-tuned – Native	1
8	Please describe a situation where you thought about wanting to be a good role model but ended up smoking	On my way to a job interview	You worry that people will notice your cigarette odor and make fun of you for it. This is why many smokers try to cover their stinky clothing with	GPT-2-Fine-tuned – Native	1

			bandanas or other cloths before going out in public		
9	Please describe a time where you experienced stress and ended up smoking	Whenever I argue with my son	You feel stressed by being unable to control how much nicotine is in your system. This can lead to addiction	GPT-2-Fine-tuned – Native - 0.7 Temp	1
10	Think back to the time when you were able to prevent yourself from smoking. What made it different from when you did smoke?	i was not able to see friends from different departments within my work enviroment	Your colleagues don't want to be seen with someone who smokes	GPT-2-Fine-tuned – Native - 0.7 Temp	0

**Table 6.5: Not Acceptable Reflection Examples**

## Chapter 7

# Deploying a Reflection Generation System

This Chapter describes the deployment of a reflection generation system making use of the few-shot model one described in Chapter 6. The system is hosted on Amazon Web Services (AWS) [74]. The first Section describes how the reflection generation software was built and how it was deployed using AWS. The second Section describes how the API was used on a web application where we asked a participant their reasons for smoking and produced a reflection on their response.

### 7.1 Reflection Generation Microservice

Over the past decade, cloud providers have made it significantly easier to deploy and host software on the internet, and use that software to fulfil functions in other programs [75] over the web. Cloud vendors such as Amazon Web Services (AWS) [74], Azure [75], and Google Cloud Platform (GCP) [76] have produced large scale infrastructure that can be used to deploy software with relative ease. The cloud systems help reduce latency, increase scalability, and reduce cost [75]. Although all three cloud providers have the functionality required to deploy a reflection generation model as an API, AWS provides the most flexibility on how the software is designed and on what kind of hardware we can run our system. At the time of development, AWS was also the lowest-cost option out of the three major providers.

The reflection generation system was deployed as a microservice [77], which means that each component of the software is hosted as different services accessible over the web. Communication between microservices and other web applications were conducted through Representational state transfer (RESTful) operations [78]. The reflection generation API was its own microservice. It was also used as part of a full chatbot that can conduct an MI conversation, that was recently deployed beyond the scope of this thesis [79]. Section 7.2 also shows an example software using this microservice as a demonstration.

The core functionality of the reflection generation microservice is to take a prompt and a response string as part of its inputs and return a generated reflection through a GET operation. The code to choose which model to use, the number of primers, and decoding strategies are all be internal to the program itself and not exposed as parameters or options through the API. For this



demonstration, we used a GPT2-XL model from the Hugging Face Transformers library [52], and generated reflections using the few-shot learning strategy, whose performance is reported in Chapter 6. The API itself is built and defined using Connexion [80], a package that integrates the OpenAPI 3.0 specification [73] and other python packages within itself to allow us to build a web server using Python without having to code the barebones HTML required to make and receive API calls. Using Connexion allows us to simply run our Python program, and Connexion routes any API calls made to the server hosting our program to the appropriate Python function simply based on the URL that is called. Connexion only requires us to define this mapping from URL to python function in a configuration file using the YAML [81] format. Any parameters that are required by the function are automatically extracted from URL parameters in the API call by Connexion. These are associated with the appropriate function parameter. For our system these parameters are the prompt and response texts. Connexion is also used to define the API response, which is mapped to take the function output and send the output back to the caller through a JSON packet. This is triggered by a GET request and it returns the reflection that is produced by our reflection generator function in our python code. The full API configuration YAML file is shown in Figure 7.1

The microservice is deployed using AWS' Elastic Beanstalk (EBS) [82]. EBS is an easy-to-use service that takes the Python code and configuration information and runs it on an AWS instance that is automatically provisioned by EBS. A key feature of EBS is its ability to automatically scale the number of instances of servers when the demand for the microservice grows. This is quite relevant to our ultimate goal of deploying and testing a full chatbot, to enable good response times for multiple simultaneously conversations.

Since we are using a large model (GPT-2-XL) and we would like each individual reflection generation to be returned in a few seconds, GPU acceleration was required. Using only the CPU (a custom Intel Cascade Lake Xeon chip designed for AWS [83]), each generation with GPT-2-XL requires approximately 100 seconds. When using GPU acceleration with Nvidia T4 GPUs [84], which are optimized for model inference tasks, and storing the model in the GPU memory, the time to generate rarely exceeds 10 seconds. We expect that as AWS updates the GPU instances available with more recent GPUs, the speed of generation can be made even faster, though it is already at a latency that is usable in downstream chatbot conversation systems [79]. The instance employed the AWS Deep Learning Conda Amazon Machine Instance (AMI) [25], which is pre-

installed with Amazon Linux 2 and is configured to come pre-installed with PyTorch [85] and Anaconda [86] packages. To ensure that the instance initializes correctly and that the model can be loaded correctly, we must also ensure that the EBS volume size is 200 GB or more, to account for the installed packages and for the downloaded GPT-2 model.

The automatic scaling of the number of instances is controlled by setting the minimum and maximum number of concurrent EBS instances than can be run. Within these constraints EBS will spin up or shut down copies of our microservice based on how much the API is being called while still being accessed through the same URL. Load balancing, or the logic that determines to which instance to route an API call based on how busy the instance is currently, is handled by AWS itself and can be configured in the EBS settings.

```

openapi: '3.0.0'
info:
  version: '0.1'
  title: 'AWS Elastic Beanstalk GPT-2 NLG API for Reflection Generation'
servers:
  - url: /v1
paths:
  /generate:
    get:
      operationId: application.generate
      responses:
        '200':
          description: Generate text from a seed.
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Generated'
      parameters:
        - in: query
          name: seed
          schema:
            type: string
            default: "Prompt: This is a sample question.\nResponse: This is a sample response\n"
            required: true
            description: The seed text to generate from.

components:
  schemas:
    Generated:
      type: string
      example: "Prompt: This is a sample question.\nResponse: This is a sample response\nReflection: This is a sample reflection"

```

**Figure 7.1: Reflection Generation API Schema**

Once set up is complete, the EBS instance status will show as ‘OK’, meaning this instance (or the multiple underlying instances when we allow for it) is now ready to be used. AWS provides a base

URL for the microservice, and this base URL can now be combined with the path from the API schema in Figure 7.1 to construct the URL that needs to be used when we want to use this microservice. To revise the code run by the microservice, we simply need the updated code to be uploaded to EBS, and it will update each instance of the microservice with the updated code automatically.

## 7.2 Reflection Generation Microservice

The microservice described in Section 7.1 is designed to be used as a standalone reflection generation service and can be integrated to be used in many web applications. To test this microservice, we built a website that asks a participant about a specific smoking behaviour they are looking to change, collects their response, and presents a reflection to their response. Figure 7.2 provides screenshots of the website with a walkthrough of a user answering the question and receiving a reflection.

**Figure 7.2: Example Reflection Generation Application**

This website was built using JavaScript React and hosted on AWS Amplify [25]. To integrate the reflection generation microservice, we need to use a package that implements a RESTful [78] API service (in this case we used Axios [87]), the URL of the microservice (provided by EBS as described in Section 7.1), and format the prompt and response appropriately in the body of the

GET request. Once the GET request is sent, the application waits for a response from the server in the background which contains the status of the call (providing a code 200 return if the API call is successful) and the response body which contains the generated reflection.

Figure 7.3 shows how this is implemented in the above web application to get a reflection from the reflection generation microservice.

```
const getReflection = async () => {
  var fetchString = reflectionUrl + "Prompt: " + question + urlNewlineChar + "Response: " + response + urlNewlineChar
  var i;
  for (i=0; i < numberOfReflections; i++){
    axios.get(fetchString)
      .then((res) => {
        console.log("Call success")
        console.log(res);
        setReflections(reflections => [...reflections, res['data']])
      })
  }
}
```

**Figure 7.3: Reflection Generation Microservice Call Code Snippet**

This example application demonstrates how a microservice can easily be integrated into a web application. In this example, we ask a question about a specific smoking behaviour and use the microservice to produce a reflection on the response with only a few lines of code. Although this web application does not conduct a full MI conversation, future implementations expand on the infrastructure by updating the reflection generation microservice with better generation [79] and building other microservices to emulate other MI-consistent therapeutic behaviours (such as summarization or affirmations). This chapter provides a small demonstration of how such a system may be built cheaply and efficiently using cloud infrastructure and software tools that are readily available.

## Chapter 8

# Conclusions, Limitations, and Future Work

### 8.1 Conclusions

This work is a first step towards moving therapeutic chatbots from responding repetitively and predictably, to a form where they can respond with human-like specificity and understanding. We demonstrated that transformer-based language models can be effective at producing MI-consistent reflections when responding to participant statements within a specific domain, even when working in a low-data environment. We also showed that the larger GPT-3 model produces acceptable reflections more frequently than the GPT-2 XL based model, but that the difference can be minimized if we have sufficient data to fine-tune the GPT-2 XL model. As one method for mitigating unacceptable reflections, we trained a reflection quality classifier (RQC) to determine whether a reflection is acceptable and showed that it has fair correlation to human labels. We also demonstrated that a GPT-2 model can be used with the RQC as a filter to build a system that can produce reflections at a better rate than just using GPT-2 with few-shot learning. This comes at the cost of slightly higher memory usage and longer execution time needed to generate extra reflections. Finally, we demonstrated one way of deploying our custom models and scripts using a microservice architecture deployed through AWS.

### 8.2 Limitations

In this work we have attempted to show how few-shot generation might be used to generate reflections within MI-style smoking cessation conversations, and we have shown how fine-tuning the same models might lead to better reflection generation. The most significant limitation in this work is that the reflections were evaluated within the scope of the specific questions and responses from the smoking cessation dataset described in Chapter 3, and we cannot conclude that the approach is generalizable beyond that dataset. However, since we've employed few-shot learning there is some hope that the few-shot technique would generalize to other topics and be usable to replicate other MI skills. The fine-tuning strategy would also likely translate to other domains and MI skills if we were able to find a similar amount of relevant training data.

Another limitation of these results is the low test-set sample size, which was necessary due to the significant labour required to label the reflections produced by the generative models. Since trained MI-experts are difficult to find and their availability is limited, only the primer set was labeled and edited by a trained MI expert. All other labeling was done by the author and non-MI-expert participants, whose experience with labeling MI came from readings [6] and brief consultations with MI practitioners. The non-MI-expert labels are also likely to be less clinically precise, which is a reason why the reflection labeling was left as a binary label and not on a more granular scale that differentiates between the different types and levels of reflections.

### 8.3 Future Work

There is significant work to be done on evaluating the reflection generation process and how effective it will be in the context of different prompts and responses. Further, we would also like to be able to control whether a simple or more complex reflection will be produced, as different situations in a full MI conversation may call for one or the other. Another area that needs more work is on using the techniques in this thesis to build a generative system for the other therapist skills used in an MI conversation, for instance to generate affirmations or summary statements.

Finally, the reflection generation system was evaluated solely on its ability to produce acceptable MI-consistent reflections and not on therapeutic impact. To evaluate the latter, the reflection generation system needs to be integrated into a full chatbot and measured as an intervention.

## Bibliography

- [1] Simon Davis, *Community mental health in Canada, revised and expanded edition: Theory, policy, and practice.*: UBC Press, 2013.
- [2] Gregory L. Fricchione et al., "Capacity Building in Global Mental Health: Professional Training," *Harvard Review of Psychiatry*, vol. 20, pp. 47-57, 2012, PMID: 22335182. [Online]. <https://www.tandfonline.com/doi/abs/10.3109/10673229.2012.655211>
- [3] N. Kates et al., "The evolution of collaborative mental health care in Canada: A shared vision for the future," *Canadian Journal of Psychiatry*, vol. 56, p. 11, 2011.
- [4] G. Whitfield, *Cognitive behavioural therapy explained. Vol. xii.*. 2007.
- [5] Laura Nisbet, "Dialectical Behavioural Therapy," in *Child and Adolescent Mental Health.*: CRC Press, 2021, pp. 448–456.
- [6] William R. Miller and Stephen Rollnick, *Motivational interviewing: Helping people change.*: Guilford press, 2012.
- [7] David D. Luxton, "Chapter 1 - An Introduction to Artificial Intelligence in Behavioral and Mental Health Care," in *Artificial Intelligence in Behavioral and Mental Health Care*, David D. Luxton, Ed. San Diego: Academic Press, 2016, pp. 1-26. [Online]. <https://www.sciencedirect.com/science/article/pii/B9780124202481000015>
- [8] Daniel Adiwardana et al., Towards a Human-like Open-Domain Chatbot, 2020.
- [9] Tom B. Brown et al., Language Models are Few-Shot Learners, 2020.
- [10] Andrea Madotto, Zihan Liu, Zhaojiang Lin, and Pascale Fung, Language Models as Few-Shot Learner for Task-Oriented Dialogue Systems, 2020.

- [11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019.
- [12] Alec Radford et al., "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, p. 9, 2019.
- [13] Alain Brailion and Françoise Taiebi, "Practicing "Reflective listening" is a mandatory prerequisite for empathy," *Patient Education and Counseling*, vol. 103, pp. 1866-1867, 2020. [Online]. <https://www.sciencedirect.com/science/article/pii/S0738399120301828>
- [14] Erik Rautalinko, "Reflective listening and open-ended questions in counselling: Preferences moderated by social skills and cognitive ability," *Counselling and Psychotherapy Research*, vol. 13, pp. 24-31, 2013. [Online]. <https://doi.org/10.1080/14733145.2012.687387>
- [15] Carolyn J. Heckman, Brian L. Egleston, and Makary T. Hofmann, "Efficacy of motivational interviewing for smoking cessation: a systematic review and meta-analysis," *Tobacco Control*, vol. 19, pp. 410–416, 2010. [Online]. <https://tobaccocontrol.bmj.com/content/19/5/410>
- [16] Ina Diener, Mark Kargela, and Adriaan Louw, "Listening is therapy: Patient interviewing from a pain science perspective," *Physiotherapy Theory and Practice*, vol. 32, pp. 356-367, 2016, PMID: 27351690. [Online]. <https://doi.org/10.1080/09593985.2016.1194648>
- [17] Fahad Almusharraf, Jonathan Rose, and Peter Selby, "Engaging Unmotivated Smokers to Move Toward Quitting: Design of Motivational Interviewing–Based Chatbot Through Iterative Interactions," *J Med Internet Res*, vol. 22, p. e20251, Nov. 2020. [Online]. <https://www.jmir.org/2020/11/e20251>
- [18] Francisco Jódar-Sánchez et al., "Using the Social-Local-Mobile App for Smoking Cessation in the SmokeFreeBrain Project: Protocol for a Randomized Controlled Trial.," *JMIR research protocols*, vol. 7, no. 12, p. e12464, Dec. 2018.



- [19] Daniel M. Low, Kate H. Bentley, and Satrajit S. Ghosh, "Automated assessment of psychiatric disorders using speech: A systematic review," *Laryngoscope Investigative Otolaryngology*, vol. 5, pp. 96-116, 2020. [Online].  
<https://onlinelibrary.wiley.com/doi/abs/10.1002/lio2.354>
- [20] Johanna Nurmi et al., "Engaging Users in the Behavior Change Process With Digitalized Motivational Interviewing and Gamification: Development and Feasibility Testing of the Precious App," *JMIR Mhealth Uhealth*, vol. 8, p. e12884, Jan. 2020. [Online].  
<https://mhealth.jmir.org/2020/1/e12884>
- [21] SoHyun Park et al., "Designing a Chatbot for a Brief Motivational Interview on Stress Management: Qualitative Case Study," *J Med Internet Res*, vol. 21, p. e12231, Apr. 2019. [Online]. <https://www.jmir.org/2019/4/e12231/>
- [22] Sarah I. Pratt et al., "Feasibility and effectiveness of an automated telehealth intervention to improve illness self-management in people with serious psychiatric and medical disorders.," *Psychiatric rehabilitation journal*, vol. 36, no. 4, pp. 297-305, Dec. 2013.
- [23] Taylor N. Stephens, Angela Joerin, Michiel Rauws, and Lloyd N. Werk, "Feasibility of pediatric obesity and prediabetes treatment support through Tess, the AI behavioral coaching chatbot," *Translational behavioral medicine*, vol. 9, pp. 440–447, 2019.
- [24] Iwan Gulenko, "Chatbot for IT Security Training: Using Motivational Interviewing to Improve Security Behaviour.," in *AIST (Supplement)*, 2014, pp. 7–16.
- [25] Amazon. (2021, Nov) Amazon Web Services. [Online]. <https://aws.amazon.com/>
- [26] Siqi Shen, Charles Welch, Rada Mihalcea, and Verónica Pérez-Rosas, "Counseling-Style Reflection Generation Using Generative Pretrained Transformers with Augmented Context," in *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 1st virtual meeting, July 2020, pp. 10–20. [Online].  
<https://aclanthology.org/2020.sigdial-1.2>

- [27] Zihang Dai et al., Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context, 2019.
- [28] Brian L. Burke, Hal Arkowitz, and Marisa Menchola, "The efficacy of motivational interviewing: a meta-analysis of controlled clinical trials.," *Journal of consulting and clinical psychology*, vol. 71, p. 843, 2003.
- [29] Todd Olmstead, Kathleen M. Carroll, Monica Canning-Ball, and Steve Martino, "Cost and cost-effectiveness of three strategies for training clinicians in motivational interviewing," *Drug and alcohol dependence*, vol. 116, pp. 195–202, 2011.
- [30] Gobinda G. Chowdhury, "Natural language processing," *Annual review of information science and technology*, vol. 37, pp. 51–89, 2003.
- [31] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman, "Using Latent Semantic Analysis to Improve Access to Textual Information," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 1988, pp. 281–285. [Online]. <https://doi.org/10.1145/57167.57214>
- [32] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
- [33] Yunsick Sung, Sejun Jang, Young-Sik Jeong, Jong Hyuk, and others, "Malware classification algorithm using advanced Word2vec-based Bi-LSTM for ground control stations," *Computer Communications*, vol. 153, pp. 342–348, 2020.
- [34] Lizhong Xiao, Guangzhong Wang, and Yang Zuo, "Research on patent text classification based on word2vec and LSTM," in *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 1, 2018, pp. 71–74.

- [35] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur, "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*, 2010.
- [36] Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur, "Extensions of recurrent neural network language model," in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2011, pp. 5528–5531.
- [37] Paweł Budzianowski and Ivan Vulić, "Hello, it's GPT-2—how can I help you? towards the use of pretrained language models for task-oriented dialogue systems," *arXiv preprint arXiv:1907.05774*, 2019.
- [38] Baolin Peng et al., "Few-shot natural language generation for task-oriented dialog," *arXiv preprint arXiv:2002.12328*, 2020.
- [39] Jacqueline Brixey et al., "Shihbot: A facebook chatbot for sexual health information on hiv/aids," in *Proceedings of the 18th annual SIGdial meeting on discourse and dialogue*, 2017, pp. 370–373.
- [40] Yizhe Zhang et al., "DialogPT: Large-Scale Generative Pre-training for Conversational Response Generation," *CoRR*, vol. abs/1911.00536, 2019. [Online].  
<http://arxiv.org/abs/1911.00536>
- [41] Ashish Vaswani et al., "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.
- [42] Katherine Elkins and Jon Chun, "Can GPT-3 pass a writer's Turing test," *Journal of Cultural Analytics*, vol. 2371, p. 4549, 2020.
- [43] Tassilo Klein and Moin Nabi, "Learning to answer by learning to ask: Getting the best of gpt-2 and bert worlds," *arXiv preprint arXiv:1911.02365*, 2019.

- [44] Shervin Minaee et al., "Deep Learning–Based Text Classification: A Comprehensive Review," *ACM Comput. Surv.*, vol. 54, Apr. 2021. [Online].  
<https://doi.org/10.1145/3439726>
- [45] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving language understanding by generative pre-training," 2018.
- [46] Peter J. Liu et al., "Generating wikipedia by summarizing long sequences," *arXiv preprint arXiv:1801.10198*, 2018.
- [47] Vinci Liu and James R. Curran, "Web text corpus for natural language processing," in *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006.
- [48] Jay M. Patel, "Introduction to Common Crawl Datasets," in *Getting Structured Data from the Internet.*: Springer, 2020, pp. 277–324.
- [49] Yukun Zhu et al., "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books," in *The IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015.
- [50] Ludovic Denoyer and Patrick Gallinari, "The wikipedia xml corpus," in *International Workshop of the Initiative for the Evaluation of XML Retrieval*, 2006, pp. 12–19.
- [51] Leiming Yan, Yuhui Zheng, and Jie Cao, "Few-shot learning for short text classification," *Multimedia Tools and Applications*, vol. 77, pp. 29799–29810, 2018.
- [52] Thomas Wolf et al., *HuggingFace's Transformers: State-of-the-art Natural Language Processing*, 2020.
- [53] Davis Yoshida, Allyson Ettinger, and Kevin Gimpel, *Adding Recurrence to Pretrained Transformers for Improved Efficiency and Context Size*, 2020.

- [54] Zhilin Yang et al., XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020.
- [55] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi, "The curious case of neural text degeneration," *arXiv preprint arXiv:1904.09751*, 2019.
- [56] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher, CTRL: A Conditional Transformer Language Model for Controllable Generation, 2019.
- [57] Matthew E. Peters et al., "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.
- [58] Russell Fulmer, Angela Joerin, Breanna Gentile, Lysanne Lakerink, and Michiel Rauws, "Using psychological artificial intelligence (Tess) to relieve symptoms of depression and anxiety: randomized controlled trial," *JMIR mental health*, vol. 5, p. e64, 2018.
- [59] Fahad Almusharraf, "Motivating smokers to quit through a computer-based conversational system," University of Toronto (Canada), Ph.D. dissertation 2019.
- [60] Verónica Pérez-Rosas, Rada Mihalcea, Kenneth Resnicow, Satinder Singh, and Lawrence An, "Building a Motivational Interviewing Dataset," in *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, San Diego, CA, USA, June 2016, pp. 42–51. [Online]. <https://aclanthology.org/W16-0305>
- [61] OpenAI, OpenAI API, Nov. 2021.
- [62] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin, "A neural probabilistic language model," *The journal of machine learning research*, vol. 3, pp. 1137–1155, 2003.
- [63] Diederik P. Kingma and Jimmy Ba, Adam: A Method for Stochastic Optimization, 2014.

- [64] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He, "DeepSpeed: System Optimizations Enable Training Deep Learning Models with Over 100 Billion Parameters," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2020, pp. 3505–3506. [Online]. <https://doi.org/10.1145/3394486.3406703>
- [65] Amazon Web Services. (2021, Nov.) Amazon SageMaker. [Online]. <https://docs.aws.amazon.com/sagemaker/index.html>
- [66] Thomas Kluyver et al., "Jupyter Notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, 2016, pp. 87-90.
- [67] Yonghui Wu et al., "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," *CoRR*, vol. abs/1609.08144, 2016. [Online]. <http://arxiv.org/abs/1609.08144>
- [68] William B. Dolan and Chris Brockett, "Automatically Constructing a Corpus of Sentential Paraphrases," in *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. [Online]. <https://www.aclweb.org/anthology/I05-5002>
- [69] Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth, "Recognizing textual entailment: Rational, evaluation and approaches—erratum," *Natural Language Engineering*, vol. 16, pp. 105–105, 2010.
- [70] Martín Abadi et al., TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015, Software available from tensorflow.org.
- [71] Gilbert Becker, "Creating comparability among reliability coefficients: the case of Cronbach alpha and Cohen kappa," *Psychological reports*, vol. 87, pp. 1171–1182E, 2000.

- [72] Mary L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica*, vol. 22, pp. 276–282, 2012.
- [73] OpenAPI specification.
- [74] Jinesh Varia, Sajee Mathew, and others, "Overview of amazon web services," *Amazon Web Services*, vol. 105, 2014.
- [75] Charlotte Kotas, Thomas Naughton, and Neena Imam, "A comparison of Amazon Web Services and Microsoft Azure cloud platforms for high performance computing," in *2018 IEEE International Conference on Consumer Electronics (ICCE)*, 2018, pp. 1–4.
- [76] S. P. T. Krishnan and Jose L. Ugia Gonzalez, *Building your next big thing with google cloud platform: A guide for developers and enterprise architects.*: Springer, 2015.
- [77] Xabier Larrucea, Izaskun Santamaria, Ricardo Colomo-Palacios, and Christof Ebert, "Microservices," *IEEE Software*, vol. 35, pp. 96–100, 2018.
- [78] Cesare Pautasso, "RESTful web services: principles, patterns, emerging technologies," in *Web Services Foundations.*: Springer, 2014, pp. 31–51.
- [79] Ash Kumar, , November 2021, Private Communication.
- [80] Welcome to connexion's documentation!
- [81] Oren Ben-Kiki, Clark Evans, and Brian Ingerson, "Yaml ain't markup language (yaml™) version 1.1," *Working Draft 2008-05*, vol. 11, 2009.
- [82] Amazon Web Services, AWS Elastic Beanstalk - Amazon Web Services (AWS).
- [83] A. van der. Leij and M. Hofstra, Intel, 1998.
- [84] (2021, November) Nvidia Data Center Tesla T4. [Online]. <https://www.nvidia.com/en-us/data-center/tesla-t4/>

- [85] Adam Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.
- [86] Damien Rolon-Mérette, Matt Ross, Thaddé Rolon-Mérette, and Kinsey Church, "Introduction to Anaconda and Python: Installation and setup," *Python for Research in Psychology*, vol. 16, pp. S5–S11, 2016.
- [87] Getting started.
- [88] Wen Xu, Jing He, and Yanfeng Shu, "Transfer Learning and Deep Domain Adaptation," in *Advances and Applications in Deep Learning*, Marco Antonio Aceves-Fernandez, Ed. Rijeka: IntechOpen, 2020, ch. 3. [Online]. <https://doi.org/10.5772/intechopen.94072>
- [89] G. Whitfield and A. & Davidson, *Cognitive Behavioural Therapy Explained (1st ed.)*.: CRC Press., 2007.
- [90] Milton L. Wainberg et al., "Challenges and Opportunities in Global Mental Health: a Research-to-Practice Perspective.," *Current psychiatry reports*, vol. 19, no. 5, p. 28, May 2017.
- [91] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang, "How to fine-tune BERT for text classification?," in *China National Conference on Chinese Computational Linguistics*, 2019, pp. 194–206.
- [92] Timo Schick and Hinrich Schütze, "It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners," *arXiv preprint arXiv:2009.07118*, 2020.
- [93] Erik Rautalinko, Hans-Olof Lisper, and Bo Ekehammar, "Reflective listening in counseling: effects of training time and evaluator social skills," *American journal of psychotherapy*, vol. 61, pp. 191–209, 2007.



- [94] Manel Pladevall, George Divine, Karen E. Wells, Ken Resnicow, and L. Keoki Williams, "A randomized controlled trial to provide adherence information and motivational interviewing to improve diabetes and lipid control," *The Diabetes Educator*, vol. 41, pp. 136–146, 2015.
- [95] Do June Min, Verónica Pérez-Rosas, and Rada Mihalcea, "Evaluating Automatic Speech Recognition Quality and Its Impact on Counselor Utterance Coding," in *Proceedings of the Seventh Workshop on Computational Linguistics and Clinical Psychology: Improving Access*, Online, June 2021, pp. 159–168. [Online]. <https://aclanthology.org/2021.clpsych-1.18>
- [96] Jeroen Lakerveld et al., "Motivational interviewing and problem solving treatment to reduce type 2 diabetes and cardiovascular disease risk in real life: a randomized controlled trial," *International Journal of Behavioral Nutrition and Physical Activity*, vol. 10, pp. 1–9, 2013.
- [97] Cyril Goutte and Eric Gaussier, "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation," , vol. 3408, Apr. 2005, pp. 345-359.
- [98] Helen Frost et al., "Effectiveness of Motivational Interviewing on adult behaviour change in health and social care settings: A systematic review of reviews," *PLOS ONE*, vol. 13, pp. 1-39, Oct. 2018. [Online]. <https://doi.org/10.1371/journal.pone.0204890>
- [99] Sean Welleck et al., "Neural Text Generation with Unlikelihood Training," *CoRR*, vol. abs/1908.04319, 2019. [Online]. <http://arxiv.org/abs/1908.04319>
- [100] Dominique Bellenger et al., "Scaling in cloud environments," *Recent Researches in Computer Science*, vol. 33, pp. 145–150, 2011.
- [101] Carolyn J. Heckman, Brian L. Egleston, and Makary T. Hofmann, "Efficacy of motivational interviewing for smoking cessation: a systematic review and meta-analysis," *Tobacco control*, vol. 19, pp. 410–416, 2010.

