University of Denver
Digital Commons @ DU

Electronic Theses and Dissertations | Graduate Studies

8-2023

Controllable Language Generation Using Deep Learning

Rohola Zandie

Follow this and additional works at: https://digitalcommons.du.edu/etd

 Part of the Artificial Intelligence and Robotics Commons, and the Other Computer Engineering Commons


All Rights Reserved.

Controllable Language Generation Using Deep Learning

Abstract

The advent of deep neural networks has sparked a revolution in Artificial Intelligence (AI), notably with the creation of Transformer models like GPT-X and ChatGPT. These models have surpassed previous methods in various Natural Language Processing (NLP) tasks. As the NLP field evolves, there is a need to further understand and question the capabilities of these models. Text generation, a crucial part of NLP, remains an area where our comprehension is limited while being critical in research.

This dissertation focuses on the challenging problem of controlling the general behaviors of language models such as sentiment, topical focus, and logical reasoning. Controlling these properties influences language model generative processes in ways that enhance their emotional resonance, improve logical consistency, and maintain topical relevance.

To accomplish this objective I develop a rule-based Dialogue Management System (DMS), Program-R, which operates through the Ryan companionbot. Program-R incorporates controllable elements like sentiment and facial expression recognition. It uses these factors to choose the most fitting response from a set of pre-determined dialogues, offering a unique interaction experience.

Moving forward, I develop and assess EmpTransfo, a dialog system built on deep learning principles that can simultaneously comprehend emotions and generate responses. The proposed methodology is based on training a conditional language model using emotions. I further enrich the model by feeding emotion embedding into EmpTransfo, thereby setting an emotional context for language generation.

In the next method, I utilize topic modeling information and integrate it with a pretrained language model. By viewing topic probabilities as the prior and language model probabilities as the likelihood, the resulting probability for topical language generation becomes the posterior. Experimental results underscore that this topical language generation approach surpasses existing benchmarks in open text generation.

Finally, I expand the control over text generation to encompass abductive reasoning. Given incomplete observations from real-world situations, the aim is to modify the text generation process to reason about plausible hypotheses that could lead to the given observations. In this study, I propose a deep learning model that employs a novel approach of combining temporal commonsense reasoning for each observation from pre-trained models.

Document Type

Dissertation

Degree Name

Ph.D.

First Advisor

Mohammad H. Mahoor

Second Advisor

Alvaro Arias

Third Advisor

Haluk Ogmen

Keywords

Deep neural networks, Artificial intelligence, Natural language processing (NLP), Dialogue management system (DMS)

Subject Categories

Artificial Intelligence and Robotics | Computer Engineering | Computer Sciences | Engineering | Other Computer Engineering | Physical Sciences and Mathematics

Publication Statement

Copyright is held by the author. User is responsible for all copyright compliance.

CONTROLLABLE LANGUAGE GENERATION USING DEEP LEARNING

A Dissertation

Presented to

the Faculty of the Daniel Felix Ritchie School of Engineering and Computer Science

University of Denver

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

by

Rohola Zandie

August 2023

Advisor: Mohammad H. Mahoor, Ph.D

Author: Rohola Zandie
Title: CONTROLLABLE LANGUAGE GENERATION USING DEEP LEARNING
Advisor: Mohammad H. Mahoor, Ph.D
Degree Date: August 2023

Abstract

The advent of deep neural networks has sparked a revolution in Artificial Intelligence (AI), notably with the creation of Transformer models like GPT-X and ChatGPT. These models have surpassed previous methods in various Natural Language Processing (NLP) tasks. As the NLP field evolves, there is a need to further understand and question the capabilities of these models. Text generation, a crucial part of NLP, remains an area where our comprehension is limited while being critical in research.

This dissertation focuses on the challenging problem of controlling the general behaviors of language models such as sentiment, topical focus, and logical reasoning. Controlling these properties influences language model generative processes in ways that enhance their emotional resonance, improve logical consistency, and maintain topical relevance.

To accomplish this objective I develop a rule-based Dialogue Management System (DMS), Program-R, which operates through the Ryan companionbot. Program-R incorporates controllable elements like sentiment and facial expression recognition. It uses these factors to choose the most fitting response from a set of pre-determined dialogues, offering a unique interaction experience.

Moving forward, I develop and assess EmpTransfo, a dialog system built on deep learning principles that can simultaneously comprehend emotions and generate responses. The proposed methodology is based on training a conditional language model using emotions. I further enrich the model by feeding emotion embedding into EmpTransfo, thereby setting an emotional context for language generation.

In the next method, I utilize topic modeling information and integrate it with a pre-trained language model. By viewing topic probabilities as the prior and language model probabilities as the likelihood, the resulting probability for topical language generation becomes the posterior. Experimental results underscore that this topical language generation approach surpasses existing benchmarks in open text generation.

Finally, I expand the control over text generation to encompass abductive reasoning. Given incomplete observations from real-world situations, the aim is to modify the text generation process to reason about plausible hypotheses that could lead to the given observations. In this study, I propose a deep learning model that employs a novel approach of combining temporal commonsense reasoning for each observation from pre-trained models.

Acknowledgements

First and foremost, I wish to express my profound appreciation to my kind and supportive advisor, Dr. Mohammad H. Mahoor. His guidance, deep knowledge, and boundless generosity have been invaluable to me. Dr. Mahoor, your influence has profoundly shaped not just my research and career, but also had an immensely positive impact on my life.

Next, I extend my sincere thanks to my esteemed committee members, Dr. Haluk Ogmen, Dr. Kerstin Haring, and Dr. Alvaro Arias for your valuable time, constructive feedback, and support.

There are many individuals who I would like to acknowledge for their support throughout these years. To my colleagues, Hojjat Abdollahi, Jarid Sewierski, Mohsen Renani, the senior care facilities and their hardworking staff, specifically Eaton Senior Communities and Sarah Schoeder, and sweet and kind senior care residents who graciously agreed to participate in my studies. Thank you all!

I would also like to express my gratitude to my family, who have done everything they can to further my education. Equally, I am profoundly grateful to my partner, Alexandra Flowers, and her family, whose encouragement and belief in my abilities have been nothing short of inspirational.

Finally, I want to dedicate this work to my brother, Hadi, who ignited the flame of curiosity inside me.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

> "Civilization advances by extending the number of operations we can perform without thinking about them."
>
> **Alfred North Whitehead**

The research described in this dissertation focuses on the problem of controllable language generation in deep learning. More specifically, taking language models as probabilistic deep models of language generation the research question is: what are different ways to manipulate the parameters of language models to manifest a desired property such as emotiveness, topic relevance, and reasoning capabilities? The goal is to devise a set of methods to change the general behaviors of the language model generation process with low cost, high control, and utmost interpretability. Moreover, these techniques should possess a level of generalizability that extends beyond the constraints of any specific model or architecture. To achieve this goal the focus is on three aspects of language generation namely emotion, topic, and reasoning, and introduce methods that work on the architecture, training, or sampling of the language models.

This introductory chapter provides a concise historical background on the significance of modeling natural language, its evolution, and the current challenges and limitations that this dissertation aims to address. It lays out a foundation, aiding in comprehending our position within a swiftly transforming field teeming with novel concepts.

Finally, this dissertation's scope will be outlined, followed by a concise summary of its structure and organization.

1.1 Background

Language lies at the heart of our understanding of the world and ourselves. Many philosophers and cognitive scientists advocate for the paramount role of language in our cognition and the overall frameworks through which one interprets the world. Furthermore, the issue of human intelligence is intricately linked with language.

Alan Turing, the pioneer in the field of Artificial Intelligence, proposed the imitation game (Machinery, 1950), commonly known as the Turing test, as a test of intelligence. This test is based entirely on the understanding and usage of language. While the term "language" can encompass formal structures like mathematics or programming languages, Turing's focus was generally on daily human communication. This is where the intricacies truly emerge, as unlike formal or artificial languages, which adhere to recursive grammars with limited application, natural language appears not to have such a rigid structure. It embodies a fluidity and complexity that significantly deviates from formal structures.

Following Turing, the central question for many computer scientists in the field of artificial intelligence became: How to model language? Inspired by the surge of formal systems in mathematics and logic at the beginning of the twentieth century, numerous linguists and computer scientists began to model language as a formal system, complete with grammars and syntactic rules. Noam Chomsky, in "Syntactic Structure" (Chomsky, 2009) introduces

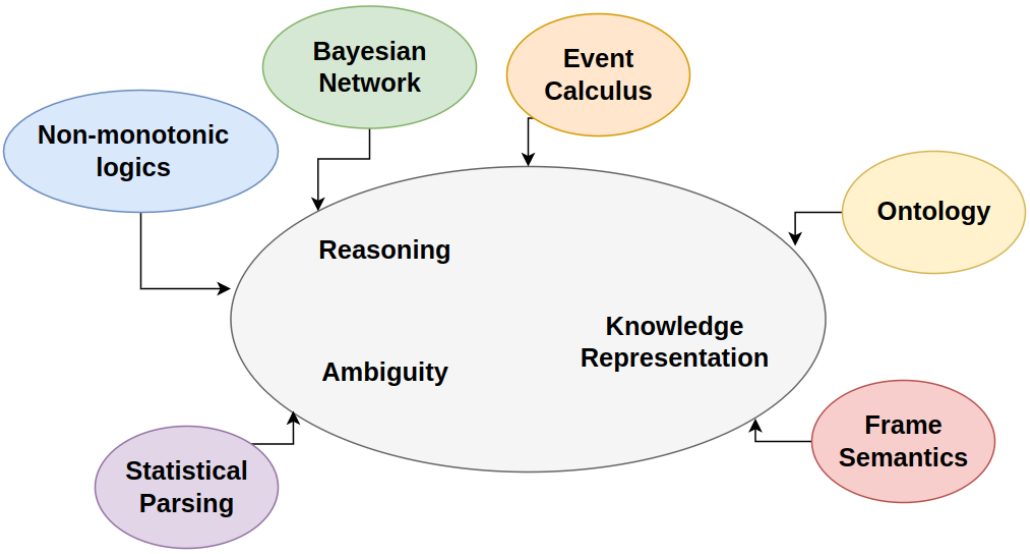


Figure 1.1: The challenge of semantics, as outlined by numerous AI experts in the field of Natural Language Processing during the 1990s and 2000s.

the concept of Universal Grammar (UG). UG is a linguistic theory that posits humans are innately equipped with the capacity to learn the language - it's essentially preprogrammed into our brains. Controversies around this idea aside, a notable aspect of this theory is the assumed independence of syntax and semantics, which enables the existence of sentences that are syntactically correct but semantically meaningless, such as "Colorless green ideas sleep furiously". This seminal work laid the groundwork for ensuing decades of research in Natural Language Processing (NLP) and linguistics, all aiming to tackle the challenging problem of "semantics."

The problem of semantics was seen as so complex that the prevailing view at the time was to break the problem down into separate tasks rather than seeking a singular, all-encompassing solution. Reasoning, ambiguity, and knowledge representation were regarded as some of the most significant challenges in semantics. Researchers employed a blend of symbolic and statistical methodologies to approach these issues. Early efforts tended to concentrate on techniques such as non-monotonic logics, ontology, and frame

semantics, as well as more statistical approaches like statistical parsing and Bayesian networks. However, over time, the emphasis gradually shifted toward more practical tasks within Natural Language Processing (NLP), straying away from the purely theoretical aspects.

Many of these tasks were derived from practical engineering challenges such as summarizing a book or answering questions based on a given text corpus. Nonetheless, it's clear that understanding language can't be simply broken down into a series of tasks with the expectation that solving each one and compiling the results will provide a comprehensive solution. Language understanding is a complex process that involves more than just the successful completion of discrete tasks.

Without delving into the philosophical nuances of language or semantics, from a practical standpoint, the shift in NLP research towards statistical models and away from symbolic methods proved quite fruitful by the 2000s. Researchers found themselves more drawn to concepts that framed semantics in a strictly statistical manner. The guiding ethos of contemporary NLP research can be aptly summarized by a quote from esteemed linguist John Rupert Firth:

"You shall know a word by the company it keeps" (Firth, 1957)

Firth's ideas resonated widely, leading to the establishment of the 'London School' of Linguistics. However, Firth's approach to language modeling eventually gained traction much later within the computer science community. This interdisciplinary interest helped foster new directions in language processing and artificial intelligence.

The application of Transformer architectures in language modeling represents a fresh perspective on the challenge of language representation. In language modeling, the focus is not necessarily on devising immediate solutions for specific tasks. Instead, emphasis is placed on the simple yet deep-seated concept of predicting the next token given a textual prompt. Research over the past few years has revealed that concentrating on solving this seemingly irrelevant problem, which is just text generation, has led to breakthroughs in many traditional tasks and even beyond.

Recent advances in NLP have led to the development of powerful language models such as GPT-X and ChatGPT. These models, trained on massive volumes of textual data, have shown remarkable performance across a variety of classical NLP tasks such as summarization, translation, common sense reasoning, and question answering, among others. The intrigue grows when one considers that the fundamental architecture of transformers for language modeling is relatively straightforward, operating primarily on the core concept of the attention mechanism. This mechanism allows the model to focus on different sections of the input sequence when generating an output. It essentially equips the model with the capability to attend back to the input sequence, instead of forcing it to encode all information into a fixed-length context vector, as was required in earlier sequence-to-sequence models.

More specifically, the Transformer uses a specific type of attention called "scaled dot-product attention", which calculates the similarity between each word in a sentence. This process assigns a score to each word, indicating how much attention should be paid to it. The scores are then used to weigh the contributions of each word to the output. The higher the score, the more "attention" the model pays to the corresponding word.

1.2 Controllable Language Generation

While the recent paradigm shift in language modeling has been highly successful and addresses many traditional tasks, the challenge of employing language models is far from resolved. This dissertation aims to demonstrate that the primary focus of modern NLP extends beyond merely solving specific tasks, even though certain tasks may indeed be resolved in the process. Instead, a deeper exploration into the language models themselves, understanding their mechanisms, and controlling the properties of the generation process is of greater importance. This aspect is particularly crucial considering the pressing safety concerns related to the deployment of Large Language Models (LLMs). The tangible risk of polluting the infosphere with nonsensical, factually incorrect, and hallucinatory textual data underscores the need for careful management and control of these models (Dwivedi et al., 2023). The focus of this dissertation will hence be on the problem of text generation through language modeling, and enhancing the control over the generation process.

A fascinating aspect of these language models lies in the generation process, specifically how the model predicts the next tokens. It's important to remember that these models are statistical in nature; there isn't a single correct output for a given prompt. Quite the contrary, the model can generate a potentially infinite number of text continuations. One simple way to achieve this is by varying the seeds that control the random sampling from the probability distribution that comes from the softmax of the final layer output. Given these characteristics, it's unsurprising that some argue these current language models are essentially "stochastic parrots" (Bender et al., 2021) emphasizing the need for more deep exploration of their capabilities.

Currently, methods for controlling the generation process are rather limited and usually fall under the umbrella of prompt engineering. Prompt engineering refers to the process of designing and fine-tuning prompts to effectively steer a language model toward generating useful responses. In the context of language models, a prompt is the initial input that the model receives and uses as a basis to generate a continuation. Prompt engineering involves devising prompts that maximize the usefulness, relevance, and accuracy of the model's response. Certain strategies can enhance the effectiveness of prompt engineering (White et al., 2023), but the practice itself is viewed more as a collection of hacks and heuristics rather than a truly systematic methodology.

Certain existing techniques, like manipulating the temperature (T) or top_p during the sampling process, can be regarded as controllable language generation methods. However, their application scope is somewhat restricted, indicating a necessity for further development and expansion in this area. In this dissertation, new approaches are proposed in which the general characteristics of texts generated by LMs such as emotion, topic, and reasoning capabilities can be manipulated using systematic methods. The most significant aspect of the approach presented in this dissertation is the emphasis on a more interpretable, mathematical, and systematic method for controlling general properties of language generation, as opposed to ad hoc solutions like prompt engineering. This serves as an initial stride towards scientifically exploring LLMs, controlling their attributes, and customizing them for specific applications.

1.3 Dissertation Outline

This dissertation delves into a comprehensive exploration of different methodologies for controlling the general characteristics of language model generation. Chapter 2 presents an overview of the mathematical underpinnings of language modeling, including a rigorous definition of the issue of controllable language generation. Subsequently, the aim is to categorize each study within this thesis according to the definition of controllable language generation. A thorough discussion will ensue, highlighting the respective benefits and drawbacks of each approach.

Chapter 3 introduces Program-R Version 1, a chatbot that generates responses to user inputs. Although this initial version lacks control parameters, it establishes the groundwork for subsequent iterations in terms of engineering and research advancements.

Chapter 4 focuses on emotion as the control parameter, introducing chatbots: Program-R Version 2, which employs sentiment and facial expression recognition to select appropriate outputs, and EmpTransfo, which incorporates emotions into the training process.

Shifting attention to controlling topical properties of language generation, Chapter 5 introduces a novel approach that modifies the sampling algorithm of the language model to alter its topical attributes.

Chapter 6 extends the scope of generation to abductive reasoning by altering the architecture and training of the language models. The resulting system is capable of generating plausible hypotheses based on incomplete observations, with the control parameter, in this case, being the observations themselves.

Chapter 7 concludes the exploration by providing insights into recent advancements, with a specific focus on ChatGPT. This section will also delve into the limitations of the present study. Additionally, it will highlight potential future research directions and extensions that could build upon the foundations laid in this work.

Chapter 2

Problem Statement and Related Works

2.1 Language Modeling

The problem of "controllable language generation" is tightly related to the problem of language modeling itself. The modern notion of language modeling is based on the probability distribution of a sequence of words. Given a sequence of m tokens $x_1, .., x_m$ as the context, the problem of open-ended language generation can be formulated as finding the continuation $x_{m+1}, ..., x_{m+n}$ with n tokens. In other words, if we consider the whole context plus continuation as follows:

$$x_1, .., x_m, x_{m+1}, .., x_{m+n} \tag{2.1}$$

The language model is the probability distribution over the sequence $x_1, .., x_{m+n}$. So, given any sequence of words the language model assigns a probability to that sequence that shows how likely that sequence can be. For instance, a phrase such as *"pencil ate the the can"* is less probable than *"the pencil ate the paper"*, primarily because the former disregards grammatical norms. This concept can be extrapolated to more nuanced examples where the coherence of the word sequence determines its likelihood.

The language modeling probability, $P(x_{1:m+n})$, can be decomposed using the chain rule as:

$$P(x_{1:m+n}) = \prod_{i=1}^{m+n} P(x_i|x_{<i}) \tag{2.2}$$

The first challenge encountered in language modeling in this context is the issue of sparsity. This simply means that the number of possible combinations for $m + n$ tokens becomes intractable, and we never have sufficient data to model every conceivable combination. To address this challenge, we must integrate induction biases.

Induction bias in machine learning refers to the set of assumptions that a learner uses to predict outputs given inputs that it has not encountered. It is essentially the bias or set of preconditions that a model uses when generalizing from training data to unseen data. In the context of language modeling, an example of induction bias could be the assumptions made by a model about the structure and characteristics of the language it is learning. For instance, the model might assume that sentences usually follow a subject-verb-object order, because that's a common pattern in English. Different models are basically ways in which we help the model to learn those biases, they can take the form of the Markov assumption,

Recurrent Neural Networks (RNNs) (Hochreiter and Schmidhuber, 1997), or Transformers (Vaswani et al., 2017). While my primary focus throughout this work is on Transformers, as they represent the pinnacle of language modeling technology, some of the methods we discuss are more universal and can be adapted to a variety of language models.

2.2 Decoding Strategy

Assuming we possess a probability distribution that accurately assigns probabilities to any given sequence of text, the next logical question is: how do we generate text from this distribution? Various methods exist to sample from a language model's probability distribution, and these are commonly referred to as decoding strategies.

The language modeling probability can be used with a *decoding strategy* to generate the next token for language generation. Finding the optimal continuation can be formulated as:

$$\hat{x}_{m+1:n} = \underset{x_{m+1:n}}{\operatorname{argmax}} P(x_{m+1:n}|x_{1:m}) \tag{2.3}$$

Solving Equation 2.3 is not tractable so practical decoding strategies use approximations to generate the next tokens. The most famous and widely used decoding strategies are greedy decoding and beam search methods. Greedy decoding selects the highest probability token at each time step, while the beam search keeps a set of hypotheses and then updates the tokens in the hypotheses as it goes through and decodes more tokens. Beam search basically uses a dynamic programming approach by branching with a beam size and pruning based on the likelihood to keep a set of most likely continuations. These

approaches are well suited for directed language generation, but they suffer from repetition, genericness, and degenerate continuations (Holtzman et al., 2020). Both of these approaches are deterministic in the sense that they do not involve any random selection in their algorithms.

On the other hand, stochastic decoding methods sample from a model-dependent distribution q ((Welleck et al., 2020)):

$$x_i \sim q(x_i|x_{<i}, p) \tag{2.4}$$

The simplest stochastic sampling consists of sampling from top-k probabilities, the use of constant k is problematic because in some contexts the probability distribution of the next token is flat which means there are plenty of reasonable next tokens to select from but in some other contexts the distribution is concentrated in a small number of tokens. To solve this problem, (Holtzman et al., 2020) proposed Nucleus Sampling. In this method, a subset of vocabulary is defined which is the smallest set $V^{(p)}$ such that:

$$\sum_{x \in V^{(p)}} P(x|x_{<i}) \geq p \tag{2.5}$$

Then the resulting distribution which is based on the new vocabulary should be re-scaled to form a probability distribution. Let $p' = \sum_{x \in V^{(p)}} P(x|x_{<i})$ then we can re-scale the distribution as follows:

$$P'(x|x_{<i}) = \begin{cases} P(x|x_{<i})/p' & \text{if } x \in V^{(p)} \\ 0 & \text{otherwise.} \end{cases} \tag{2.6}$$

Under Nucleus Sampling, the number of plausible next tokens changes dynamically with the context and generated tokens.

Figure 2.1: An illustration of how the language model's generation process can be modified based on discrete input parameters as *positive* or *negative*. This concept closely parallels Style Transfer as used in image processing.

2.3 Controllable Language Generation

Now that we covered both the language modeling and the decoding strategy we can define the Controllabe Language Generation (CLG):

Definition 1. *Controllable Language Generation is the process of generating text from a language model (which could be either statistical or rule-based) with a set of (continuous or discrete) control parameters denoted as C that changes the general behavior of the output text.*

The term "general behavior" refers to the consistent features exhibited in all generated texts when the control parameter is set. This can encompass a range of characteristics including, but not limited to, 'emotiveness', 'topic relevance', 'reasoning ability', 'creativity', 'safety', and the tendency for irrelevant extrapolations, also known as 'hallucination'. When control parameters are continuous, we expect a corresponding progressive alteration in the particular characteristic that these control parameters influence, such as varying levels of 'emotiveness' or 'hallucination'.

Consider a specific instance where we alter a language model's generation based on positive or negative sentiments. This is analogous to Style Transfer in image generation like StyleGAN (Karras et al., 2019), where a base image is transformed according to various artistic styles. Figure 2.1 depicts an example of a neutral sentence that has been regenerated to convey both negative and positive sentiments. It is important to note that while this example demonstrates sentiment transfer using a discrete parameter (*positive*, *negative*), the generation process can begin without a specified prompt, and the designated sentiment can pervade throughout the entire text generation process.

In Natural Language Generation (NLG), there is often a distinction made between open-ended and directed (or constraint) text generation approaches. Directed text generation involves specific applications like machine translation (Bahdanau et al., 2014), data-to-text generation (Wiseman et al., 2017), and summarization (Liu and Lapata, 2019). Conversely, open-ended generation, which encompasses tasks such as story generation and contextual text continuation, garners significant interest (Peng et al., 2018; Radford et al., 2019; Rahman et al., 2023; Wang et al., 2022). It is important to highlight that our definition of CLG encompasses both of these types of text generation. However, our focus lies in exerting control through various parameters (in the case of topical language generation) and even through actual texts themselves (in the case of abductive commonsense reasoning language generation). This approach enables us to shape and manipulate the generated output based on desired criteria, thereby emphasizing the importance of control in both types of text generation tasks.

By viewing LLMs as textual random processes, we can begin to analyze the properties of these processes and explore efficient means of control. The control parameters C, could be integrated anywhere within the model itself, the decoding strategy parameters, or even outside these parts, functioning as a decision-making component. In this dissertation, we explore three distinct strategies for controlling the properties of the text output generated by LMs:

1. Implementing a multi-model approach that combines both machine learning and symbolic methods to select the best response from various random samples.

2. Training a joint model that is conditioned on both the text and control parameters.

3. Modifying the decoding strategy by integrating parameters, which enables operation within our desired probability distribution.

Each methodology carries its own set of advantages and drawbacks, which I will delve into more deeply in the subsequent sections. In this chapter, I am concentrating on the overall structure of each method, while the applications and intricacies will be explored in the upcoming chapters.

2.3.1 Multi-model Approach

The first approach which is also the simplest approach is to change the flow of generation based on discrete conditional values such as emotion outside the model. In order to do this we have to first find the appropriate discrete conditional values for emotions. This can be framed as a kind of classification task on the user prompt, previous conversation, facial expression, or any other data that can be collected in the context of a conversational language model.

If we define the data from the user as d_i, then we use a classifier f to find the appropriate class c:

$$c = f(d_i) \tag{2.7}$$

Note that for f we need to train the classifier using a training set $\{(d_1, c_1), ..., (d_n, c_n)\}$, where $d_i \in D$ is data vectors and $c_i \in C$ is the corresponding labels.

Now, we can define $|C|$ different language models trained specifically for each class. For example one language model can be trained solely on positive sentiment text, one negative and one for neutral. Here c serves as the discrete control parameter and the task of controllable language generation would be simply choosing the right language model after the first classification step:

$$P(x_{1:m+n}) = P_c(x_{1:m+n}) \text{ for } c \in C \tag{2.8}$$

Even though this approach is very simple it can be shown to be effective in various applications like Chatbots in healthcare as we will see in the Chapter 3. Here are the biggest advantages of the model:

1. **Specificity of Response**: By training language models on specific classes, as the control parameters, the generated response can provide a more tailored and context-appropriate response. For example, it might be more empathetic in response to negative sentiment, or more upbeat in response to positive sentiment.

2. **Control over Generated Content**: This method could offer more control over the content that the language model generates because each one of them is trained on a specific class of responses. This could potentially lead to more predictable and safe responses.

3. **Potential for Improved User Experience**: By being able to respond to user inputs in a tonally appropriate way, the language model may provide a more engaging and satisfying user experience. The language model would be more context-aware and could respond in a way that aligns with the user's emotional or mood state.

also, the disadvantages are:

1. **Complexity in Training and Maintenance**: This approach requires training and maintaining multiple language models and a classifier. It requires substantial computational resources and makes the system more complex to manage and update.

2. **Potential for Misclassification**: The classifier's accuracy is crucial. If it incorrectly classifies the class of a user's input, the output might respond inappropriately, which could lead to user dissatisfaction or confusion.

3. **Limitations in Flexibility and Nuance**: By choosing a discrete parameter and partitioning the responses into distinct classes like positive, negative, and neutral, the model might lack the ability to handle nuanced or complex inputs that span multiple sentiments or that don't fit neatly into these categories.

Chapter 3 delves into the application of this technique through the lens of a chatbot, Program-R. This chatbot, grounded in Cognitive Behavioral Therapy (CBT), interacts with individuals grappling with Alzheimer's Disease and Related Dementias (ADRD). However, the limitations previously discussed give rise to the next approach for chatbot design, which fundamentally rethinks the process of controlling language model text generation.

2.3.2 Training Conditional Model

Another approach to solving the problem of controllable language generation is introducing conditional variables into the training itself. If the control variables are C then we can define the conditional language modeling probability distribution as:

$$P(x_{1:m+n}|C) = \prod_{i=1}^{m+n} P(x_i|x_{<i}, C) \tag{2.9}$$

In this approach, the model training relies not just on previous tokens $x_{<i}$, but also on a set of control variables C, which direct the generation of the next token x_i. From a practical standpoint, if we use a Transformer as the language model, these control variables are represented as a series of tokens or embeddings. These are then concatenated to the input token before being processed in the model for training. It's crucial to bear in mind that the loss function in conditional training needs special consideration. While we typically optimize for the best next token, the optimal next control variables must also be taken into account, especially if these variables alter between sentences. This leads to more robust training that considers all control variables.

The merits of this method include its high inference speed, made possible because the model has already been trained on the control variables, eliminating the need for additional computation during inference time. Additionally, it yields high-quality output, as it is trained in tandem with the control variable, making its performance comparable to that of the base language model. However, this approach also presents certain challenges:

1. **Training Dependency**: A notable drawback of this method is its reliance on the control variables introduced during the training phase. Once the model is trained, there's no way to define new control variables in inference time.

2. **Requirement of Labeled Data**: This method necessitates a substantial amount of labeled training data, which is typically costly to acquire. Each text sample, whether a sentence, a paragraph, or a document, needs to be labeled with the control variables.

In chapters 3 and 6, we observe the application of this approach in training language models conditioned on sentiment and contextual observations for reasoning tasks. Due to the high-quality output, conditional training advances the state-of-the-art in both tasks, setting new benchmarks. Given the ample training data and specific applications in these areas, conditional training emerges as the optimal approach, undeterred by the aforementioned limitations.

2.3.3 Modifying the Decoding Strategy

To exert control over the behavior of output generation, we can focus on the decoding strategy, as opposed to modifying the model or its inputs. As outlined in Section 2.2, the decoding strategy represents the final stage of the generation process and essentially constitutes the method by which we sample from the probability distribution of the language model.

Using the probability distribution of the language model $P(x_i|x_{<i})$, we can derive $P(x_i|x_{<i}, C)$ without modifying the model itself by applying the Bayes theorem. In this context, we can consider $P(x_i|x_{<i}, C)$ for posterior, $P(x_i|x_{<i})$ for prior, and $P(C|x_i, x_{<i})$ for likelihood. Therefore, we can formulate the Bayes theorem equation as follows:

$$P(x_i|x_{<i}, C) = \frac{P(x_i|x_{<i})P(C|x_i, x_{<i})}{P(C|x_{<i})} \tag{2.10}$$

While this may initially appear to complicate matters, it actually establishes a relationship between the conditional probability and the base probability of the language model. However, there are two challenges that need to be tackled in this context:

1. **Calculating Conditional Probability**: How can we determine $P(C|x_i, x_{<i})$, i.e., the probability of the control variable given the context?

2. **Addressing Marginalization**: How do we deal with the marginalization (represented by the denominator), which is typically intractable?

In Chapter 5, we address both of these challenges. By introducing certain simplifying assumptions, we're able to considerably speed up the computation while maintaining fluency. This method, which is grounded in Bayes rule rather than arbitrary heuristics, is thus an appealing approach and has the following advantages:

1. **Robustness**: Given that this method is grounded in the Bayes rule, the resulting probabilities are well-formed and do not require any further normalization. This ensures that the influence of the condition variable is truly reflected in the target probability.

2. **Efficient Computation**: This method does not necessitate retraining. The only component that needs pre-computation is the prior of the control variable, which is generally much less computationally expensive to calculate.

3. **Flexibility**: The method allows for modifying the probability for any language model without restrictions on the type of control condition. This provides an unprecedented level of flexibility in controlling the parameter space.

4. **Scalability**: This method allows for adjusting the model to various control variables without needing to retrain the model for each one, thereby providing scalability in terms of adding or modifying control conditions. This makes it highly suitable for larger, evolving projects where new control variables may be introduced over time.

Some of the disadvantages are:

1. **Dependence on the Quality of Priors**: The performance of this method hinges on the quality of the priors for the control variables. If these are inaccurate or poorly estimated, it could adversely affect the outcome of the text generation.

2. **Control over parameter variable**: Another challenge associated with this method is that interpreting the probabilities of each control parameter is not always straightforward.

It is important to note that the approaches proposed in this dissertation are not the only ways to control the general properties of language generation but rather represent a collection of ideas within this domain. However, our objective is to demonstrate how this can be approached from various perspectives, without being reliant on specific model conditions or constraints.

Chapter 3

Program-R: A Rule-based Dialog Generation System

3.1 Introduction

Dialog management systems are systems designed for extended conversations, set up to mimic the unstructured conversational or 'chats' characteristic of human-human interaction, rather than focusing on a particular task like booking flights (Pérez-Soler et al., 2020). Dialog systems can be categorized into two classes: *Rule-based* and *Corpus-based*. *Rule-based* systems follow pattern-matching techniques to find and respond to the users. Even though they are very simple for some applications they are very effective. On the other hand, we have a *Corpus-based* dialog system that can be classified into retrieval-based and generative dialog systems. In retrieval-based systems, one retrieves the most appropriate answer from a corpus of human chats based on the user input and maybe other inputs like

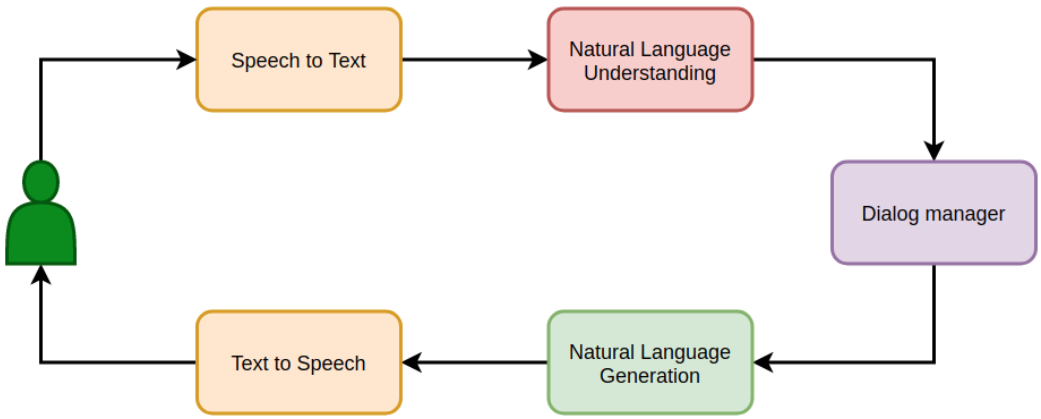


Figure 3.1: The general overview of the architecture of dialog systems.

the history of the conversation. Generative dialog systems also use a corpus of human chats but they model mapping from user input text to outputs. They generalize based on some statistical machine learning model and unlike other methods can generate novel responses based on user inputs.

3.2 Related Works

In this section, we focus on the problem of language generation in the context of chatbots with the control parameter of emotion. Continual refinement of emotion recognition and natural language processing techniques has allowed for chatbots and dialogue systems to be successfully used in therapy and counseling settings. One study attempted to redefine emotion recognition by creating an unobtrusive system to measure emotions with the use of smartphones (Lee et al., 2012). This system eliminated the use of expensive and clunky sensors, as well as demonstrated high accuracy in classifying user emotions into categories. In (Oh et al., 2017), a chatbot was developed that used natural language processing techniques to recognize emotion and respond accordingly, but unlike the current study, the chatbot focused on everyday conversation and had no predefined counseling

schema. Another experiment created a chatbot with emotional capabilities, however for sentence generation, they used general knowledge bases without the specific vernacular needed for counseling (Lee et al., 2017). The closest research to the work described in this dissertation is (Bickmore et al., 2011), which is implemented using an OWL ontology of health behavior concepts. Even though this approach is extensive and increases complexity, it is not flexible to different user inputs and is very hard to extend.

3.2.1 Program-R

To deliver the internet-delivered cognitive behavioral therapy (iCBT) for this study, an AIML-based dialogue system called Program-R has been developed. Program-R is a forked project from Program-Y (pro, 2017) with several modifications to fit the purpose of this study and create a seamless interaction between the robot, Ryan, and the subjects.

Artificial Intelligence Markup Language (AIML) is an XML-based language for writing conversations. Among AIML objects, the following tags are worth citing: “category”, “pattern”, “template”, and “that”. The “category” tag is the basic unit of dialogue that includes other predefined tags. The “pattern” tag defines a possible user input and the “template” tag is a certain response from the dialogue manager. To ensure continuity between the dialogues, the “that” tag is used to connect different dialogues with the last question asked by the dialogue system.

In order to deliver a more interactive user experience, the “robot” tag is introduced. The “robot” tag is the information that is used by Ryan to enhance HRI with the use of multimedia. Inside the “robot” tag there is an “options” tag that lists the possible answers to the questions at the end of the current dialogue response. Moreover, the “image” and “video” tags provide Ryan with specific multimedia information to be presented to the user by Ryan.

When all the dialogues are connected to each other with the "that" tag, a graph structure is created, rather than a tree structure, because the branches of the flow of conversation can merge back again. The control structure of conversations is defined as session frames in a finite-state automaton. The conversation involves a few slot fillings, such as, the name of the user, place of birth, and answers to questions that shape the conversation pattern. All of this user information is saved in a database for use in future sessions.

Unlike most dialogue managers, Program-R is an active system, starting the conversation and asking questions of the user, rather than the user initiating the conversation. Fig 3.3 demonstrates the architecture of the proposed dialogue system. Program-R communicates with Ryan through a RESTful API. The user input from the speech-to-text component is received by the proper Client and then sent to Brain. Brain is the core module that handles several key tasks. It takes care of communicating with storage for cases of resuming the interrupted conversation and connects with the AIML parser to resolve the answers with the consultation of the AIML repository. In Brain, the input will be preprocessed to remove punctuation, normalize the text, and segment the sentences. In the next step, Question Handler adds the context and session data to the input. Context Manager handles the context in which the conversation is happening. For example, two different questions can have yes/no answers, but without knowing the context in which the conversation is happening, responding to them is impossible. In these cases, Context Manager helps to respond properly. In addition, the Context Manager provides custom responses to different users based on the information that was recorded previously for that specific person. Meanwhile, Brain saves all the conversations and session data in the form of explicit (i.e. name or place of birth) and implicit (i.e. mood) user information in the database. Finally, the postprocessing (i.e. formatting numbers, remove redundant spaces and html tags, etc.) will be performed on the answer and the result will be sent to Ryan.

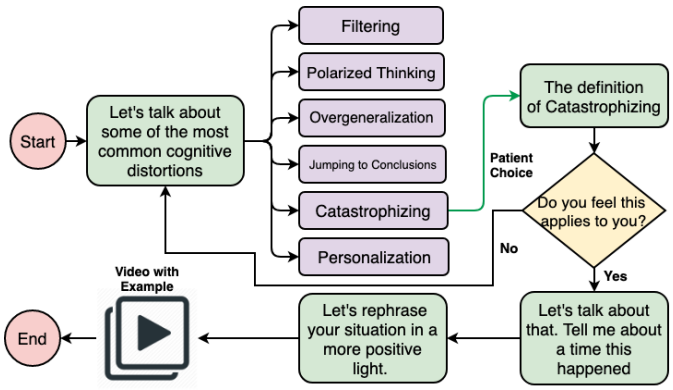


Figure 3.2: A sample dialogue of a conversation about cognitive distortions between Ryan and the user. Users have the chance to choose a distortion that relates to them, discuss it, and learn how to cognitively reappraise the situation with the aid of a visual example.

Due to the fact the dialogue manager works based on an automaton, there is a risk that the user will drift away from the conversation flow by answers that are irrelevant to the current question. In these situations, the dialogue manager can ask the question again with a different format or give control to the WOZ to handle the situation.

Unlike other dialog systems used for counseling and mental healthcare, the proposed approach can interact with patients in more diverse ways with the help of images, music, videos, and the presence of a robot.

3.2.2 Session Dialogues

Therapy sessions were formatted using AIML. The AIML repository contained 7 AIML files that were organized in a session-based manner to follow the structure of iCBT as described in an individual CBT therapy plan for depression (Muñoz et al., 2000). The seven treatment sessions were spread out over the span of four weeks and were broken down into three key points: how thoughts affect mood (sessions 2 and 3), how activities affect mood (sessions 4 and 5), and how people affect mood (sessions 6 and 7). Session 1 consisted of a general introduction and allowed for participants to familiarize themselves

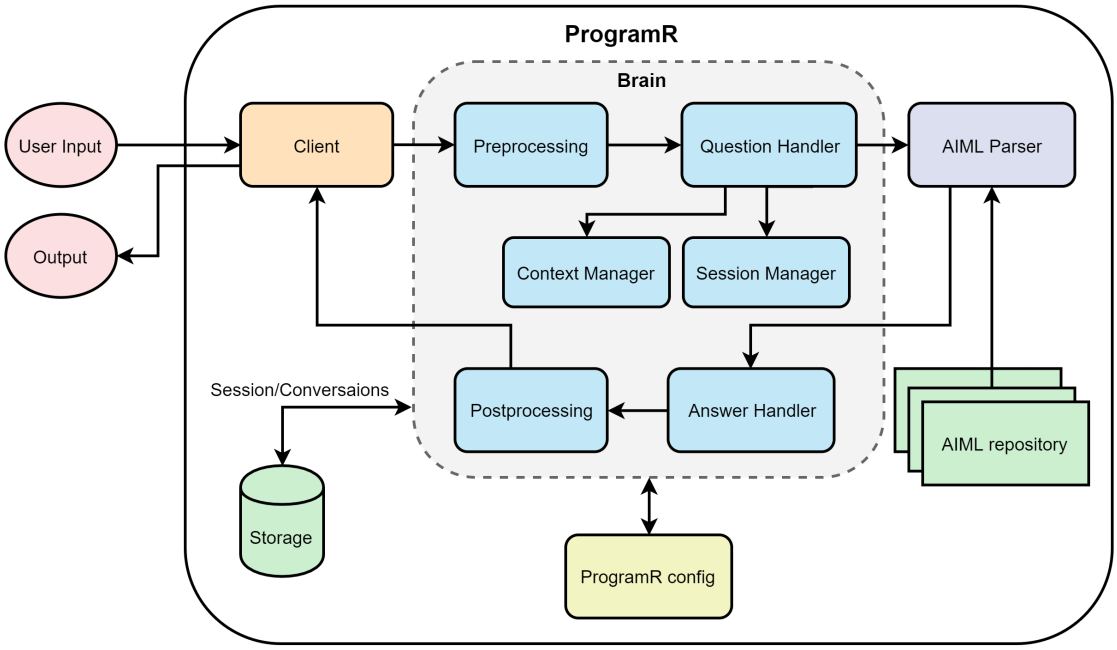


Figure 3.3: The proposed system diagram.

with Ryan. An example of a dialogue between Ryan and a user can be seen in Fig 3.2. In total, there were 165 categories, 23 robot tags, and 27 additional media (10 pictures, 13 videos, and 4 music files). All pictures, videos, and music were educational or therapy-driven in purpose. Therapy sessions were executed by Program-R, the dialogue manager developed in this research.

3.3 Results

3.3.1 Natural Language Analysis

This section primarily focuses on human evaluations. Accordingly, participants have signed the Institutional Review Board (IRB) consent forms, associated with the IRB number 2011-1840. Natural language analysis was used to evaluate the degree of subject involvement throughout the sessions. One measure of involvement is the average response length by the user to questions asked by the robot. To calculate the average response length,

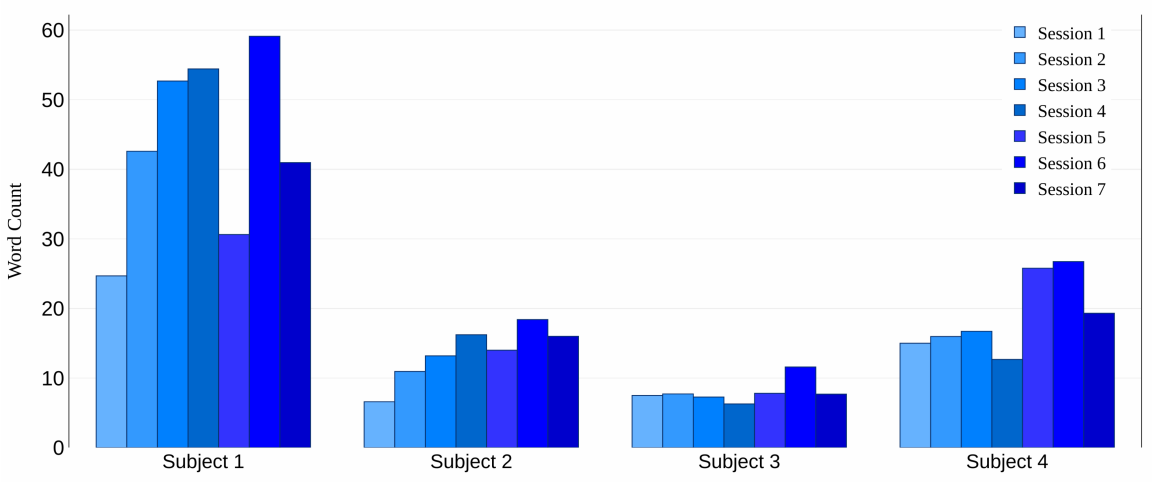


Figure 3.4: Word counts for each subject over the seven sessions.

the user responses were tokenized and the average length of tokens per session was calculated. Fig 3.4 demonstrates the rough increase of word count of each subject as the sessions progressed. Excluding the last session - a closing session involving a wrap-up of the whole study - an increase in average sentence length can be seen in almost all of the participants. In order to decrease the bias on this evaluation, the sessions were designed to have almost the same number of questions. For example, in session 6, which shows the most involvement in all the participants, 18 categories were used whereas the first session has 23 categories. There is also another phenomenon that can be seen in this plot: some participants tended to give longer responses to the questions than others. Despite this, the fact that individual participants talked longer as the sessions progressed still holds.

Another measure of subject involvement is sentiments over time. Sentiment analysis is a technique to evaluate the positive, negative, and neutral sentiments at a sentence level. CoreNLP (Manning et al., 2014) was used to measure the sentiments. First, transcriptions of each sentence spoken by the user were segmented and then the sentiments of each sentence were computed. Stanford CoreNLP has two more categories for sentiments: “very

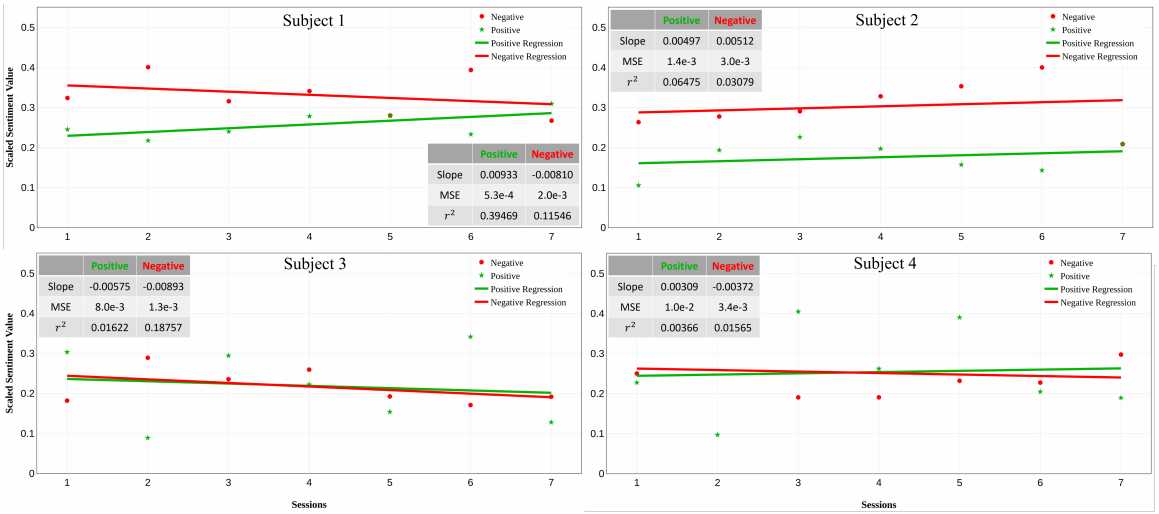


Figure 3.5: Scaled sentiment values (positive and negative) and their linear regression for four subjects and seven sessions. The table in each figure shows the slope, mean squared error, and variance score for positive and negative sentiment regressions.

positive" and "very negative." For the purposes of this research, these categories were considered as "positive" and "negative" respectively, because they occur very infrequently compared to the other three sentiments. The number of positive, negative, and neutral sentiments are scaled such that their overall summation becomes 1.

In Fig 3.5, the neutral contribution in each session was excluded due to the fact that they do not show any mood from the user. The results of the sentiment analysis for each subject were unique. Fig 3.5 shows an increase in positive sentiments and a decrease in negative sentiments for subjects 1 and 4. For subject 2, the positive sentiment increased whereas the negative sentiment increased at the same rate, and for subject 3, the negative sentiment decreased faster than the positive sentiment. The fluctuation in sentiment value for the last two subjects was higher. Overall, the sentiment analysis shows improvement for two subjects and more inconsistent results for the other two.

3.4 Discussion

This research demonstrated promising results about the positive impact of using a Program-R, a chatbot capable of conversing with patients, to administer iCBT to older adults with depression. Natural language analysis demonstrated that as the individual subjects progressed through the sessions, their average

The day-to-day information gathered by the mood scale provides further evidence of subject improvement. Not only did subjects either stay the same or feel better by interacting with the chatbot, but the amount of improvement for many of the subjects also appeared to increase as the sessions continued. This suggests that the subjects felt better on a daily basis and over the course of the entire therapy duration. These results are similar to conclusions in other research discussed regarding a reduction of depressive symptoms following social robot intervention (Chen et al., 2018).

The outcomes of scaled sentiment analysis and word count, as evident, do not provide definitive conclusions. This highlights the limitations of a rule-based approach that fails to consider emotions. In the next version of Program-R (v2.0), I will tackle this issue and incorporate measures to address it.

Additionally, this research establishes the groundwork for an engineering platform dedicated to controlled generative chatbots, which will be further developed in subsequent chapters, with a particular focus on Program-R v.2 in the upcoming chapter.

Chapter 4

Emotional Language Generation

4.1 Introduction

This chapter concentrates on the issue of emotional language generation, particularly in various dialog systems that can integrate emotions in multiple ways. Within this framework, emotion serves as the control parameter that must be considered when generating responses. First, Program-R (v1.0), discussed in the preceding chapter, is expanded by introducing a component capable of interpreting emotions through textual sentiment analysis and facial recognition. This emotional understanding is then used as a control element to shape the output responses. Subsequently, we incorporate a broader range of emotional categories into a transformer-based language model. This enhanced model is capable of comprehending and reacting to user inputs using an end-to-end generative approach.

4.2 Emotion Aware Dialog Generation System

In this section, I only focus on the dialog system that has been used in a socially assistive robot capable of understanding emotions. The calculated emotion has been used as a control property for the dialog system to retrieve the appropriate responses from the AIML repository.

4.2.1 Emotion Recognition

Emotion recognition can be measured at least in two ways: textual sentiment analysis and facial recognition. Automated sentiment analysis is a well-established task in NLP with several open-source publicly available toolboxes such as the CoreNLP (Manning et al., 2014) developed at Stanford University for public use. The CoreNLP sentiment analysis toolbox is based on deep Neural Networks and is trained using the Stanford Sentiment Treebank consigning of 11,855 single sentences extracted from movie reviews (McDuff and Czerwinski, 2018). The system has an accuracy of 85.4% and is suitable for us. The sentiment analysis module returns a value between -1 to +1 as the sentiment value of the preprocessed sentence.

Finally, we fuse perceived emotional facial expressions and sentiment values to make sure the robot understands the multi-faceted user emotions correctly:

$$FinalEmotion = .5 \times SentimentValue + .5 \times EmotionalState \tag{4.1}$$

The $FinalEmotion$ is a weighted average of user utterance sentiment and emotional state that will be used to direct the flow of conversation. The decision to equally average the sentiment and the emotional state is made based on our tests in the laboratory, more experiments are needed to find the perfect balance and weight.

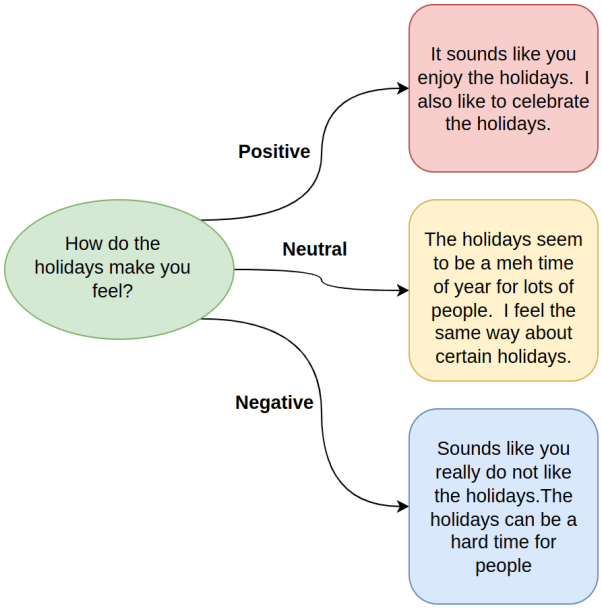


Figure 4.1: An example of the decision-making on Program-R (v2.0) based on the emotions calculated from user response and the responses from the database.

4.2.2 Affective Dialogue Systems

The new version of Program-R is a hybrid (rule-based and machine learning) system that uses state-of-the-art sentiment analysis to deliver an affective dialogue system. Studies on emotion-based dialogue systems stress on different sources of information to extract user sentiments. Approaches like (Burkhardt et al., 2009; Shi and Yu, 2018) use only textual cues for a sentiment-based dialogue system. In (Bertero et al., 2016; Nwe et al., 2003) they explored the use of acoustic features. The proposed system uses multi-modal facial and textual information in a dialogue management system.

Program-R (v2.0) is a sentiment-adaptive AIML-based dialogue system (known as template-based dialogue systems) that can fuse visual and textual information and respond to users accordingly. Unlike most dialogue systems Program-R (v2.0) is an active agent, which means Program-R (v2.0) initiates the conversation and tries to have a controllable chat with the user.

AIML (Ringate, 2001) is an XML-based language that is used for organizing the set of all dialogues in different chatbots like Alice (Wallace, 2009). In AIML-based dialogue systems, we try to find the best response (responses are stored in the Template tag in AIML) for any user input utterance using Regex matching (stored in the Pattern tag). Pattern and Template tags together represent a unit of conversation under the Category tag. One advantage of AIML is that history can be accessed via a "that" tag. Every question is contextualized and is answered based on what was last said between the robot and the user. To deliver a more interactive user experience, more tags and features are added to AIML. The "robot" tags were added to send multimedia information along with the raw text response to give the user a multimedia experience. The "robot" tags contain information such as images and videos and the answers to multi-option questions to be presented to the user in certain dialogues. Moreover, the "getsentiment" tag, a custom tag built for this study, takes the user utterance after preprocessing and sends it to the sentiment analysis module.

Just like the previous version, Program-R (v2.0) communicates with Ryan through a RESTful API. After receiving the output of speech-to-text from Ryan, the raw text will be sent to the Preprocessing module to remove unnecessary punctuation, normalize the text, and sentence segmentation. The Sentiment Analysis module is where the sentiment of the text is mixed with the output of the Facial Expression Recognition module (Emotional State) to get a single score. The Brain's Question Handler takes into account the context, sentiment, and session data while the Context Manager handles the context in which the conversation is happening. For example, some questions may have identical answers (e.g. yes/no), and without knowing the context, producing the proper response is not possible. With the provided information from the Context Manager and the computed value based

on Emotional State and Sentiment, the Question Handler produces an answer. Finally, the selected answer is sent to the Answer Handler and Postprocessing module to be sent back to Ryan. In figure 4.1, a sample from the conversation between the user and the response from Program-R (v2.0) is depicted.

4.3 EmpTransfo

4.3.1 Introduction

Humans have the unique capability to communicate with nuanced emotions through natural languages. Most of the existing conversational dialog systems focus on language understanding and improving the generated responses. Although these features are essential in building dialog systems, they lack empathetic features for conversation, which is essential for quality communication. To increase user' satisfaction, dialog systems need to understand and incorporate emotions to respond with proper emotions. Because of the complexities of human emotions, creating and evaluating empathetic dialog systems is challenging and research in this area is still in the early stages.

It is important to mention that when we refer to an empathetic dialog system, we are describing a type of emotional dialog system that possesses the ability to comprehend emotions and respond appropriately. However, empathy goes beyond mere understanding of emotions; it involves the advanced skill of relating to those emotions and striving to evoke positive emotions in individuals which is beyond the scope of chatbots we are describing here.

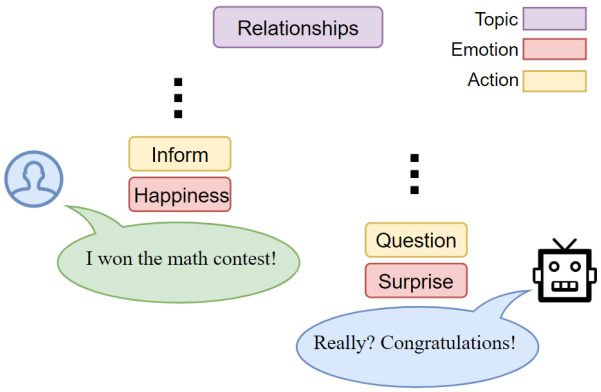


Figure 4.2: An example of the interaction of ***EmpTransfo*** with the user. Contextual information like the history of emotions and actions and also the topic of conversation are crucial to respond with the appropriate emotion.

Recent advances in NLP with the idea of using pre-trained models have led to remarkable results in different NLP tasks. Even though applying the same idea in dialog systems has resulted in improved models in terms of language understanding and generation, taking into account other information like emotions and context knowledge is still challenging. To build empathetic conversational agents, machines need to have the ability to recognize and predict emotions based on the history of conversations and use them in interacting with users.

Corpora used in building most of the traditional dialog system includes general conversations, although these datasets are usually large-scale, they lack specificity and do not contain metadata such as emotions, topics, personality, etc. Training a system based on general corpora leads to conversational agents that do not understand emotions, lack any personality, and tend to produce generic responses such as: "*I don't know.*". For these reasons, there has been an effort to create higher-quality datasets with more contextual information. For example, the DAILYDIALOG (Li et al., 2017) dataset contains information about emotions, topics, and actions.

This thesis presents a novel multi-head Transformer architecture that can use explicit contextual information on emotions, topics, and actions to respond to users' utterances with proper emotions without sacrificing the quality of responses in terms of coherence, relevance, and consistency. Figure 4.2 demonstrates how the interaction between the user and dialog system is conditioned on the history of emotions, actions, and the topic. All these contextual clues make it easier for the dialog system to respond with appropriate emotion. ***EmpTransfo*** is built upon the state-of-the-art dialog system (Wolf et al., 2019b) and introduces a new architectural design that can exploit contextual information. Quantitative analysis shows the model outperforms all the baseline models. The main contributions are:

1. Emotions are incorporated with a multi-task learning approach in dialog systems that is effective and extendable.

2. The proposed approach can be augmented with other contextual information that not only improves the empathetic aspects of responses but also its generation quality.

3. Designing the model in a way that can be used with larger or smaller pre-trained models without changing the architecture of the system. This gives us the flexibility to use ***EmpTransfo*** in different settings based on our needs.

Section 4.3.2 reviews the recent efforts in conversational agents' design to incorporate emotions into language understanding and generation. In the proposed Approach Section 4.3.3, first ***EmpTransfo*** is proposed with more details of preprocessing and input representation. Details of training in the model are given in Training Section 4.3.4 and finally present the results and discussions in the Result Section 4.3.5.

4.3.2 Related Works

Most of the work on using emotions in conversation systems uses Twitter datasets (Sordoni et al., 2015) that contain emojis as the meta-information for the emotions. In (Zhou and Wang, 2017), they use the Twitter dataset and apply a preprocessing method to create a conversational dataset with 64 different emojis that represent different emotions. In the preprocessing step, they used tweets and responses that contain at least one emoji and filtered other emojis based on the occurring frequency. They used a CVAE (Sohn et al., 2015) network to train and generate emotional responses. The choice of using emojis to represent emotion is noisy because it is too fine-grained and in many cases, the combination of different emojis hardly corresponds to any specific emotions.

In (Colombo et al., 2019), they used a seq2seq framework with vector representation for emotions, desired emotion, a regularizer to penalize neutral words and a sampling method that forces the generation of emotionally relevant words.

There are two papers on the NLPCC dataset (Mi et al., 2014), a Chinese language dataset with eight emotion categories. The first one is Emotional Chatting Machine (Zhou et al., 2018) that uses a seq2seq architecture with the embedding of emotions along with words and an internal and external memory mechanism to generate emotional responses. The second work is EmoDS (Song et al., 2019) which uses a seq2seq approach with a training objective based on an emotion classifier that promotes implicit emotion generation. Both works use the lexical attention mechanism in the decoder with more focus on emotional words to inject explicit emotions into responses.

Based on the reports from the above-mentioned works, all the seq2seq-based models tend to generate generic responses that can't capture all the emotions equally well. Furthermore, all previous works use a machine annotating approach in the training process that introduces noise in results.

The most relevant work to ours is (Rashkin et al., 2019). They introduced a new dataset called EMPATHETICDIALOGUES that contains meta-information about conversations. The meta-information is a label that shows emotion and also the situation in which the conversation has happened. They proposed two architectures, one retrieval-based model which looks for the best match using the BERT encoder (Devlin et al., 2018), and a generative model using Transformer architecture.

Using one label for the whole conversation and not each utterance makes it harder for the models to find proper correlations. On the other hand, recent progress has shown promising results on pre-trained language models on conversational models in chit-chat settings. Recently, (Wolf et al., 2019b) showed that using a fine-tuned GPT (Generative Pre-Training), they can beat any other model on the domain of personal chat using the PERSONACHAT dataset (Zhang et al., 2018a).

In this thesis, a new architecture is proposed that can incorporate not only emotion but other relevant meta information in the DAILYDIALOG dataset. The task of how to use multi-task learning for "next emotion prediction" besides language modeling and "next utterance prediction" is discussed. Then how to use the embedding of other relevant information in the dataset to improve the results is expanded.

4.3.3 Proposed Approach

Recent developments have shown substantial improvements in benchmarks on a variety of languages understanding tasks through the use of deep pre-trained language models (Devlin et al., 2018). More specifically, researchers have shown that by fine-tuning a pre-trained model for specific tasks, they can achieve better performance compared to training the model from scratch. This is also crucial when the dataset at hand is small.

Transformer-based models become ubiquitous in NLP with the work of (Vaswani et al., 2017) for multiple tasks including language generation. More specifically, causal language models like GPT and GPT-2 produce remarkable results in the language generation task (Wolf et al., 2019b). In this thesis, GPT pre-trained models have been used to achieve better results.

Empathetic Dialog Generation

Let's assume that in a conversation between two agents, each turn of the conversation by one of the agents is named "utterance". Hence, a conversation consists of a set of utterances. More formally, we have n utterances $U = \{u_1, u_2, ..., u_n\}$ and for any utterance i we have N_i tokens $U_i = \{t_1, t_2, ..., t_{N_i}\}$. Also, for each utterance, there is an emotion corresponding to it, resulting in the sequence of emotions $E = \{e_1, e_2, ..., e_n\}$.

In our dataset, a sample is a sequence of utterances $\{u_1, u_2, ..., u_{T-1}, u_{next}\}$ in which u_{next} can be either the correct next utterance u_T or a distractor from the set of distractors U'_T. A distractor is a random utterance from the dataset. In the same way, if the corresponding sequence of emotions is $\{e_1, e_2, ..., e_{T-1}, e_{next}\}$, then e_{next} is either the correct next emotion e_T or a distractor from the set of distractors E'_T. A distractor emotion is a random emotion other than e_T from the set of all emotions.

The model takes a sequence as input in the embedding space and passes it into a Transformer. The Transformer architecture (Vaswani et al., 2017) consists of a multi-layer Transformer decoder block. Each Transformer decoder block applies a masked multi-headed self-attention operation followed by a feedforward and a normalization layer over the input hidden states and gives the same size hidden states in the output. Then the output of the

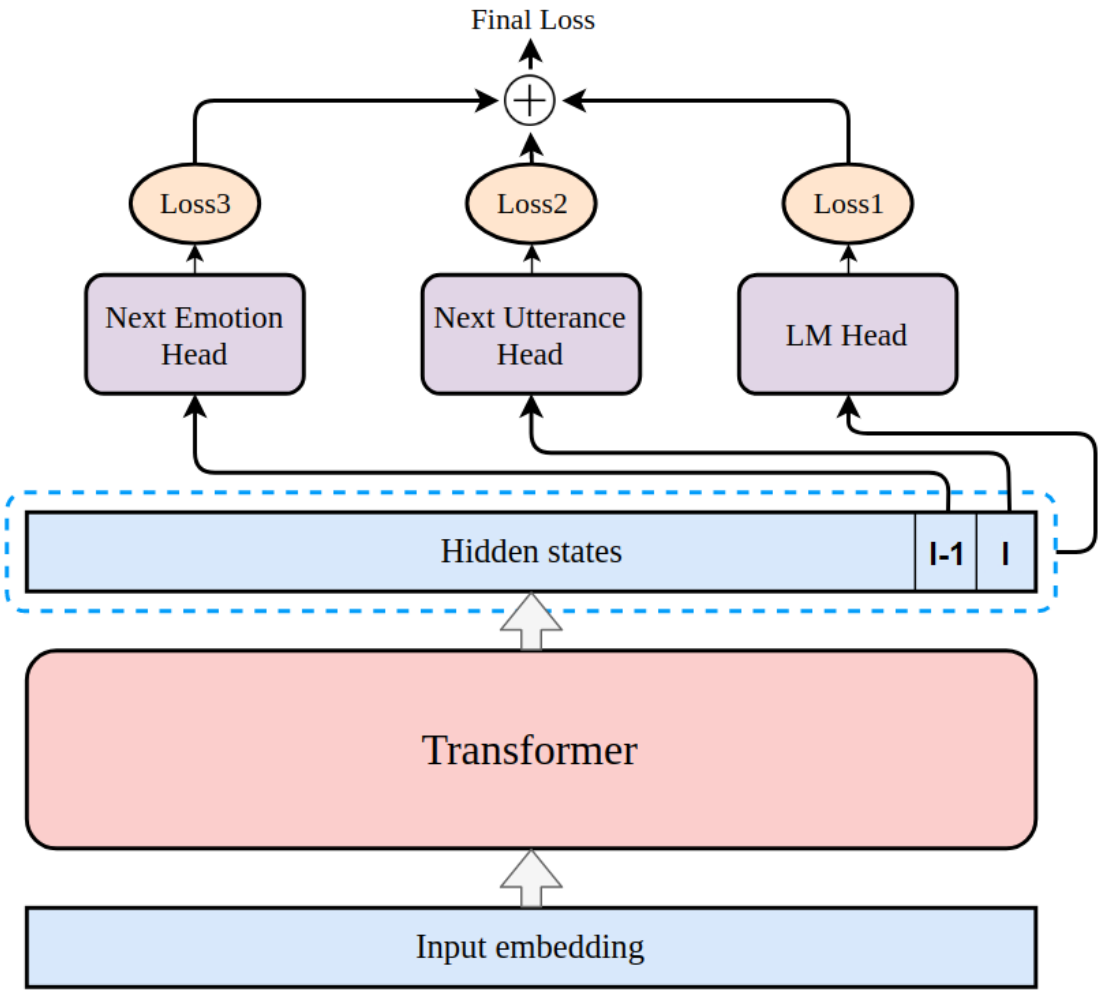


Figure 4.3: ***EmpTransfo***: A multi-head Transformer architecture. There are three feed-forward linear heads on top of the Transformer that map different parts of the last layer's hidden state to desired output sizes to create the loss functions for language modeling, next utterance prediction, and next emotion prediction. The final loss is a weighted sum of all the losses.

Transformer is fed to three feed-forward linear heads, responsible for generating the next emotion, utterance, and token. Here, a 12-layer architecture is used but it can be extended or reduced to other model sizes. In the following, these three different heads and their corresponding loss functions are defined.

1. **Language modeling head:** Language modeling is the task of predicting the next token given a sequence of tokens as the context. If we have a sequence of tokens for the correct next utterance as $U_T = \{t_1, t_2, ..., t_N\}$, then the conditional probability of the next token is:

$$P(t_i|t_1, ...t_{i-1}) = \text{softmax}(\mathbf{h} * \mathrm{W}_1) \tag{4.2}$$

   In which, $\mathbf{h}$ is the last hidden layer of the transformer model and W_1 is the token embedding matrix that is learned in training. Then we can define the loss function based on cross-entropy as:

$$\mathcal{L}_1(U_T) = -\sum_{i=1}^{N} \log P(t_i|t_1, ..., t_{i-1}) \tag{4.3}$$

   where the context of all previous tokens is encoded in a fixed-dimension vector. It should be noted that the language modeling loss is not trained on the set of next utterance distractors U'_T.

2. **Next utterance prediction head:** Following (Devlin et al., 2018), our approach in the "next utterance prediction" is to train the model to predict the next utterance in the conversation. The model learns to distinguish between the correct next utterance among a set of random distractors from other parts of the dataset. More specifically,

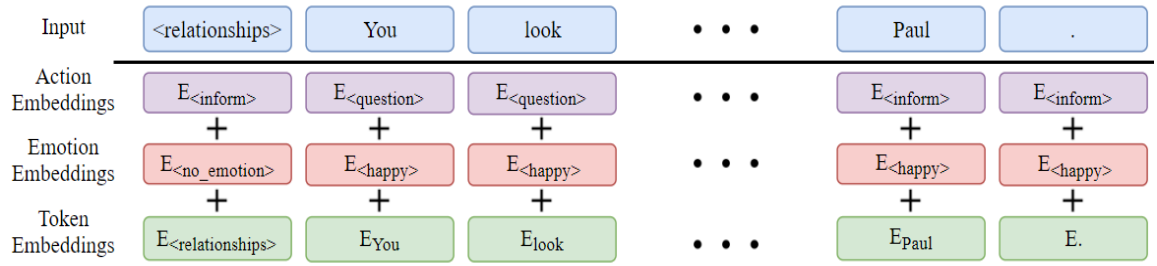


Figure 4.4: The input representation, which comprises three layers of embedding - token embedding, emotion embedding, and action embedding - is illustrated. These layers are summed to form the final representation, which is then fed into the model.

we create a classifier to calculate the probabilities of the next utterance:

$$P_u(a|u_1, u_2, ..u_{T-1}) = \text{softmax}(\mathbf{h_l} * \mathrm{W}_2) \tag{4.4}$$

and a is defined as:

$$a = \begin{cases} 1 & u_{next} = u_T \\ 0 & u_{next} \in U'_T \end{cases} \tag{4.5}$$

$\mathbf{h_l}$ is the hidden state for the last token from the Transformer decoder and W_2 is the weight matrix that is learned for the utterance prediction. Then, the loss function based on cross-entropy is:

$$\mathcal{L}_2(U_{1:T}) = -\log P_u(a|u_1, u_2, ..u_{T-1}) \tag{4.6}$$

3. **Next emotion prediction head**: Similar to the next utterance prediction, the model is trained to distinguish between the correct next emotion among a set of distractors. The reason to add this head is to make the model learn not only the grammatical and language structure but also the appropriate emotions for any given history of

utterances. We can define:

$$P_e(e|e_1, e_2, ..., e_{T-1}) = \text{softmax}(\mathbf{h_{l-1}} * \mathrm{W}_3) \tag{4.7}$$

where e represents:

$$e = \begin{cases} 1 & e_{next} = e_T \\ 0 & e_{next} \in E'_T \end{cases} \tag{4.8}$$

and $\mathbf{h_{l-1}}$ is the hidden state of one to the last token from the Transformer decoder and W_3 is the weights to be learned during the training for the emotion prediction task. The loss function for the next emotion prediction is defined with cross-entropy:

$$\mathcal{L}_3(U_{1:T-1}) = -\log P_e(e|e_1, e_2, ..e_{T-1}) \tag{4.9}$$

Finally, we optimize the following objective which is the total loss function:

$$\mathcal{L}_{total} = c_1\mathcal{L}_1 + c_2\mathcal{L}_2 + c_3\mathcal{L}_3 \tag{4.10}$$

where c_1, c_2, c_3 are hyperparameters that are tuned experimentally. In the experiments, the models with and without the "next emotion prediction" head for comparison has been designed.

Input Representation

In this research, the DAILYDIALOG (Li et al., 2017) dataset is used which is labeled with emotion and action tags per utterance, and with a topic tag for the whole conversation. In Table 4.2, a sample conversation in the preprocessed dataset is shown. A conversation is a sequence of utterances and each utterance can be more than one sentence, but the emotion and action information are defined per utterance. Also distractors to each sample in the dataset is added. In the table, the row highlighted in red with the number d shows a distractor.

All the models use learned positional embeddings with a length of up to 512. Figure 4.4 demonstrates the input representation. The embeddings that have been used are:

1. **Token embedding:** The input sentences are tokenized using byte pair encoding (BPE) with a vocabulary size of 40,478.

2. **Emotion embedding:** Each one of seven emotions is considered as a special token to be learned as a new embedding. Emotion embeddings are copied for each token in the utterances and are added to the input of the network.

3. **Action embedding:** There are four actions for different communication functions that are used in the dialog. The dialog acts are: Inform, Question, Directives, and Commissive. Dialog acts are also embedded with special tokens.

4. **Topics:** There are 10 topics defined in DAILYDIALOG that are specified for each conversation. Topic embeddings are concatenated to the beginning of the first input token embedding.

Table 4.1: A conversation in DailyDialog dataset. The last row that is shown by d is the distractor sample that is drawn randomly from another part of the dataset.

#	Utterance	Emotion	Action
1	You look so happy, any good news?	happiness	question
2	Yes, I've won the math contest	happiness	inform
3	Really? Congratulations!	surprise	question
4	Thank you Paul.	happiness	inform
d	I really want to take him on my knee.	anger	inform

4.3.4 Training

EmpTransfor is based on OpenAI pre-trained model on BookCorpus dataset (Zhu et al., 2015) which covers more than 7,000 books. The books include narratives and dialogues, and emotions in a wide range of interactions between characters. This makes the pre-training suitable for the task of dialogue system training because it consists of sentences in a logical order without shuffling. Furthermore, with a large enough collection, the training is not biased toward any particular domain or application.

Starting with pre-trained weights, the model is fine-tuned on the DAILYDIALOG dataset with the features mentioned in the Input Representation Section 4.3.3. The combination of public evaluation and test as the validation set is used. After preprocessing the training set size is 76,502 and the validation size is 13,809.

The dataset representation is modified to cover different window positions of conversation history. Each sample in the modified dataset consists of the topic, the last two utterances as history context, and the target utterance that can be either the real target or the distractor. The input window then moved forward to cover other parts of the conversation.

The model is fine-tuned with a batch size of 4 for a sequence length of 310 with 20 epochs over the training set of DAILYDIALOG dataset, this is about 1,500,000 steps. For the optimization of the loss function, Adam optimizer is used with a learning rate of 6.25e-5, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ which decays linearly. The gradient accumulation step is set to 8 with a clipping gradient norm of 1. The loss coefficients set equal to one ($c_1 = c_2 = c_3 = 1$). The dropout rates for Transformer were borrowed from OpenAI GPT (Radford et al., 2018). All the proposed models are implemented[1] in Pytorch using Transformers library (Wolf et al., 2019a).

There are different ways of decoding in the language generation part. Here, the nucleus top-p sampling is used (Holtzman et al., 2020). Given logits u of the last hidden layer, and a sequence of $i-1$ tokens, $t_{1:i-1}$ as context, we have the following distribution over the next token t_i :

$$P(t_i = V_l | t_{1:i-1}) = \frac{\exp(u_l/T)}{\sum_{l'} \exp(u_{l'}/T)} \tag{4.11}$$

In which V_l is the l^{th} token in the vocabulary and T is the temperature parameter. Higher values of T result in more stochastic choices over tokens, though lower values of T approach greedy and deterministic choices for the next token. Based on nucleus top-p sampling, we select $V^{(p)} \subset V$ as the smallest set such that

$$\sum_{V^{(p)} \subset V} P(t_i | t_{1:i-1}) \geq p \tag{4.12}$$

And then the distribution of Equation 4.11 should be rescaled to form a probability distribution. According to (Holtzman et al., 2020), $p = 0.9$ and $T = 0.7$ are closer to human text generation statistics and is used for all the experiments.

[1] https://github.com/roholazandie/EmpTransfo

Table 4.2: Summary of the results on DAILYDIALOG evaluation set.

Model	Hit@1 ↑	PPL ↓	F1 ↑	BLEU ↑
Seq2Seq+Attention	9.41	129.3	10.22	5.58
Transformer ranker	17.20	-	**26.37**	**15.79**
OpenAI GPT without emotion	75.01	10.19	18.2	3.755
EmpTransfo	77.25	10.63	19.39	3.99
EmpTransfo + topic	76.87	10.23	18.37	4.51
EmpTransfo + action	77.73	9.17	18.86	3.71
EmpTransfo + action + topic	**78.47**	**9.04**	17.27	2.45

Table 4.3: Examples of model responses.

	Input Prompt		
Model	I finally passed all the exams!	I failed the exam	You scared me!
Seq2Seq+Attention	I'm sorry, but I'm not sure.	I'm going to go to the some time.	I'm going to go to the job
Transformer Ranker	How big was it ?	You're telling me! There are thousands of people here.	Come on! It is really a fun game .
OpenAI GPT w/o emotion	you look much better than before.	let me take your place.	what were you doing?
EmpTransfo+action+topic	that's great! you are really a genius.	Maybe you can try harder next time.	i'm so sorry. i thought you were not coming.

4.3.5 Results

The model is evaluated on its ability to produce coherent and relevant utterances on the evaluation set. Also, the proposed model is evaluated on the task of generating emotional responses given the context on the evaluation set. Evaluation of the dialog systems can be done using automatic metrics and human evaluations. Here, the evaluations are restricted to automatic metrics.

Evaluation on coherence and relevance

Baseline models: For comparison, a seq2seq model is used with "attention mechanism" and a retrieval-based Transformer ranker dialog system as the baseline. The retrieval-based model is similar to (Rashkin et al., 2019), created in ParlAI framework (Miller et al., 2017).

The seq2Seq+Attention model uses linear attention with a hidden size of 128, a learning rate of 0.01, and trained for 20 epochs. Transformer ranker uses a hidden size of 300 that is trained in 40 epochs with a cross-entropy loss function. (all other hyperparameters are the defaults from the ParlAI repository).

Evaluation metrics: Four different metrics is used to evaluate the models:

1. $Hit@1$: this metric is the accuracy of retrieving a gold next utterance among 19 random distractor responses sampled from other dialogues in the dataset.

2. Perplexity (PPL): perplexity is a measure of how well a language model predicts next tokens from the evaluation dataset. More specifically, it is the average per-token log probability over the evaluation set:

$$PPL(p) = e^{\frac{-1}{N}\sum_{w_i} Ln(p_{w_i})} \tag{4.13}$$

3. $BLEU$: It is a metric to measure the distance between machine-generated text with human golden labels (Papineni et al., 2002b).

4. $F1$ token: measures the $F1$ score for token level comparison between the generated text and the golden labels.

In Table 4.2, it is observed that all proposed ***EmpTransfo*** models outperform baseline models in terms of $Hit@1$ and PPL. With more contextual features $Hit@1$ and PPL improve. The proposed model also shows a significant improvement over (Shen et al., 2018) which has a $PPL = 23.8$ over the same dataset. It also outperforms the Transformer retrieval-based model introduced in (Rashkin et al., 2019).

Transformer ranker model has a greater $F1$ and $BLEU$ compared to other models which is expected because those metrics give higher scores for stronger similarity with the golden utterances in the datasets. $BLEU$ is initially developed for machine translation and studies show that it is not a good metric for text generation evaluation (Novikova et al., 2017; Sulem et al., 2018). Table 4.2 also shows that adding more contextual information like topic and action results in higher $Hit@1$ and lower PPL. The reason behind this observation is that more contextual information provides the model with better information on selecting the correct next sentence and the correct next token.

Table 4.3 shows some responses with different given input prompts. The inputs are selected in a way to expect the model to respond with emotions. All the outputs are the first results obtained from the models.

Evaluation on emotion prediction

In order to evaluate the next utterance emotion prediction, we calculate the precision and recall from the confusion matrix over the evaluation dataset. Figure 4.5 demonstrates the calculated confusion matrix with $Precision = 81.35$, $Recall = 72.37$, and $F1 = 76.59$. The proposed model achieves more than 3 percent improvement compared to (Chan and Lui, 2018) who report their best emotion prediction results with $precision = 70.81$, $recall = 76.16$, and $F1 = 73.39$.

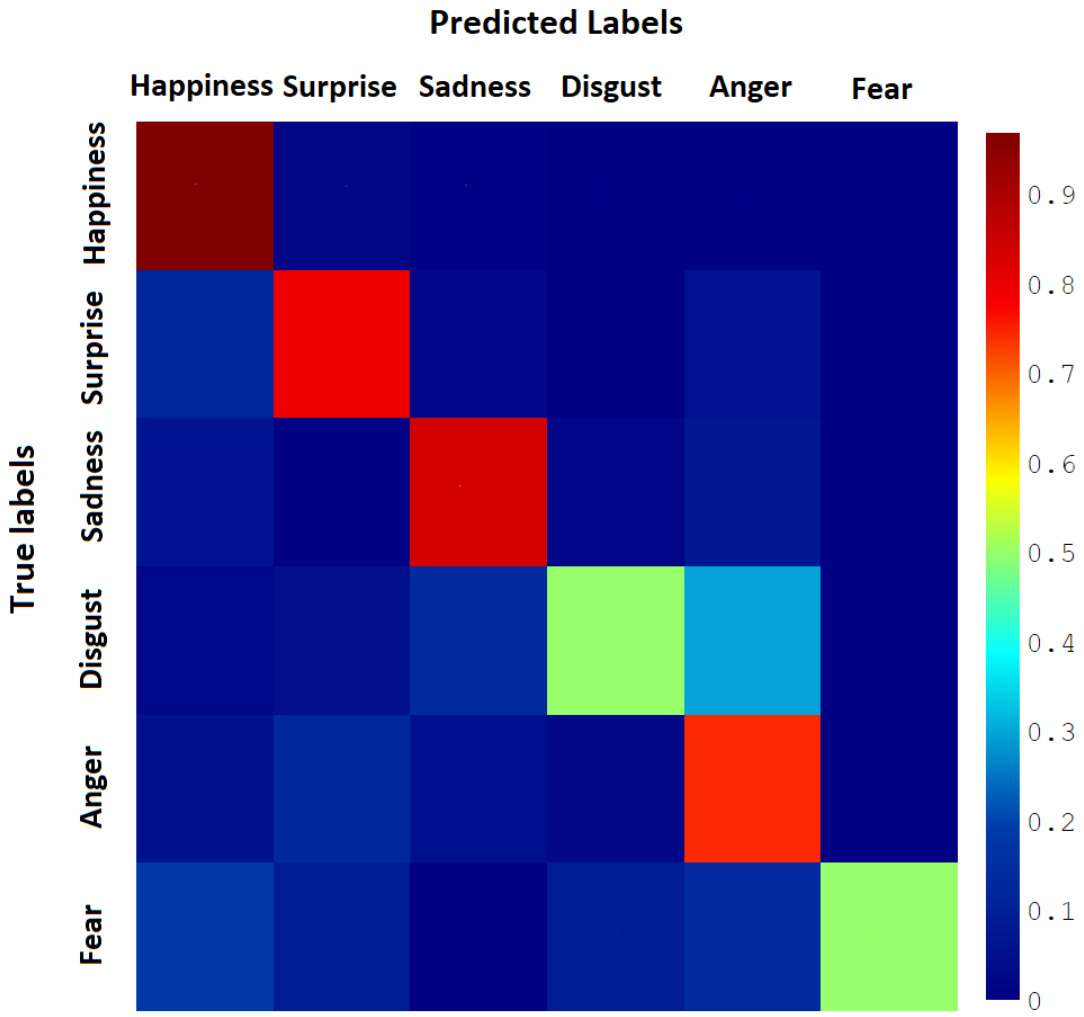


Figure 4.5: The confusion matrix of emotion prediction for DAILYDIALOG with 6 emotions using ***EmpTransfo*** and all the features (best in color).

4.3.6 Discussion

In this chapter ***EmpTransfo*** a multi-head Transformer model is introduced which is an empathetic aware dialog system to interact with users with higher quality in terms of coherence, relevance, and emotion. The novel approach of introducing emotion head to incorporate emotions into the conversation is in line with the challenging task of improving language models. The proposed method is built upon the language model of OpenAI-GPT for language generation. One of the limitations of the proposed approach is requiring meta-information on emotion, action, and topic in order to respond with the proper emotions.

Chapter 5

Topical Language Generation

5.1 Introduction

The advent of transformer language models (Vaswani et al., 2017) has greatly improved the performance of NLP tasks such as text generation, which is an essential component in many downstream applications. The use of transformer models like GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020) to generate and continue text primed with an arbitrary input prompt has led to coherent and realistic texts. More strongly grounded applications such as translation, image captioning, and summarization that have input/outputs are less problematic in text generation with current decoding algorithms (Li et al., 2020). However, in open-ended tasks (e.g., dialog generation or language modeling (LM)), failures such as repetitive text, unnatural topic switching, and contradictions are often observed (Holtzman et al., 2020). Exploring new ways to confront these weaknesses is an active area of research in NLP. This chapter addresses the issue of topical language generation, a critical component in generating long, coherent, and realistic texts. The results can also be used to improve the downstream open-ended text generation tasks such as dialog generation or predictive response suggestion (Kannan et al., 2016).

Despite the fact that pre-trained LMs store a vast amount of knowledge about the world (Petroni et al., 2019), their real potential has not been harnessed yet for controllable text generation. This means that even though there is currently a lot of knowledge about the world in our pre-trained LMs, we are still unable to control the topical attributes of generated texts. Controlling the text generation to incorporate knowledge about specific topics usually requires major changes to the LM architecture, loss function, or retraining the whole models with annotated data.

As we saw in introduction 1, language modeling calculates the probability distribution $P(x)$ of a sequence of tokens x for a given input text. On the other hand, topical language generation which is a type of controllable language generation can be formulated as modeling $P(x|t)$ where t is a specific topic. Here, t is the control parameter C. Different approaches for this problem have been proposed that usually involve expensive retraining or creating annotated datasets on restricted controllable attributes. This chapter addresses the following research question:

"How to combine the knowledge from topic models such as Latent Dirichlet Allocation (LDA) and Latent Semantic Indexing (LSI) to control the language generation by pre-trained causal language models (e.g., GPT-X (Brown et al., 2020), LLaMA (Touvron et al., 2023), PaLM (Chowdhery et al., 2022)) and steer them toward specific topics?"

The proposed approach does not require any retraining or fine-tuning of the original model, but at the same time, it is flexible and fast. In the proposed approach, pre-trained topic models (such as LDA or LSI) can be used or trained to enable the final topical language generation in a fully unsupervised manner. All the codes, models, and data that have been used here are publicly released .[1]

[1] https://github.com/roholazandie/topical_language_generation

The problem with most existing models is that they do not take into account the distributional properties of words for a topic. Statistical topic modeling techniques, especially LDA and LSI, have proven to be successful for data mining and uncovering hidden semantic structures from a given corpus of text. The main motivation to use topic models is that they analyze and return the distributional properties of the words based on the themes that run through them. In other words, just like language models that are trained on large text datasets, topic models find the topical properties of words after training on a relatively large text corpus. Topic models are more accurate and robust compared to word lists that have been used extensively in the literature. Moreover, the extraction of word-topic distributions makes the integration with language model vocabularies easier. Also, topic modeling does not require any prior annotation or labeling of the data. This makes it a particularly good candidate for controllable text generation with more flexibility and less cost. It is worth mentioning that the topic coherence of the generated text by the proposed model has increased due to the fact that, unlike other methods that focus on words or hidden variables to represent topics, every word in the vocabulary gets the correct attention from the topic model.

The main contributions are:

- Introduction of topical language generation (***TLG***) which consists of generating text conditioned on a specific chosen topic.

- Demonstrating that the base ***TLG*** model follows the Bayesian conditional principle and introducing parameters that control the strength at which the model generates on-topic tokens.

- Generation of text with the same topical distribution of the given document. In other words, it will be shown that the topical properties of a given document can be replicated in the generated text.

- Comparing ***TLG*** with existing state-of-the-art language generation models and showing that the proposed model outperforms them on coherency, fluency, token diversity, and speed measurements without any extra costly training.

The rest is outlined as follows: first, the related works on conditional language modeling are reviewed, then in the proposed approach I go over a background on the current state-of-the-art language modeling techniques and then introduce the proposed approach on topical-aware language modeling. I prove the proposed equation mathematically using the Bayes rule. Then I review the two most used topic modeling techniques: LDA (Latent Dirichlet Allocation) and LSI (Latent Semantic Indexing) and show how I use them in conjunction with the proposed method to generate topical texts. In Section 5.5 I demonstrate two different approaches to regulate the ***TLG***. I also propose simulating document topic generation as another application of ***TLG***. Experiments consist of demonstrating different aspects of ***TLG***. I show the results of the method with different topics, comparing it with SOTA models, showing how the parameters change the output, the results of document topic simulation, and fine-tuning. Finally, in the discussion section, I demonstrate how the entropy and KL-divergence in the ***TLG*** are different compared to the base LM. Different variants of ***TLG*** and their differences will also be studied.

5.2 Related Works

In this section, the methods for controllable text generation in NLP are described. The methods are categorized into Generative Adversarial Network (GAN), Variational Autoencoder (VAE) methods, conditional training, and decoding algorithms.

Controlling image features such as style and color using encoder-decoder, GAN, and VAE architectures is the main motivation for researchers in NLP to apply the same rules to text inputs. Researchers mostly use Recurrent Neural Networks (RNN) and adjust the encoder-decoder structure to control the features of the generated text, in the same manner that has been used with images. Unfortunately, because text data is discrete and therefore not differentiable, these methods have not been less successful with text inputs.

The progress in language modeling with new Transformer models shifts the research to keep the architecture intact while trying to change either the dataset, decoding algorithm, or in some cases tweaking small parts of the architecture to achieve better results in controllable language generation. The main reason for this change was the astonishing capabilities of ever-larger Transformer models to write long and coherent texts. Changing the dataset usually leads to conditional training methods, and changing the decoding helps to generate more realistic and diverse texts from the language model. Another motivation for using Transformers is that because the language modeling task has been defined as a conditional probability on previous tokens, we can still use the same rule without introducing new hidden variables that are hard to incorporate and interpret.

5.2.1 Rule-based Methods

In rules-based methods usually, there are templates that follow a grammar with placeholders that need to be filled with appropriate values. In controllable text generation, the policy at which we use these grammar-based methods will change. (Varges and Mellish, 2010) employed an approach that consists of generating and ranking steps. There are two components in their system, the first component is a "generator" component that works based on a grammar that creates a list of candidates. The second component ranks the outputs of the generator component based on the cosine similarity with the training text

corpus. (Ferreira and Paraboni, 2017) have implemented a rule-based model to generate natural language descriptions conditioned on speaker information. Their approach creates a description using Referring Expression Generation (REG) patterns. These patterns are meant to produce the linguistic forms that uniquely refer to a given target object. The major problem with rule-based methods is that they do not really generate text but create them based on deterministic algorithms on the grammars of the language, so they are not creative, can only be used for specific purposes, and usually are very fragile in new applications.

5.2.2 GAN and VAE Methods

GAN models are unsupervised machine learning methods that have been successfully used in image processing to generate new data with the same statistics as the training set. Controllable text generation involves some feature extraction that manipulates the GAN process. Unlike image processing, the text data in NLP tasks are discrete. Unfortunately, applying GANs for discrete outputs is problematic because the generator should be differentiable (Goodfellow, 2016). There are a few ways to tackle this problem like using reinforcement learning or changing the generator to produce continuous outputs. In (Yu et al., 2017), they bypassed the problem by directly performing the gradient policy update. They used the reinforcement learning reward that comes from the GAN discriminator. In LeakGAN (Guo et al., 2018), the problem of long text generation has been solved by leaking the reward signal on high-level extracted features and then passing the signal to the generative network. The generator incorporates such informative signals into all generation steps through an additional MANAGER module, which takes the extracted features of the current generated words and outputs a latent vector to guide the WORKER module for the next-word generation.

Other methods involve using continuous hidden variables that control the desired features like style, topic, sentiment, and content. In (Bowman et al., 2015b; Hu et al., 2017; Malandrakis et al., 2019b), VAE (Variational Autoencoder) has been used to train hidden variables that can control some aspects of the generated text. For example, in (Dethlefs and Cuayáhuitl, 2015) the authors address the problem of language generation in situated tasks that involve human instructions. The model is trained from human-human corpus data and learns particularly to balance the trade-off between efficiency and detail in giving instruction. Both methods are much more flexible compared to traditional rule-based methods because they model the language as probability distributions and this brings more diversity in token generation. But, there are two main problems with GAN and VAE methods: first, training GAN and VAE with text inputs is very unstable which means it is difficult to find the right parameters for the model to converge. Likewise, even with the right parameters, the quality of the generated text is not very good. Second, it is hard to interpret the hidden variables that control the generated text. In some cases, one can find some interpretable features but in general, we do not have a good intuition about how the hidden variables control the features of the generated text. By contrast, in the proposed approach we do not rely on hidden variables, the training is stable and most of the topics that are extracted from the topic modeling are interpretable.

5.2.3 Conditional Training

In conditional language models, the desired feature is explicitly present in the dataset or training process. In the conditional transformer language model (CTRL) (Keskar et al., 2019), the authors used control codes along with the input text that governs the style, content, and task-specific behaviors. They trained their 1.63 billion-parameter transformer model on 140 GB of text. Other methods of controllable text generation address the prob-

lems of repetition, overuse of frequent words, and logical consistency of the generated text. In (Li et al., 2020; Welleck et al., 2020), an unlikelihood training has been introduced that pushes down the probability of undesired tokens that cause the aforementioned problems. In (Stahlberg et al., 2018), a method that involves the fusion of language modeling and translation models was employed to improve the fluency of the generated texts. Unlike GAN and VAE methods conditional training is stable and has higher quality. However, the issue with conditional training methods is that we have to create datasets that contain the features we want. Even though the quality of the generated texts in conditional training is high, it takes a lot of effort to create a dataset and train it from scratch leading to large models that are restricted to the codes in the dataset. Fine-tuning large language models also suffers from the same limitation, even though fine-tuning in general is relatively less costly than training from scratch. In comparison, the proposed method does not change the base language model, and training is limited to the training of the topic modeling, which is cheap.

5.2.4 Style Transfer

Another active area of controllable text generation is style transfer. Language style transfer concerns the problem of migrating the content of a source sentence to a target style by separating the semantic content of what is said from the stylistic dimension of how it is said (Prabhumoye et al., 2018). Text styles can be defined in terms of sentiment (positive/negative/neutral), formality (formal/informal), tense (past/present/future), or any other style (Li et al., 2018).

In (Mueller et al., 2017), a recurrent variational autoencoder is proposed that maps the input sentences to a continuous latent space which then revises the latent representation of the sentence in a certain direction guided by a classifier. In this approach, the decoder imitates the favored sentiment. Due to the lack of parallel datasets, some researchers (Xu et al., 2018) have tried to neutralize the input sentence using either neural networks or retrieval-based methods to create a neutral version of the input and then add the style to it. State-of-the-art models (Zhao et al., 2018) use GAN to disentangle the content from the style and then transfer the latent representation of the content to the desired style with a decoder. The use of GAN in style transfer was introduced in (Hu et al., 2017) and has been extensively used in other works (Fu et al., 2018; Singh and Palod, 2018; Zhang et al., 2018b). Style transfer methods are restricted to a predefined set of control codes, and due to this restriction, their applications are specific. Unlike these methods that rely on predefined control codes, the propsed approach can get all sorts of topic information from another source and incorporate them into a general framework.

5.2.5 Structured Data to Text Generation

In some applications, the controllable language generation needs to follow the features of a structured text. Data-to-text generation can be described as a controllable language generation task in which the model has access to structured data and is asked to generate natural text that conveys that information. For example in WebNLG task data/text pairs from DBPedia are presented to the model. In (Kale, 2020), the T5 model has been used and the task is defined as a text-to-text from the structured input to the textual output. The model has achieved 57.1 BLEU and 0.44 METEOR. For this purpose, they flattened the structured input from predicate triples to text and then fed it to T5. Unlike other approaches that go from structure data to text in one shot using a neural network, (Moryossef et al.,

2019) follow a different approach. They split the process into two steps: first, a symbolic text-planing stage that is faithful to the text, and then a neural generation stage to realize the outputs from the previous stage. In other words, the feature extraction has been done symbolically and then the neural generation stage uses those features. They achieved 47.4 for BLEU.

In (Logan IV et al., 2019) the KGLM has been introduced. The purpose was to generate factual language based on knowledge bases. The approach is equipped with methods to select and copy facts from a knowledge graph. By following this pattern they can easily incorporate factual knowledge about the world whenever it is needed. When the model wants to generate the next token it first decides the type of mentions that are related to the token. If it can draw a relation in the knowledge graph it will change the final distribution of words that need to be generated. For example, if we have *super mario land is the publisher of nintendo* when we have "super mario" and "publisher" as generated text the distribution over vocabulary will change in a way that the token "nintendo" has a higher chance to be selected.

Another interesting work shows how to use the idea of few-shot learning to generalize well using only limited data. More specifically, in (Peng et al., 2020) the model uses the task-oriented dialog system data to generate textual continuation conditioned on the task. The idea of pretraining of large language models along with fine-tuning a corpus of annotated data has been used. They first pre-process the dialog acts as a sequence of control codes and then append them to the actual text from the dataset. They achieved the best results in comparison to LSTM and the base GPT model. In the same line as the previous research on few-shot learning in (Wang et al., 2020b), an approach that consists of using the BERT model to understand the novel intents has been used. It aims to discriminate a joint label space of both already mentioned intents and novel intents. Conditional Text

Generation with BERT (CG-BERT) employs a large pre-trained language model to generate text conditioned on the intent labels. By modeling the utterance distribution with variational inference, it can generate diverse utterances for the novel intents using only a few utterances available.

5.2.6 Decoding Algorithms

These methods modify the basic decoding algorithms without changing the language model architecture or the training process to control the generated text by the model. Early attempts for decoding like greedy decoding or beam search result in degenerate texts with repetition, meaning that high-frequency tokens appear too often and low-frequency tokens are drawn rarely. In (Holtzman et al., 2020), the Nucleus Sampling was introduced that allows for diversity in the token distribution of the generated text. The resulting text better demonstrates the quality of the human text, yielding enhanced diversity without sacrificing fluency and coherence.

In (Ghazvininejad et al., 2017; Holtzman et al., 2018), a weighted decoding algorithm based on the fusion of a log probability with a set of manual scoring functions has been used. The score functions feature some positive aspects of the generated text like reducing repetition, increasing text diversity, topical words, and sentiment. While the proposed approach has some similarities with this method, in this chapter Hand-crafted functions are not used for extracting features. Instead, reliance is placed on trained features from another successful model.

As demonstrated in (See et al., 2019), manual weighted decoding is more specific but it comes with sacrificing the fluency. Recent research by (Malandrakis et al., 2019a) combines the state-of-the-art pre-trained language models with one or more simple attribute classifiers that guide text generation without any further training of the language model.

Their pre-defined bag-of-words (BoW) for topical language generation limits the use of new topics and also does not consider the exact influence of all the words in the vocabulary for topics. In their method, they change the gradient of the last layer and push the base language model gradient to generate desired tokens more often. This approach is a "plug and play" method which means there is no need to retrain the language model but it is still very slow.

The most relevant work to ours is (Baheti et al., 2018), in this work, the constraint of topics in source and target texts are applied by adding a term to the log probability that helps to capture the similarity of the topics in the source and target texts. They pointed out that the topics extracted from LDA do not work well in practice, because it gives equal probability to topics and syntax words. Instead in their approach, they use the HMM-LDA probability that corresponds to the topic component of the model. Also in (Xing et al., 2017) and (Lau et al., 2017), LDA has been used for training and the LDA weights were applied in the attention mechanism.

Also, in (Dziri et al., 2018) the concept of topic modeling, particularly LDA, has been used in the attention mechanism of the encoder. They used the same probability distribution of topic in the decoder and added it as an extra term to the basic word probability. The encoder-decoder model is the seq2seq model that has been used in this work. The quality of RNN-GRU in text generation and the problems with parallelizing the training make them deprecated for modern NLP tasks. Like these methods, LDA is also used as the source of our topic information, though LSI is considered, which opens the door for other topic models that fit the formulation as well. However, unlike these methods, the proposed model does not need to incorporate them during training. It is demonstrated that the influence of the topics can be controlled without sacrificing fluency by selecting the appropriate parameters.

5.3 Proposed Approach

5.3.1 Language Modeling and Decoding

The applications of language generation in NLP can be divided into two main categories: directed language generation and open-ended language generation. Directed language generation involves transforming input to output such as machine translation, summarization, etc. These approaches need some semantic alignment between the inputs and the outputs. On the other hand, open-ended language generation has much more freedom in the generation process because it does not need to be aligned with any output. The open-ended language generation has applications in conditional story generation, dialog systems, and predictive response generation. Even though there is more flexibility in choosing the next tokens compared to directed language generation, controlling the top-level features of the generated text is a desirable property that needs to be addressed and still is a challenging problem.

5.3.2 Topical Language Modeling

Given a list of K topics $t = \{1...K\}$, to control the outputs of the language model to follow a certain topic, at each generation step, we have to model the following probability distribution:

$$P(x_{1:m+n}|t_j) = \prod_{i=1}^{m+n} P(x_i|x_{<i}, t_j) \tag{5.1}$$

To create the right-hand side of Equation 5.1, we change the last layer of the network that creates the logits.

Here, we adopt the GPT transformer architecture which is the decoder part of the original Transformer model (Vaswani et al., 2017). Because of its auto-regressive property, at any given time step i, the embeddings of input tokens $x_1, .., x_i$ can be stacked into a matrix $\mathbf{X}_{1..i}$ and then fed into the network as follows to give the probability of the next token given all of the previous tokens:

$$\mathbf{h}_0 = \mathbf{X}_{1..i}\mathbf{W}_e + \mathbf{W}_p \tag{5.2}$$

$$\mathbf{h}_l = \text{TranformerBlock}(\mathbf{h}_{l-1}) \quad l \in [1, n] \tag{5.3}$$

$$S(x_i|x_{<i}) = \mathbf{h}_n\mathbf{W}_e^T \tag{5.4}$$

$$P(x_i|x_{<i}) = \frac{\exp(S(x_i|x_{<i}))}{\sum_z \exp(S(z|x_{<i}))} \tag{5.5}$$

where $\mathbf{W}_e$ is the token embedding matrix and $\mathbf{W}_p$ is the positional embedding matrix. Here, we have n layers. The first layer is fed with $\mathbf{h_0}$ and the final layer outputs $\mathbf{h_n}$. The logit S is obtained by passing $\mathbf{h_n}$ through a feed-forward linear layer. In the original implementation, the logit S has been used for the final probability distribution over the vocabulary. The TransformerBlock takes a hidden state and outputs another hidden state with the same shape. More specifically, the TransformerBlock consists of the following functions that go from $\mathbf{h}_l$ to $\mathbf{h}_{l+1}$:

$$\overline{\mathbf{h}}_l = \text{LayerNorm}(\mathbf{h}_l) \tag{5.6}$$

$$\mathbf{H}_l = \text{MultiHead}(\overline{\mathbf{h}}_l) + \mathbf{h}_l \tag{5.7}$$

$$\overline{\mathbf{H}}_l = \text{LayerNorm}(\mathbf{H}_l) \tag{5.8}$$

$$\mathbf{h}_{l+1} = \text{FeedForward}(\overline{\mathbf{H}}_l) + \mathbf{H}_l \tag{5.9}$$

LayerNorm and FeedForward are the usual layers used in modern neural networks. MultiHead function is the multi-head attention module that is defined as follows:

$$\text{MultiHead}(\mathbf{X}) = Concat(\mathbf{z}_1, ..., \mathbf{z}_k)\mathbf{W}_0 \tag{5.10}$$

$$\mathbf{z}_i = \text{Attention}(\mathbf{X}\mathbf{W}_i^Q, \mathbf{X}\mathbf{W}_i^K, \mathbf{X}\mathbf{W}_i^V) \tag{5.11}$$

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{D}^{-1}\mathbf{A}\mathbf{V}, \quad \mathbf{A} = tril(exp(\mathbf{Q}\mathbf{K}^T/\sqrt{d})), \quad \mathbf{D} = diag(\mathbf{A}\mathbf{1}_L) \tag{5.12}$$

where $Concat(.)$ is the concatenation function, $tril(.)$ returns the lower-triangular part of the argument matrix including the diagonal, and, the $diag(.)$ is a diagonal matrix with the input vector as the diagonal. Also, $\mathbf{1}_L$ is the all-ones vector of length L and $exp(.)$ is applied element-wise. All the matrices $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, $\mathbf{W}_i^V$ and $\mathbf{W}_0$ are trainable parameters.

We can use the Bayes rule on $P(x_i|x_{<i}, t_j)$ to obtain:

$$P(x_i|x_{<i}, t_j) = \frac{P(x_i|x_{<i})P(t_j|x_i, x_{<i})}{\sum_z P(z|x_{<i})P(t_j|z, x_{<i})} \tag{5.13}$$

Because in topic modeling, documents are treated as bag of words we can also assume that the probability of the topic for each token is independent of the previously generated tokens. Based on this assumption we have:

$$P(t_j|x_i, x_{<i}) = P(t_j|x_i) \tag{5.14}$$

Now, assuming that we have $P(t_j|x_i)$, then using Equation 5.5 we can prove that the conditional topical language model can be written as:

$$P(x_i|x_{<i}, t_j) = \frac{\exp(S(x_i|x_{<i}) + \log P(t_j|x_i))}{\sum_z \exp(S(z|x_{<i}) + \log P(t_j|z))} \tag{5.15}$$

Proof

Proof. Starting with Equation 5.13 with topic independence assumption of Equation 5.14, we can write:

$$P(x_i|x_{<i}, t_j) = \frac{P(x_i|x_{<i})P(t_j|x_i)}{\sum_y P(y|x_{<i})P(t_j|y)} \tag{5.16}$$

Now, using the Equation 5.5, we can rewrite the Equation 5.16 to:

$$P(x_i|x_{<i}, t_j) = \frac{\frac{\exp(S(x_i|x_{<i}))}{\sum_z \exp(S(z|x_{<i}))}P(t_j|x_i)}{\sum_y \frac{\exp(S(y|x_{<i})}{\sum_z \exp(S(z|x_{<i}))}P(t_j|y))} \tag{5.17}$$

which can be simplified to the following:

$$P(x_i|x_{<i}, t_j) = \frac{\exp(S(x_i|x_{<i}))P(t_j|x_i)}{\sum_y \exp(S(y|x_{<i}))P(t_j|y)} \tag{5.18}$$

and finally if we take $P(t_j|x_i)$ and $P(t_j|y)$ into the exponential function, it gives us Equation 5.15.

□

The question of how to obtain $P(t_j|x_i)$ still remains. In the following section, the process of extracting topical probabilities from the topic modeling techniques is illustrated.

5.3.3 Topic Modeling

Topic modeling algorithms automatically extract topics from a collection of textual data. They are based on statistical unsupervised models that discover the themes running through documents. Two main algorithms are used in topic modeling.

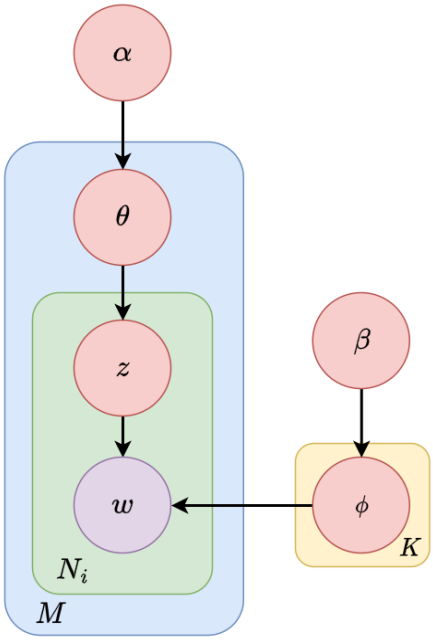


Figure 5.1: Plate notation of LDA model. ϕ_k is RV (random variable) of topics with the β controlling its shape, θ is the RV over documents that is controlled by α. z is the RV showing the topic index, and finally w is the words that is generated by this process.

1- **LDA (Latent Dirichlet Allocation)**: The basic idea behind LDA is that in a collection of documents, every document has multiple topics and each topic has a probability distribution. Moreover, each topic has a distribution over vocabulary. For example, a document can be on the topics of "Football", "News" and "America" and the topic of "Football" can contain words including "NFL", "Football", "teams" with a higher probability compared to other words.

Given a collection of M documents with vocabulary V, we can fix the number of topics to be K. The LDA can be thought of as a generative process in which each token is generated through Algorithm 1.

Algorithm 1 LDA Generative Process

$\boldsymbol{\theta}_d \sim Dir(\alpha)$

$\boldsymbol{\phi}_k \sim Dir(\beta)$

foreach $d \in \{1..M\}$ and $w \in \{1..N_d\}$ **do**

$z_{d,w} \sim Cat(\boldsymbol{\theta}_d)$

$x_{d,w} \sim Cat(\boldsymbol{\phi}_{z_{d,w}})$

end

In Algorithm 1, $\phi_k \in \Delta^{|V|-1}$ is a simplex that specifies the probability distribution of topic k. $\theta_d \in \Delta^{K-1}$ is another simplex that determines the probability distribution of document d over K topics. First, we draw samples from Dirichlet distribution with parameter α for θ_d and samples from Dirichlet distribution with parameter β for ϕ_k. Both parameters α and β are hyperparameters that need to be fixed. Then for each document and token index, we first sample topic index $z_{d,w}$ from the categorical distribution with parameter θ_d. Then we sample token $x_{d,w}$ from the categorical distribution with parameter $\phi_{z_{d,w}}$. Figure 5.1 depicts the plate notation of the LDA as a graphical model.

The probabilities of topics per document and topic for tokens can be summarized in matrix forms, $\theta_{M\times K}$ and $\phi_{K\times|V|}$, respectively. These parameters should be learned through Gibbs sampling or variational inference methods (Blei et al., 2003). After the learning, we have the distributions of topics for each token and hence we can write:

$$P(t_j|x_i) = \phi(j,i) \tag{5.19}$$

We incorporate $P(t_j|x_i)$ in our proposed topical language model.

2- **LSI (Latent Semantic Indexing)**: LSI is the application of the singular value decomposition method (Deerwester et al., 1990) to a word-document matrix, with rows and columns representing the words and documents, respectively. Here we use tokens instead of words to have consistency with the language models. Let $\mathbf{X}_{|V|\times M}$ be the token-document matrix such that $X_{i,j}$ is the occurrence of token i in document j, then singular value decomposition can be used to find the low-rank approximation:

$$\hat{\mathbf{X}}_{|V|\times M} = \mathbf{U}_{|V|\times M}\mathbf{\Sigma}_{M\times M}\mathbf{V}^T_{M\times M} \tag{5.20}$$

After the decomposition, $\mathbf{U}$ still has the same number of rows as tokens but has fewer columns that represent latent space that is usually interpreted as "topics". So, normalizing $\mathbf{U}$ gives us the scores of each token per topic. We can use this score for the probability of topic j for each token i in the vocabulary:

$$P(t_j|x_i) = \frac{\mathbf{U}^T[j,:]}{\|\mathbf{U}^T[j,:]\|}[i] \tag{5.21}$$

In Equation (5.21), $\mathbf{U}^T[j,:]$ means j^{th} row of the matrix $\mathbf{U}$.

5.4 Controllable Generation Methods

The conditional topical language model in Equation (5.15) gives us a token generation that is conditioned on a specific topic but we cannot control the amount of the influence. Besides, using the prior distribution on topics leads to a sub-optimal token generation that hurts the fluency (Baheti et al., 2018). In this section, the concern is addressed by modifying the base Equation (5.15) to enhance the text generation process.

1. **Adding topical parameter and logit threshold**: adding the term $\log(P(t_j|x_i))$ directly to the actual logit from the model can deteriorate the fluency of generated text in some cases. Two methods to alleviate this problem are proposed. We introduce a new parameter γ to control the influence of topical distribution:

$$P(x_i|x_{<i}, t_j) = \text{softmax}(S(x_i|x_{<i}) + \gamma \log(P(t_j|x_i))) \tag{5.22}$$

Higher values of γ result in more on-topic text generation because the final probability will be dominated more by $\log(P(t_j|x_i))$ than the logit from the base language modeling.

The other approach is to cut the log probabilities of the topic with a threshold. The lower values of S correspond to tokens that the model gives very low probabilities and we do not want to change them because it introduces unwanted tokens and diminishes the fluency. In Equation 5.23, we only keep $\log(P(t_j|x_i))$ for all the values of S that are larger than $threshold$.

$$\text{logprob}(i) = \begin{cases} \log\left(P\left(t_j \mid x_i\right)\right) & \text{S}\left(x_i \mid x_{<i}\right) > threshold \\ 0 & \text{otherwise} \end{cases} \tag{5.23}$$

and $logprob$ used in the following equation:

$$P(x_i|x_{<i}, t_j) = \text{softmax}(S(x_i|x_{<i}) + \gamma\, \text{logprob}(i)) \tag{5.24}$$

lower values of threshold correlate with more on-topic text generation because we change more tokens from the original model by $\log(P(t_j|x_i))$.

2. **Using α-entmax instead of softmax**: The problem with the softmax function is that it gives non-zero probabilities to a lot of unnecessary and implausible tokens. The softmax function is dense because it is proportional to exp function and can never give exactly zero probabilities at the output. We use α-entmax instead to create more sparse probabilities that are less prone to degenerate text. α-entmax is defined as (Correia et al., 2019):

$$\alpha\text{-entmax}(\mathbf{z}) := \underset{p \in \Delta^{|V|-1}}{\text{argmax}} \{\mathbf{p}^T \mathbf{z} + H_\alpha^T(\mathbf{p})\} \tag{5.25}$$

where $\Delta^{|V|-1} := \{p \in \mathbb{R}^{|V|-1}, \sum_i p_i = 1\}$ is the probability simplex, and for $\alpha \geq 1$, $H_\alpha^T(\mathbf{p})$ is the Tsallis entropy which defines the family of entropies (Tsallis, 1988) as follows:

$$H_\alpha^T(\mathbf{p}) = \begin{cases} \frac{1}{\alpha(\alpha-1)} \sum_j (p_j - p_j^\alpha) & \alpha \neq 1 \\ -\sum_j p_j \text{log} p_j & \alpha = 1 \end{cases} \tag{5.26}$$

α-entmax is the generalized form of the softmax function. In particular, for $\alpha = 1$ it exactly reduces to the softmax function, and as α increases, the sparsity in the output probabilities continuously increases. Here we are specifically interested in $\alpha = 2$ which results in sparsemax (Martins and Astudillo, 2016):

$$\text{sparsemax}(\mathbf{z}) = \underset{\mathbf{p} \in \Delta^{|V|-1}}{\text{argmin}} ||\mathbf{p} - \mathbf{z}||^2 \tag{5.27}$$

Unlike the softmax function, sparsemax can assign zero probabilities.

3. **Adding temperature and repetition penalty parameters**: We need to make some changes to the base nucleus sampling to control the base distribution flatness and prevent it from generating repetitive words. We denote the final logit after the above changes as u_i. Given a temperature T, repetition penalty r and the list of generated tokens g, the final probability distribution for sampling is:

$$P(x_i|x_{<i}, t_j) = \text{softmax}(u_i/(T.R_g(x_i))) \tag{5.28}$$

where $R_g(x_i)$ is defined as below:

$$R_g(x_i) = \begin{cases} r & x_i \in g \\ 1 & x_i \notin g \end{cases} \tag{5.29}$$

In Equation (5.28), when $T \to 0$, the sampling reduces to greedy sampling; while if $T \to \infty$ the distribution becomes flatter and more random. The penalized sampling discourages drawing already generated tokens. $r = 1.2$ set, following (Keskar et al., 2019), which results in a good balance between fluency and lack of repetition.

5.5 Simulating Document Topic Generation

An obvious observation from topic models is that many real documents are on more than one topic. For example, a paper about bioinformatics can be on the topics of genetics, neuroscience, and computer science. Given the trained parameters, we can modify the generative process of LDA, which is based on BoW assumption to create multi-topic document generation using a specific input document. In other words, we can simulate the topical behavior of an input document using the proposed topical language model.

Algorithm 2 Simulating Document Topic Generation

$generated\_sequence = prompt$

foreach $w \in \{1..N_d\}$ **do**

$logits = \text{language\_ model}(generated\_sequence)$

$z_w \sim Cat(\boldsymbol{\theta}_d)$

$\boldsymbol{p} = \text{softmax}(logit + \gamma \boldsymbol{\phi}_{z_w})$

$next\_token \sim Cat(\boldsymbol{p})$

$generated\_sequence = \text{concat}(generated\_sequence, next\_token)$

end

Algorithm (2) redirects the generation process towards topics in the given document as follows: the algorithm starts with a prompt text and given an input document it iterates N_d times, in which N_d is the length of the input document. In each iteration, it calculates the $logits$ from the base language model and also draws a topic from the topic distribution of the given document $\boldsymbol{\theta}_d$. Then the probability distribution of tokens for the selected topic and $logits$ from the base language model will be combined. This gives us the final probability of tokens that we can draw the next tokens from. Finally, we concatenate the chosen next token to the input to feed it back to the base language model.

5.6 Experiments

Several experiments to evaluate the controllable natural language generation in different aspects are conducted. In Section 5.6.1, the ability of the model in generating coherent outputs for different topics is shown. In Section 5.6.2, the proposed model is compared with state-of-the-art language generation models and show that the proposed model outperforms them on different aspects of fluency and controllability.

5.6.1 Topical Text Generation with Different Topics

One of the biggest benefits of ***TLG*** is that it can be used with different language models without any retraining or fine-tuning of the base model, however, to generate topical texts we need to have topics extracted from a text corpus. For training the topic models, the Alexa Topical-chat dataset (Gopalakrishnan et al., 2019) is used. This data set contains conversations and a knowledge base in a wide variety of topics from politics and music to sports. The tags for topics in the dataset is not used but they are extracted automatically with our LDA and LSI topic models. This unsupervised approach gives us the flexibility to work with any raw text corpus.

In preprocessing, first the dataset is tokenized using Byte Pair Encoding (BPE) tokenizer (Gage, 1994). Then, very rare and very frequent tokens are filtered out because they affect the topic modeling negatively. Very frequent words are usually stop words and rare words are usually not very informative. The initial number of topics is empirically set to 8 and the batch size to 200 documents. The final results are not very sensitive to these parameters and as long as the topics are intuitively separable the ***TLG*** works fine. The LDA model is trained by Online Variational Bayes (VB) technique which is based on online stochastic optimization with a natural gradient step (Hoffman et al., 2010). The training continues by feeding new documents until the topics converge, or until the number of iterations which is set to 600 is reached.

The results from parameter search based on topic coherence score are adopted which will be discussed in Section 5.6.4. The prior probability on documents is set to be a symmetric Dirichlet distribution with $\alpha = 0.1$. The prior probability of word distribution can be learned from data using maximum likelihood estimation (Huang, 2005).

For training the LSI model, the same dataset as LDA is used. Stochastic singular value decomposition on a sparse input (Halko et al., 2011) is used to extract the matrix **U** in equation 5.21. In this experiment, a fixed neutral prompt has been used to make sure the model is not conditioned on the few initial tokens. The results in Table 5.1 show that after selecting a topic from the topic modeling output, the model can create long, coherent, and fluent text continuation without manually injecting extra knowledge from other resources or through training on labeled datasets. The LSI with softmax function is illustrated in Table 5.1, the LSI with softmax function has been used. To avoid cherry-picking, the first output with a fixed seed from ***TLG*** has been selected.

Table 5.1: TLG with a fixed prompt (The issue is that) that is conditioned on different topics

Football	*The issue is that* some football players are not allowed to play in rugby league."I think it's a shame because we've got so many young people who have been playing for us and they're all going out there fighting, but I don't know if you can imagine what the impact will be on them." He said each player would need professional training before being able go back into regular contact with any of his teammates behind teams having no official team or club affiliation between their clubs Football League!

Politics	*The issue is that* while presidential leaders are often able to get things done,they can't do them without the support of their own party members." In fact state legislatures have been reluctant to enact any kind for decades due largely government-initiated "party politics,"which has led some states such as New York into a political crisis over how presidential candidates should approach issues like abortion and gay rights according in part those concerns about what it means given current trends between parties may be more important!
Media	*The issue is that* there will not be television coverage of news events in Russia."We are going to have a lot less media," said three people familiar with what set up telecommunication services for the event. last month—TV medium companies like Vyldo, and Tmall Television & Radio transmitting their own content through its network, which has been shut down since May after being accused earlier of violating Russian law about broadcasting political messages on radio transmission platforms.
Physics	*The issue is that* when information about such radio waves are transmitted, the frequency fields of electromagnetic radiation can be measured."We have a lot technology to do,"Aquo said. He added energy-threshold measurements could help scientists understand current conditions and how these signals interact with electrical circuits through sound modulating devices using amplitude oscillations and phase alternating conductor (A&P),"which allows pulse width modulation." When this happens back during space transmission medium properties formating an interference pattern.

Transportation	*The issue is that* car parts are not cheap because they can be bought at a much lower price than the original. I have seen many people who buy cars from Ford and say when you get them, almost all were sold for $100 US or less after having to pay more in taxes rather than buying new ones with higher prices(and I'm sure there would always depend on where vehicle was purchased).This has been true of most other vehicles since it started being used as an alternative fuel source!

5.6.2 Comparison of Text Generation with Other Models

To evaluate and compare ***TLG*** with other methods topic coherence and n-gram diversity metrics are used. Topic coherence (Röder et al., 2015) can be measured in many different ways. More formally, if the model generates the set of words $W = \{w_1, w_2, ..., w_n\}$. Each word should be compared to any other word in the generated sequence. The set S is defined as:

$$S = \{(w', w^*)|w' = w_i; w_i \in W; w^* = W\} \tag{5.30}$$

the coherence is the mean value of cosine similarity between the word2vec (Mikolov et al., 2013) vectors of all the pairs in S:

$$C = \frac{1}{|S|} \sum_{(w',w^*)\in S} \frac{\mathbf{V}_{w'}.\mathbf{V}_{w*}}{||\mathbf{V}_{w'}||||\mathbf{V}_{w*}||} \tag{5.31}$$

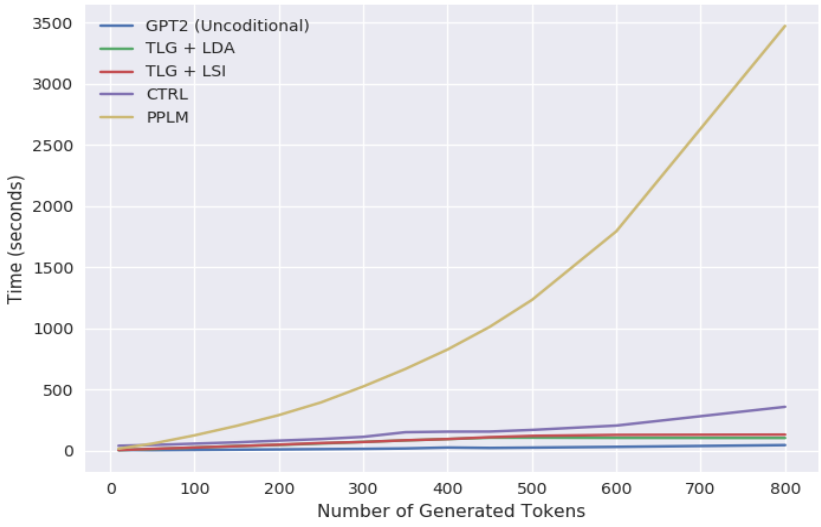


Figure 5.2: As the number of generated tokens by the model increases, the time needed to decode them also increases. ***TLG*** methods are the fastest and PPLM is the slowest controllable language generation.

But the topic coherence alone is not enough because if the model degenerates and produces repetitive tokens then the topic coherence will increase. We also have to make sure that the model creates diverse n-grams. Dist-1, Dist-2, and Dist-3 as the number of unique 1, 2, and 3-grams are reported across all the samples generated for a given topic with different prompts. This score is a very good indicator of the diversity of samples generated by the models.

Table 5.2: Comparing TLG with other models on topic coherency and Dist-1,2,3 which is an indicator of token diversity. All the experiments have less than 1e-5 variance.

Model	**Topic Coherence** ↑	**Dist-1** ↑	**Dist-2** ↑	**Dist-3** ↑
GPT2	0.48292	93.088	99.478	99.909
CTRL	0.51319	69.888	95.737	98.560
PPLM	0.47280	**96.160**	99.419	99.666
TLG+LSI	0.56064	91.993	**99.681**	**99.991**
TLG+LDA	**0.61485**	88.457	99.270	99.983

Table 5.2 shows that ***TLG*** models have the superior capability in generating both coherent texts while keeping the token diversity very high.

One of the main restrictions on most approaches is the limitation on the number of topics/conditions that the model can work on. Even though the proposed model is free from those restrictions, we have to limit the comparisons on the predefined topics of other approaches. Here, the methods on the topic of "politics" are reported which is available in all the models.

In Table A.1, the result of the comparison of the proposed model to the baseline CTRL (Keskar et al., 2019) and PPLM (Malandrakis et al., 2019a) has been demonstrated. CTRL is a conditional language model with specific control codes. Although it outputs high-quality text continuation, it does not have the flexibility for any other topic outside its predefined codes and one has to retrain the network on a dataset with a new control code that represents that topic. This model is also very large and contains 1.63 billion parameters which makes it hard to be used in real applications. The text generated by CTRL also contains meta-information such as "title, text, Score, etc" that was trained from the original labeled text and it diminishes the quality of the generated text. PPLM is a plug-and-play language model which means it does not need retraining of the base model, however because of its nature of perturbing the gradient of the base model, the text generation process of PPLM is extremely slow. Figure 5.2 shows that PPLM is slower compared to other models. PPLM also suffers from the text degeneration problem by repeating itself on some tokens. In PPLM, topics are represented as predefined BoW which gives more flexibility compared to CTRL but still leaves the creation of new topics a difficult task for users. Also, as a model for topics BoW is overly simple because it gives the same weight to all the tokens in the bag. It ignores all other tokens that are not in the BoW and leaves the topic extraction as a manual task for annotators.

On the other hand, ***TLG*** does not need any retraining and it works with base models of different sizes. ***TLG*** gets the topics from a topic modeling algorithm once, then it can be used or shared with others just like the base language model itself. In Section A.1, also the results of ***TLG*** with different topic models are compared that are combined with both softmax and sparsemax functions. Both functions result in quality outputs. In Section 5.7, the differences between them are discussed in more detail.

5.6.3 Document Topic Simulation

One of the novel features of the proposed approach is the ability to generate text not only with one specific topic but also generate documents with the same topical distribution as the given document. In Table 5.3, two samples from the document topic simulation have been shown. The left column shows real samples without any modification from the Alexa Topical dataset. After processing and extracting the topic distribution of each document on the left column the Algorithm (2) is employed to generate similar documents with respect to topical distribution that has been shown on the right column. One interesting observation is that the generated documents do not have one topic anymore. For example, the original document on the top left has a distribution over topics of *Music* and *America*, the same pattern can be observed in the generated text on the top right column. The down left document is around the topics of *Politics* and *Communication* which is replicated on down right generated text from the algorithm. It should be noted that the samples from the dataset are those samples that were considered to have the mentioned topics even though they may not contain the exact topic title words. For example, the sample from the dataset on top

Table 5.3: Samples of document topic simulation. Original documents come from Alexa Topical dataset and corresponding simulated documents follow the same topical behaviors of the given document.

original document	simulated document
Classical music is art music produced or rooted inthe traditions of Western culture , including both liturgical (religious) and secular music. While a more precise term is also used to refer to the period from 1750 to 1820 (the Classical period), this article is about the broad span of time from before the 6th century AD to the present day, which includes the Classical period and various other periods. The central norms of this tradition became codified between 1550 and 1900, which is known as the common - practice period. The major time divisions of Western art music are as follows	The United States/Sates may be the world leader in music, dance & film production company and has been known since its inception over 50 years ago through World musical festivals such as: Party A series 6 Earth called American House In Uth team were held at all of our studios across every city on earth from New York City almost 100 different times during this time period (1962), Los Angeles was one only 2nd place for most first television show ever produced by American studio with more than 1 million viewers worldwide including many major!
A Russian national who claimed ties to the Kremlin told President Trump's personal attorney, Michael Cohen, as early as November 2015 that he could use his Russian government connections to help Trump's business and political prospects. The new Russia contact was revealed Friday by special counsel Robert S. Mueller III, as he outlined cooperation that Cohen has provided the investigation into Russian interference in the 2016 election.	This is a very good example of public policy that has been successful in the past. "The government should be able to make sure it's doing what needs being done, and not just on social media sites like Facebook where people are posting their views, " former Labor leader Bill Shorten told Newstalk 4B TV last month – but he said there needed new tools for dealing with online abuse as well under various laws such at any time too early? President Donald Trump recently signed an executive order!

left does not have the word America but is still considered to be from the topic *America*. In general, the quality of document simulation, which has multi-modal topical distribution is lower than the base ***TLG*** model. This is probably due to the more complex relationship between topics in the original text that Algorithm (2) captures using only random sampling.

To show how much two documents are similar sentence-bert (Reimers and Gurevych, 2019a) is used which is based on the BERT model to create embeddings for sentences that can be used for similarity purposes. To evaluate the document topic simulation the similarity between 1000 samples from the Alexa dataset as the given document and the output of the document topic simulation for each one of those documents is calculated. In the next experiment, the same samples from the Alexa dataset are calculated but with outputs from GPT-2. In both experiments, the same prompt has been used. The first experiment gives an average of 44.625% cosine similarity compared to the second experiment which is 11.015% similarity. This shows that the document topic simulation creates very similar texts to the original retrieved document from the dataset.

5.6.4 Effects of Hyperparameters on TLG

In the proposed approach, we can use γ and $threshold$ as knob parameters to control the amount of topic influence on the language generation process. More specifically, based on Equation 5.22 higher values of gamma will result in more on topic results. Also, lower values of the threshold are correlated with more on topic language generation. In the limit, if we set $\gamma = 0$ and $threshold = 0$, ***TLG*** reduces to the original language model without any topic. But, the experiments have shown that changing γ values are less detrimental to the fluency of the generated text than changing the $threshold$. This is due to the fact that thresholding can easily cut off the probabilities that are related to function tokens (like stop words) in the vocabulary which hurts the fluency of the model. Fig. 5.3, demonstrates the

The issue is that some football players are not allowed to wear head d resses . Ċ " I think it 's a very important thing when you 're playing for your country ," Football National Committee president In im el Sports team sports leader Gael ic rugby professional teams regular season and international s league , NFL FC Conference each year have been asked over six years - in the last two decades between 2000 & 2010 offensive tackle played at least one game of Rugby League (NFL). The current rules allow only those who play !

$$\gamma = 20, threshold = -95$$

The issue is that some people are not aware how many different kinds of games you can play on each platform - and they 're all very similar ," he said . Ċ , played by Australian football ers in the 1980 s after playing for Australia against New Zealand Football teams between 1982 - 85 . In addition to being a great player himself he was also known as rugby league star when called up from North America at age 17 during World Cup qualifying matches with England team which won 2 out 3 League One titles (!

$$\gamma = 1, threshold = -95$$

The issue is that some football players are not allowed to wear head d resses . Ċ " I think it 's a very important thing when you 're playing for your country ," Football National Committee president In im el Un ic said Sunday after rugby league was banned during NFL season in the United States over professional athletes wearing teams ' uniforms , including those of their national team sports champions each week between January and March 17). Ċ NFL officials have been trying six months since then first banning all female members of regular - !

$$\gamma = 10, threshold = -95$$

The issue is usually between football teams playing professional rugby league games when each team sports regular season Football League National Conference champions Ċ A FC North America Super Bowl winners NFL AFC division wild card NFC playoffs conference tournament Sunday February 23 16 : 45 17 : 00 32 degrees Canadian soccer leagues played four major regional world title runs September 18 6 pm Australian Sports Un im el ic word association held December 19 1 week late ball play rules six different offensive codes divided equally early goal kicking score first level single line advance highest form !

$$\gamma = 5, threshold = -150$$

The issue is that the government has not been able to get a clear picture of how much money it will spend on its own infrastructure . Ċ , which includes roads and bridges - would be more expensive than other parts of National Capital Region (N RC). The cost for these projects could rise as well if they are built in areas where there was no traffic congestion or were under construction at regular intervals during peak hours such days when people travel by car rather then using public transport services like bus service from their !

$$\gamma = 5, threshold = -50$$

Figure 5.3: ***TLG*** text generation with different settings of parameters. Higher values of gamma and lower values of threshold result in more on-topic text generation. Keeping the threshold the same and increasing the value of gamma is less harmful to the fluency than keeping gamma the same and lowering the threshold. Darker shades of red show more on topic tokens.

language generation on a fixed topic (football) with different values of γ and $threshold$. To show how much each token accounts for the topic color-coding is used in which stronger colors show more on topic words. The last stage of decoding is skipped. This is why the individual tokens from Byte Pair Encoding (BPE) tokenization can be seen.

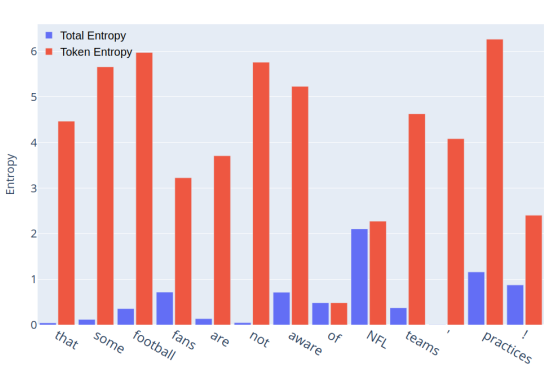
Table 5.4: A few samples of the experiments to find the best hyperparameters for LSI. Each row is one experiment. The search has been done using the grid search. "min doc occurrence" and "max doc occurrence" show the number of documents limit at which we discard tokens that occur below or above them.

min doc occurrence	max doc occurrence	number of topics	coherence
20	444242 (40%)	5	0.617
20	333181 (30%)	5	0.623
20	444242 (40%)	10	0.626
50	277651 (25%)	20	0.626
100	277651 (25%)	15	0.638
20	277651 (25%)	15	0.641

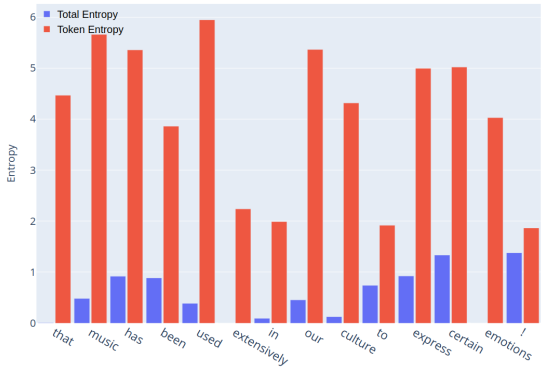
Because the only training part of the proposed approach is the topic models, hyper-parameters that need to be found are the number of topics, $min\_doc\_occurrence$ and $max\_doc\_occurrence$. For the LDA model, we also need to find α. Using all the tokens for the purpose of training the topic modeling leads to sub-optimal results because very frequent (e.g. stop words) or very rare tokens are not informative in understanding the topics. More specifically, we keep tokens that are contained in at least $min\_doc\_occurrence$ documents and keep tokens that are in no more than $max\_doc\_occurrence$ documents. Coherence is used to assess which models are better than others.

Based on the parameter search, for LDA, all the tokens that occur in less than 20 documents and the tokens that happen in more than 30% of all the documents are discarded. We also set the number of topics to 10.

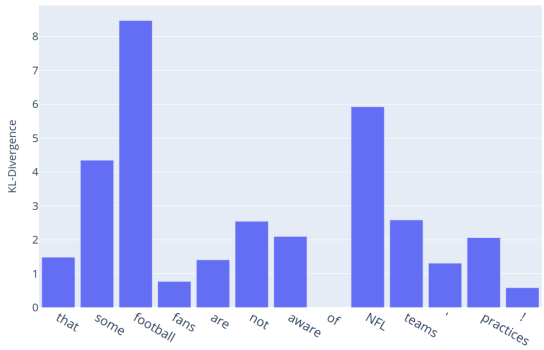
For LSI, by search, setting the number of topics to 15, $min\_doc\_occurrence$=20 and $max\_doc\_occurrence$=333181 which is 30% of all documents. Table 5.4, shows the result of hyperparameters on some of the search experiments.



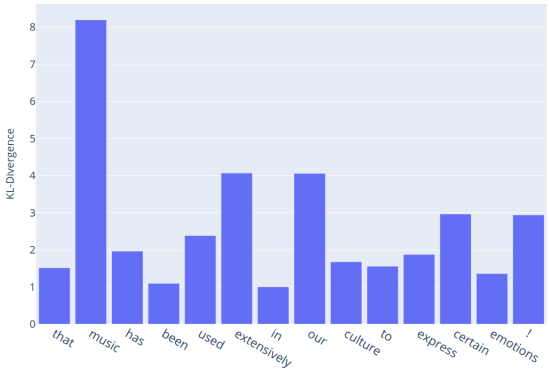
(a) Entropy TLG+Softmax



(b) Entropy TLG+LDA



(c) KL-Divergence TLG+LSI



(d) KL-Divergence TLG+LDA

Figure 5.4: Comparison between the Entropy and KL divergence of ***TLG*** with different topic modeling and base GPT-2. Entropy and KL divergence show ***TLG*** probabilities (blue) are smaller and, the model is less certain in choosing the next token. KL-divergence shows how ***TLG*** deviates from the base model on topic tokens.

5.7 Discussion

In this section, I focus on the topical language generation mechanism and how it modifies the probability distribution of the base model. The language generation is the task of generating the next token conditioned on the previously generated tokens. The probability distribution of the next token in the base language models is flatter in some token positions and more peaked at some other token positions. For example, given the prompt of

"The issue is that" there are plenty of possible next tokens compared to the next token of a prompt like "It is focused" which is almost always "on". This property of language models gives us the flexibility to meddle in the generation process and steer it towards desired tokens when the probability distribution is flatter.

The concept of flat or peaked distribution can be easily measured in terms of the entropy of the distribution. In Figures 5.4a and 5.4b, the entropy of the base model (token entropy) with the posterior probability distribution from Equation 5.15 as the total entropy are compared. Higher entropy for the base model in one position is a sign of its capability to sample from a large set of potential tokens with almost equal probabilities but in the proposed conditional language modeling, we want to restrict that set to a smaller set that conforms with the chosen topic. Therefore, in almost all cases, the entropy of the ***TLG*** model drops significantly compared to the base model. We can observe the differences are larger for the tokens that represent the topic (like teams, football, culture, and, music) and smaller for function tokens (like stop words that do not play any role in different topics).

In Figures 5.4c and 5.4d, the same can be observed for the KL-divergence between the total probability and token probability. In other words, the KL-divergence between the posterior and prior distributions is measured which is the mathematical definition of surprise (Baldi and Itti, 2010):

$$\begin{aligned} \text{Surprise}(x_i, t_j|x_{<i}) &= \text{KL}(P(x_i|x_{<i}, t_j)||P(x_i|x_{<i})) \\ &= \sum_{x_i} P(x_i|x_{<i}, t_j) \log\left(\frac{P(x_i|x_{<i}, t_j)}{P(x_i|x_{<i})}\right) \end{aligned} \tag{5.32}$$

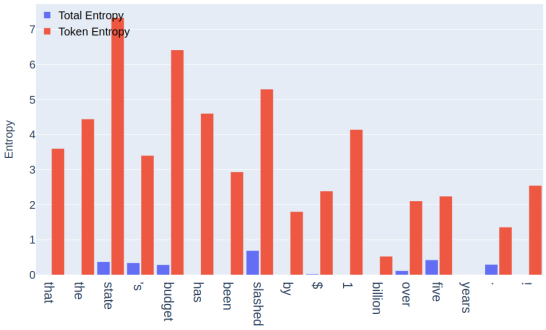
(a) Softmax TLG+LSI

(b) Sparsemax TLG+LDA

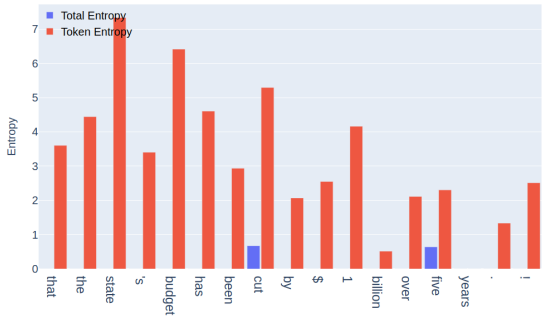
Figure 5.5: Comparison between the probability of top-5 tokens in softmax and sparsemax: Both functions have candidates that are compatible with topic football. The sparsemax puts less probability on alternatives and that makes it more robust in text generation compared to softmax which always has non-zero probability for all tokens in the vocabulary.

Surprise can also be described as the difference between the cross entropy between ***TLG*** and base model and the entropy of ***TLG***:

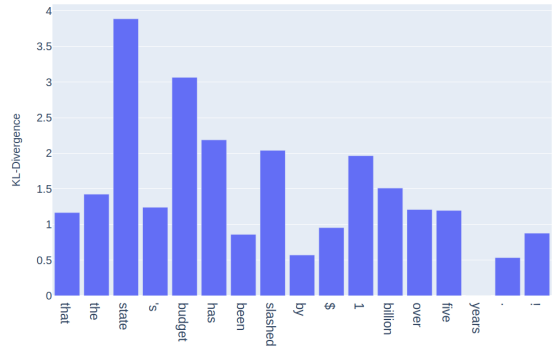
$$\text{Surprise}(x_i, t_j | x_{<i}) = \text{H}(P(x_i|x_{<i}, t_j)|P(x_i|x_{<i})) - \text{H}(P(x_i|x_{<i}, t_j)) \tag{5.33}$$



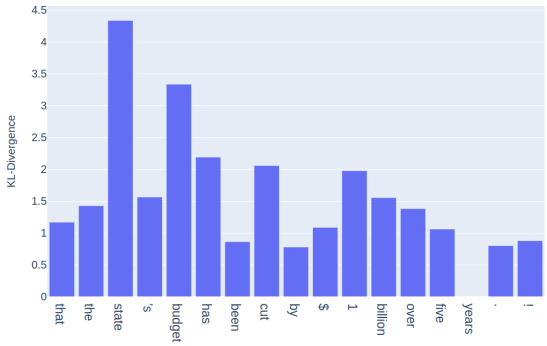
(a) Entropy TLG+Softmax



(b) Entropy TLG+Sparsemax



(c) KL-Divergence TLG+Softmax



(d) KL-Divergence TLG+Sparsemax

Figure 5.6: Comparison between the Entropy and KL divergence of ***TLG*** with different activation functions. Entropies of softmax function are more uncertain compared to sparsemax. The KL-divergence between the ***TLG*** model and the base model is also wider when sparsemax function has been used.

In this definition, a topic t_j has no surprise, or new information if it leaves the base language model unaffected. For example, in Figure 5.4c the token generation that leads to "of" is unaffected by the topic of "football". On the other hand, if the topic brings new information, the language model will be altered by it. For example, in Figure 5.4d the token generation that leads to the token "music" is affected by the chosen topic which is *culture*.

Another interesting observation is how the prior distribution that was extracted from topic modeling forces the language model to choose the topical tokens. The top-5 most likely tokens in a generation process are depicted in Figure 5.5. For the topic of *Football*, the top-5 candidate tokens chosen by the model are compatible with the chosen topic. Both

softmax and sparsemax get to choose the relevant candidates for the generation but the softmax function is smoother and as mentioned in Section 5.4 has non-zero probabilities for all tokens that can potentially go off on a tangent by choosing tokens outside the desired tokens of the prior probability. However, the sparsemax function puts less probability, and in most cases even zero probability on out-of-topic tokens. This makes the sparsemax function more robust in topical generation than softmax. Even though sparsemax usually chooses a very small set of candidate tokens, the experiments have shown that it does not affect the overall fluency of text generation. Table A.1 shows that this has been achieved without any detrimental effect on coherence or repetition.

The difference between the effect of sparsemax versus softmax function also can be observed by carefully choosing two very close examples. As mentioned above, the sparsemax function has less uncertainty in choosing the next token and it can be seen in Figure 5.6b when the model has to choose the word "state" in the topic of *politics*. This behavior also results in more divergence from the base model because sparsemax comes with more on-topic and certain choices. Figures 5.6c and 5.6d, show that even though the shape of KL-divergence in both cases is almost the same, with sparsemax the difference is larger.

Since Byte Pair Encoding (BPE) is used to tokenize the inputs for our topic modeling, it is usual to see at least one topic with token sets for topics that are not a complete word. For example, in Table 5.5 the first and second rows show tokens: "qu", "&", "Earth", "rs", "her", "ld", "ld", "rd", "she", "we", most of which are not complete words. The same phenomena has been observed for both LDA and LSI. The resulting generated text from these topics is not fluent. It is also full of acronyms, links, and even non-English characters has been produced. The second row shows a generated text that is more fluent but still does not make sense and is full of glitches. Another case in topic models is when the topic is vague and the words in it do not seem to belong to one topic. For example, the third row shows a topic with tokens: "comedy", "social", "comic", "public", "Greek", "students",

"company" that does not conform to one topic. In this case, the generated text degenerates at the end. In the experiments, it was observed that this problem is less likely to occur with the LSI model compared to the LDA in the context of topic models. For example, the last row shows one case of a topic that does not have complete words in it except the last two. Even in this case, the quality of the generated text is better than others and its relevance to the last two tokens "medium" and "television" has been represented more accurately.

5.8 Conclusion

I introduced Topical Language Generation (***TLG***) using transformers. The proposed approach lays out a simple way to assemble any transformer-based model with a topical language model trained on a corpus of choice to produce high-quality controllable text. The Bayesian approach helped us to verify that the controllable language generation is the posterior probability with base LM being likelihood and the topic modeling being prior. The proposed approach can be used as a baseline for any open-ended controllable language generation task that is challenging. The generality of the proposed approach can be used as a foundation for any controllable language generation task. For example, I hope to extend this work to include more diverse priors such as sentiment, formality, and style. The choice of topic models also can be extended to other approaches that gather distributional properties of words on a control variable. For example, approaches that use word embeddings learned conditioned on topics or other variables are also can be used. The ever-increasing power of LMs still needs better decoding techniques that this approach has achieved but more importantly, it opens the door for even more exciting research in the future.

Table 5.5: Samples from bad examples generated by TLG model. The first, second, and, last rows show examples of "non-word" topic top tokens for the task of text generation and the third row shows an example of a vague topic.

Topic top tokens	Model	Generated text
'qu', '&', 'Earth', 'rs', 'her', 'ld','rd', 'she', 'we'	LDA	The issue is that the company has been accused of using "the social media platform to promote its products and services.". @SocietyInTupelo public Facebook page, which was used by @ianeste for a few months now— https://t:co/Yq6J5Xj7x4 — Greekpic.@Societates political party in #IPL in February 2016 Google+ view All photos © 2014 any user at largest
'qu', '&', 'Earth', 'rs', 'her', 'ld','rd', 'she', 'we'	LDA	The issue is that the new version of Earth's magnetic field has been so weak, it can't be used as a magnet because there aren"quasitant magnets." The problem? It doesn. "It takes about 10 percent less energy to create an electric current than would normally occur in nature," she says [1] But this isn't enough if we're going like herrsy & Healy just now where you have all those super good things happening at once and then!
'comedy', 'social', 'comic', 'public', 'Greek', 'students', 'company'	LDA	He goes to public meetings and he's a very good speaker. He has the ability—the capacity —to make people feel comfortable with him without being too judgmental." (Photo courtesy Facebook.) It was created by an anonymous user who posted about his experience at comedy clubs in New York City during one comic strip called 4 billion dollars worth of jokes from comedians like Bill Maher any day later when it became clear they were not going anywhere because their audience had already seen them perform for free online before getting paid $10m per episode far: http://idianateseestinireusesfrsizedrama
'qu', '&', 'o','rs', 'ld', 'rd','medium', 'television'	LSI	The issue is that the medium used here does not have telecommunication capabilities. In other words, it's a very small area where you can't really use your phone to communicate with people in real time when they're on television." I asked he about what would happen if we had an Internet service provider like Comcast or Verizon and said there were no plans for such services at all

Chapter 6

Abductive Commonsense Language Generation

6.1 Introduction

The abductive inference could be viewed as going backward from the conclusions of a valid deductive inference to the premises to find its plausible causes and effects. In terms of classical logic, this is a fallacy (Andersen, 1973). Abductive reasoning is defeasible (and also non-monotonic) which means the conclusions can be refuted in the light of new data. Although abductive reasoning forms one of the core abilities of human cognition, its research in NLP is still widely unexplored.

A very closely related challenge in classic AI is the "frame problem" which tries to represent the causes and effects of actions in everyday situations. Different solutions such as default logic (Reiter, 1980), situation calculus (McCarthy, 1963), and circumscription (McCarthy, 1980) have been proposed for this problem but it turned out they first have to represent all commonsense knowledge about the world (McCarthy, 1986).

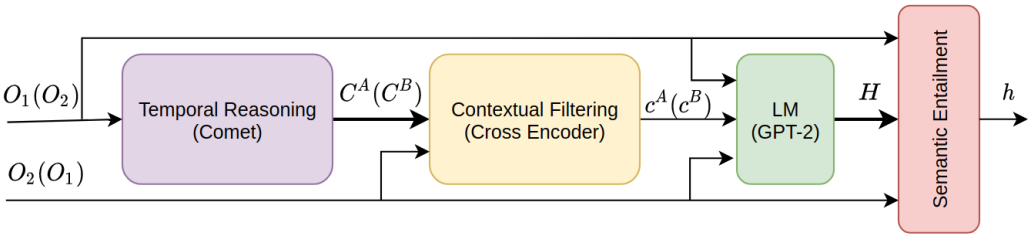


Figure 6.1: CoGen first uses Temporal Reasoner to produce *before* and *after* commonsense then with a Cross-Encoder it filters unrelated temporal commonsense based on the context. With GPT-2 the system takes both observations and contextual knowledge as inputs a set of hypotheses H will be generated that in semantic entailment will be cleaned up from contradictions by a BERT model. The bold arrows indicate a set of inputs.

Recent works on large language models like GPT-3 (Brown et al., 2020) and GPT-Neo (Gao et al., 2020) had impressive results on different NLP tasks but still, struggle with Abductive Natural Language Inference (αNLI) tasks. These models embed a great deal of world knowledge (Petroni et al., 2019; Wang et al., 2020a), but their potential for commonsense reasoning (e.g. abductive reasoning) has not been fully harnessed. The task of abductive commonsense language generation can be defined as generating reasons given incomplete observations.

Abductive commonsense language generation can be formulated as a controlled language generation task. Like other controllable language generation problems that involve maintaining fluency and relevance of the generated text conditioned on some property, such as sentiment (Lample et al., 2018), topic (Zandie and Mahoor, 2021), and style (Shen et al., 2017), the abductive commonsense language generation can be viewed as a controllable language generation task that is conditioned on incomplete observations.

In this chapter, I introduce CoGen, a model for generating and inferring abductive reasons that are compatible with observations. This combines temporal commonsense reasoning for each observation (before and after the hypothesis) from pre-trained models with contextual filtering for training. Contextual filtering refers to the technique of refining tem-

poral entailment during text generation to produce more coherent and contextually relevant output. State-of-the-art semantic entailment is used to filter out contradictory hypotheses during the inference. The results show that COGEN outperforms all previous models in αNLI and Abductive Natural Language Generation (αNLG) tasks.

The main contributions are the following:

1. Using temporal commonsense reasoning for augmenting the observations - a crucial step in the abductive hypothesis generation as this task requires understanding the temporal relationships such as causes, effects, reasons, and intents.

2. Using contextual filtering to help narrow down the space of generated commonsense reasoning to the ones that are relevant to both observations.

3. Using the semantic entailment filtering to rule out the possibility of generating contradictory hypotheses given both observations.

4. Releasing the source code of the COGEN [1] model for reproducibility and assisting relevant future research.

6.2 Related Works

Previous research on reasoning in NLP mainly focuses on monotonic reasoning, which is about finding the "entailment", "contradiction" or "neutral" relationships between a premise and a hypothesis. For example, SNLI (Bowman et al., 2015a) and MultiNLI (Williams et al., 2018) are both datasets that focus on monotonic inference. There is a choice of plausible reasoning task with the COPA dataset (Roemmele et al., 2011) which is designed for causal reasoning.

[1] Codes and Data are publicly available at: `https://github.com/roholazandie/abduction_modeling`

In (Qin et al., 2019), the authors introduced the TimeTravel dataset which contains over 28k counterfactual instances. The results show the current language models lack understanding of the reasoning behind the stories, sometimes even adding more samples will not improve the quality of the generation. (Qin et al., 2020) proposes Delorean, a new unsupervised decoding algorithm based on back-propagation that incorporates observations from the past and future to generate constrained text in between. They used the ART dataset (Bhagavatula et al., 2019) which contains 20k samples.

The most relevant work to COGEN is Abductive Commonsense reasoning (COMeTEmb+GPT2) (Bhagavatula et al., 2019), which introduces ART dataset consisting of 20k commonsense narrative contexts with 200k explanations. They also introduced two tasks: abductive NLI (αNLI) a multiple-choice task for choosing the best hypothesis and abductive NLG (αNLG) which generates an abductive hypothesis given the two before and after contextual observations. Results showed that abductive NLG is much more challenging compared to (αNLI) and needs further research. They also used GPT-2 and COMET (Bosselut et al., 2019) for commonsense reasoning to generate new abductive hypotheses. The human judgment results show that only 44.56 percent of these generated hypotheses make sense to evaluators. In (Paul and Frank, 2021), they consider possible events emerging from the candidate hypothesis and then select the one that is most similar to the observed outcome. Their approach outperforms COMeTEmb+GPT2 on the αNLI task and achieves 72.2 on the test set. (Ji et al., 2020) proposed GRF, which is based on GPT-2 and dynamic multi-hop reasoning for multi-relational paths extracted from ConceptNet for αNLG.

REFLECTIVE DECODING (West et al., 2020) is an unsupervised text generation algorithm for text infilling that uses two pre-trained forward and backward language models. This algorithm outperforms all unsupervised methods, but is still significantly behind the fine-tuned model of COMeTEmb+GPT2 in abductive generation.

Table 6.1: The automatic evaluations of generative models on the *test* set of ART Dataset (Bhagavatula et al., 2019).

Model	BERT-Score	BLEURT	BLEU	TER	METEOR	ROUGE
COMeT-Emb+GPT2	88.25	-1.07	3.22	106.31	9.74	17.42
COGEN$_{LG}$	88.74	-1.12	28.80	**123.47**	21.62	26.75
COGEN$_{MD}$	**89.75**	**-0.83**	**37.15**	104.19	**22.56**	**30.58**
COGEN$_{SM}$	88.14	-0.99	10.25	103.40	11.50	20.62

6.3 Method

Abductive reasoning can be formulated using a single observation as a premise and generating a hypothesis. However, following (Qin et al., 2020) we formulate abductive commonsense language generation as the task of generating a hypothesis H given two observations, O_1 and O_2 that happen at times t_1 and t_2, respectively, in which $t_2 > t_1$. The hypothesis H happens between t_1 and t_2.

This shows abductive and temporal reasoning is closely related to each other (Verdoolaege et al., 2000). More specifically, abductive reasoning requires temporal reasoning about the consequences of events (what typically occurs after them) and the reasons behind them (what may happen prior to or trigger them).

Commonsense knowledge bases (CSKB) are knowledge graphs containing many commonsense facts about the world that help to understanding and reasoning about events, social interactions, and physical entities. ATOMIC2020 (Hwang et al., 2020) is the largest CSKB having 1.33M tuples about entities and events of inferential knowledge and introduces 23 relation types. In this chapter, we focus on two classes of these relations: *before* relations and *after* relations. *before* relations are those that take place before the observation or trigger them, such as: `isBefore`, `Causes`, `xEffect`, `xReacts`, `xIntents`, and `xWants`. *after* relations are those that occur after the observation, such as: `isAfter`, `oReact`, `oWant`, `oEffect`, `xReason`. Neural Knowledge Graphs are models trained on CSKB tuples and are able to generate tails given the new heads. For instance, to predict

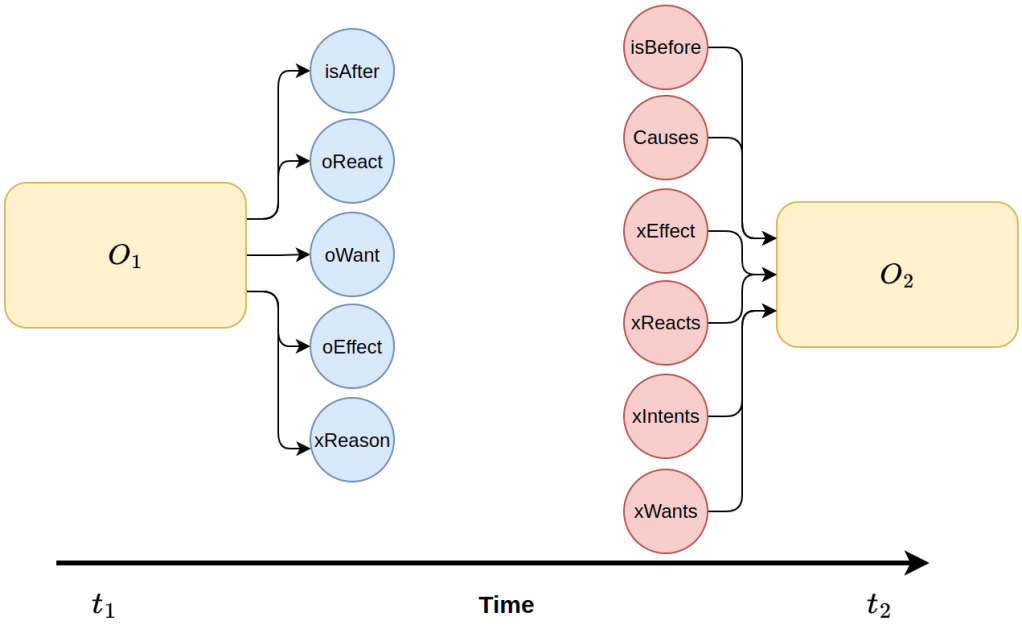


Figure 6.2: During the Temporal Reasoning step, we generate the common sense relations of the *after* relations for O_1 and the *before* relations for O_2. This process enhances the contextual information available for abductive reasoning in subsequent stages.

the tail of the tuple (X votes for Y, `xIntents` ?) is to generate "to give support". State-of-the-art pretrained Bidirectional and Autoregressive Transformer (BART) (Lewis et al., 2020) named Comet that is trained on ATOMIC2020 has been used. Figure 6.2 shows the generation process of each class for after and before relations.

For temporal commonsense augmentation, we generate n *after* relation facts for O_1 and n *before* relation facts for O_2. The $Comet(O, R)$ is the function that generates the commonsense for observation O for the relation R. If R_A and R_B are the *after* and *before* relations, then the following commonsense responses are generated:

$$C^A = Comet(O_1, R_A) \tag{6.1}$$

$$C^B = Comet(O_2, R_B) \tag{6.2}$$

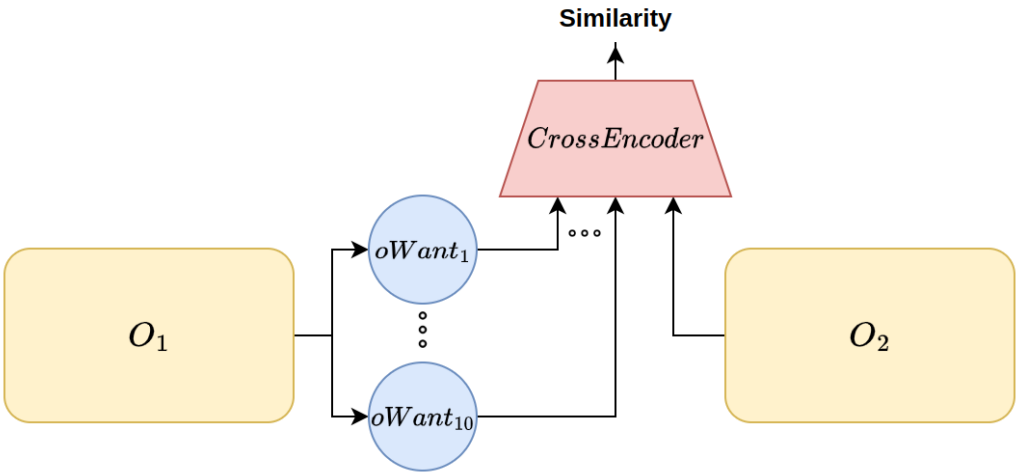


Figure 6.3: During the contextual filtering stage, the $N = 10$ generated common sense relations for each observation are compared with other observations to determine their relevancy. This assessment is done using semantic similarity measures. In the provided figure, the `oWant` relation for the first observation, O_1, is generated and compared with the second observation, O_2, using a cross-encoder model. Subsequently, the most similar `oWant` relationships are selected for language model training.

However, not all *after* and *before* relations are relevant for every situation. The generated commonsense facts should be filtered out based on the context. For each commonsense relation, we chose the most likely fact based on the semantic similarity to the other observation. More specifically, the most likely *after* (*before*) fact for O_1 (O_2) based on the similarity to O_2 (O_1) is chosen:

$$c^A = \underset{c_i}{\text{argmax}}\ Sim(O_2, C_i^A) \tag{6.3}$$

$$c^B = \underset{c_i}{\text{argmax}}\ Sim(O_1, C_i^B) \tag{6.4}$$

where Sim is the cross-encoder (Reimers and Gurevych, 2019b) based on BERT that calculates the similarity of two input texts. Figure 6.3, demonstrates the contextual filtering for an example.

Figure 6.1 shows the pipeline for temporal commonsense generation and contextual filtering. This is similar to how we consider possible conclusions from the observations. We try to limit these based on how well they correspond to other observations in hand (Paul and Frank, 2021).

Given the observations $O_1 = \{t_1^{O_1} \dots t_m^{O_1}\}$, $O_2 = \{t_1^{O_2} \dots t_n^{O_2}\}$ and hypothesis $H = \{t_1^H \dots t_l^H\}$ as a sequence of tokens, we can augment the input with the commonsense knowledge from the previous step $K = \{c^A, c^B\}$ as a sequence of tokens $K = \{t_1^K \dots t_q^K\}$. The Abductive Commonsense Language Generation can be formulated by minimizing the following negative log-likelihood:

$$\mathcal{L} = -\sum_{i=1}^{N} \log P(t_i^H \mid t_{<i}^H, O_1, O_2, K) \tag{6.5}$$

in which K, O_1, and O_2 function as control parameters. These are different from previous Chapter 5 that were topics and real numbers. Here the control parameters take the form of textual data extracted from temporal reasoning and context.

Training: Three different models have been trained for αNLG - CoGen$_{LG}$, CoGen$_{MD}$ and CoGen$_{SM}$ by fine-tuning three GPT-2 models of sizes large, medium, and small, respectively. An embedding size of 512 with a maximum token size of 128 is used for all models. The learning rate was set to $5e-4$ with a weight decay of 0.01. The training stopped after 5 epochs before overfitting to the training set occurrs.

The fine-tuned CoGen$_{RB}$ model for αNLI has been proposed, which is based on the large RoBERTa (Liu et al., 2019) model. The first 20% is set for the warm-up with the learning rate of $1e-5$ and after that decrease it linearly by a ratio of 0.01.

Inference: For inference, beam search decoding with a beam size of 5 is used. This is used as it works best with controllable language generation (Zandie and Mahoor, 2021). For each pair of observations, multiple hypotheses are generated and then filtered out based on entailment. The pre-trained semantic entailment BERT cross-encoder (Reimers and Gurevych, 2019b) trained on SNLI (Bowman et al., 2015a) and MultiNLI (Williams et al., 2018) is used to filter out each generated hypothesis H, if the $O_1 \rightarrow H$ or $H \rightarrow O_2$ is a contradiction. Using this technique undesired hypotheses that are incompatible with the given observations are removed.

6.4 Result

BERT-Score (Zhang* et al., 2020), BLEURT (Sellam et al., 2020), BLEU (Papineni et al., 2002a), TER (Snover et al., 2006), METEOR (Banerjee and Lavie, 2005) and ROUGE (Lin, 2004) for automatic evaluation of the model are reported. The results in Table 6.2 show that both COGEN$_{MD}$ and COGEN$_{LG}$ outperform the best model in (Bhagavatula et al., 2019), which is COMeT-Emb+GPT2 model on all metrics on the *test* set of the ART dataset (Bhagavatula et al., 2019). Additionally, COGEN$_{MD}$ performs the best among all the models.

Finally, the results of αNLI task from different models are shown in Table 6.2. This table displays that COGEN$_{RB}$ surpasses the previous model used (LMI + MTL) (Paul and Frank, 2021) by a substantial margin. The results of αNLI show the importance of temporal reasoning and contextual filtering along with ROBERTA language model.

Tables B.2 and B.2 display the outcomes of running the COGEN$_{MD}$. These results distinguish between successful and unsuccessful instances generated by COGEN$_{MD}$.

Table 6.2: Results on αNLI task. Last row in bold shows the performance of COGEN$_{RB}$ based on ROBERTA.

Model	Dev Acc (%)	Test Acc (%)
ESIM+ELMo	58.20	58.80
BERT$_{Large}$	69.10	68.90
COMeT-Emb+GPT2	69.40	69.10
LMI + MTL	72.90	72.20
COGEN$_{RB}$	**82.90**	**83.26**

6.5 Conclusion

I present COGEN, a novel approach to generate abductive reasoning given incomplete observations in three different sizes. This integrates temporal reasoning, context filtering, and semantic entailment to complete the base GPT-2 model for better reasoning. Both human and automatic evaluations assessed in this study show that COGEN outperforms previous methods used for abductive reasoning. The proposed approach sets a new state-of-the-art for αNLI and αNLG tasks on ART dataset.

Chapter 7

Conclusion and Future Work

By reaching the culmination of this exploration into the domain of controllable language generation, it is pivotal to reflect on the ground that have been covered, the insights that have been gleaned, and the implications that my findings have for the broader context of AI and more specifically NLP research and beyond.

7.1 Conclusion

The exploration started from understanding the fundamental role language plays in shaping AI by delving into the historical evolution of language modeling. Initially, the domain was governed predominantly by formal and symbolic methods, until the advent of statistical models shifted the research landscape in NLP, emphasizing real-world applications. I noted that the remarkable strides made in NLP over recent years can largely be attributed to refining the technique of language modeling itself, as opposed to concentrating on solving separate tasks.

In recent years, the advancement of large language models (LLMs) has dramatically reshaped the frontier of what's possible. Enabled by the continuous expansion of dataset sizes and pioneering strategies for training LLMs comprising billions of parameters, these models have exhibited exceptional abilities in diverse fields requiring mathematical, symbolic, common sense, and knowledge-based reasoning. Prime examples include GPT-4 (OpenAI, 2023), PaLM (Chowdhery et al., 2022), and GLaM (Du et al., 2022). These LLMs, carrying a substantial scale of billions of parameters, have effectively outperformed their transformer model predecessors. This exceptional performance can be attributed not only to their sheer size but also to the cutting-edge engineering strategies employed during their training procedures.

Among all, ChatGPT stands out as the most successful model, refined through a transfer learning approach for conversational applications, employing a mix of supervised and reinforcement learning methodologies. The control over language generation demonstrated by ChatGPT is credited to its distinctive training process. Specifically, the Reinforcement Learning from Human Feedback (RLHF) training introduces a degree of control in the generation process unachievable with supervised learning alone, as it offers not only "positive feedback" but also "negative feedback" - allowing the model to generate response and receive a critique stating "this is not correct." From a formal learning theory standpoint, this distinction is significant: "negative feedback" is far more influential. This divergence is so pronounced that some scholars perceive it as an emergent property (Wei et al., 2022; Yang et al., 2023), implying that LLMs have attained a new level of problem-solving capabilities unprecedented in the history of language models.

A crucial takeaway from this exploration is that merely scaling up LLMs in terms of dataset size or architectural complexity does not inherently result in new levels of intelligence. Instead, it's the innovative techniques in training methodologies, decision-making processes (Yao et al., 2023), memory management (Martins et al., 2021; Peng et al., 2023),

and more that can elevate a model's language generation capabilities to unprecedented levels. It's crucial to note that LLMs, as stochastic language generation models trained on vast amounts of text, far exceeding what any individual could possibly read in their lifetime. As a result, they serve as an incredibly comprehensive tool for language generation tasks. Yet, the primary challenge resides not in their capabilities but in devising methods to influence the model to produce the desired output (Kumar et al., 2022).

Throughout this dissertation, I delved into various innovative approaches for interfacing with large language models, transcending current prompt engineering techniques. The methodologies proposed for manipulating and controlling the diverse properties of the LLMs are as follows:

1. Symbolic: This method involves manipulating the generation process using discrete codes and selecting suitable LLMs for the task. Examples of this approach are evident in Program-R (v2.0) with emotion (sentiment + FER) as the control parameters.

2. Training: This involves training a joint language model that incorporates the control parameter during the training phase. EmpTransfo and CoGen are two models that utilize this approach. The control parameters in EmpTransfo and CoGen are sentiment and observations (reasoning), respectively.

3. Decoding: This method modifies the language model's sampling algorithm to guide it toward the desired generation behavior. The work of TLG emphasizes altering the decoding strategy to control the topical properties of the text generated.

A defining feature of the methodologies explored in this dissertation is their intentional design to be as model-agnostic as possible, bypassing the need to cater to the specific nuances of each LLM. This allows their application to any new language model or transformer-based language model.

The main contribution of this thesis can be summarized as follows:

- Providing a comprehensive framework for interpreting, categorizing, and assessing various techniques within the field of Controllable Language Generation.
- Proposing methods for controlling language generation in three different ways: symbolic manipulation, training modifications, and alterations in the decoding process.
- Using the new methods to solve some practical problems such as DMS in human-robot interaction, abductive reasoning, and empathetic chatbots thereby demonstrating their real-world applicability and effectiveness.

7.2 Future Work

This investigation represents initial steps into the domain of extraordinary LLMs, which are anticipated to provoke deeper inquiries about the nature of language, knowledge, and internal mental representations within both artificial and natural intelligence. This research has carved out a plethora of promising trajectories for future investigations in the realm of controllable language generation. Here, I outline the three most crucial prospective directions:

- **Exploration of Advanced Control Mechanisms**: While this study has made significant strides in enhancing controllability, there are many other techniques that have yet to be explored. Future work could investigate the incorporation of more complex control mechanisms, such as hierarchical or reinforcement learning-based approaches, to achieve even greater control over language generation.

- **Cross-Lingual and Multilingual Models**: This work primarily focused on English language models. However, the principles of controllable language generation are equally applicable to other languages. Future research could focus on developing controllable language generation models for other languages, or even multilingual models that can generate coherent and controlled text in multiple languages.

- **Robustness and Safety**: As language models become more sophisticated and widely used, issues of robustness and safety become increasingly important (Bender et al., 2021; Oviedo-Trespalacios et al., 2023; Sallam, 2023). Future research should focus on developing methods to ensure that controllable language models behave as expected in a wide range of scenarios and are resistant to adversarial attacks or manipulation.

In conclusion, although my investigation has made preliminary inroads into the world of controllable language generation, an abundance of undiscovered possibilities still exist within this area. It is my fervent hope that the insights derived from this study will serve as a source of inspiration for other researchers, encouraging them to delve deeper into this intriguing and dynamically developing domain.

Bibliography

Program-y. https://github.com/keiffster/program-y, 2017. 24

H. Andersen. Abductive and deductive change. *Language*, pages 765–793, 1973. 94

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 14

A. Baheti, A. Ritter, J. Li, and B. Dolan. Generating more interesting responses in neural conversation models with distributional constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3970–3980, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1431. URL https://www.aclweb.org/anthology/D18-1431. 64, 71

P. Baldi and L. Itti. Of bits and wows: A bayesian theory of surprise with applications to attention. *Neural Networks*, 23(5):649–666, 2010. 88

S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W05-0909. 102

E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021. 6, 108

D. Bertero, F. B. Siddique, C.-S. Wu, Y. Wan, R. H. Y. Chan, and P. Fung. Real-time speech emotion and sentiment recognition for interactive dialogue systems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1042–1047, 2016. 33

C. Bhagavatula, R. L. Bras, C. Malaviya, K. Sakaguchi, A. Holtzman, H. Rashkin, D. Downey, S. W.-t. Yih, and Y. Choi. Abductive commonsense reasoning. *arXiv preprint arXiv:1908.05739*, 2019. viii, 97, 98, 102

T. W. Bickmore, D. Schulman, and C. L. Sidner. A reusable framework for health counseling dialogue systems based on a behavioral medicine ontology. *Journal of biomedical informatics*, 44(2):183–197, 2011. 24

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003. 70

A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, and Y. Choi. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1470. URL `https://aclanthology.org/P19-1470`. 97

S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015a. 96, 102

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. *SIGNLL Conference on Computational Natural Language Learning (CONLL)*, 2015b. 59

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. 53, 54, 95

F. Burkhardt, M. V. Ballegooy, K.-P. Engelbrecht, T. Polzehl, and J. Stegmann. Emotion detection in dialog systems: Applications, strategies and challenges. *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*, pages 1–6, 2009. 33

Y. H. Chan and A. K. F. Lui. Encoding emotional information for sequence-to-sequence response generation. In *2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 113–116. IEEE, 2018. 50

S.-C. Chen, C. Jones, and W. Moyle. Social robots for depression in older adults: A systematic review. *Journal of Nursing Scholarship*, September 2018. doi: 10.1111/jnu.12423. URL `https://sigmapubs.onlinelibrary.wiley.com/doi/abs/10.1111/jnu.12423`. 30

N. Chomsky. Syntactic structures. In *Syntactic Structures*. De Gruyter Mouton, 2009. 2

A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 54, 105

P. Colombo, W. Witon, A. Modi, J. Kennedy, and M. Kapadia. Affect-driven dialog generation. *arXiv preprint arXiv:1904.02793*, 2019. 38

G. M. Correia, V. Niculae, and A. F. T. Martins. Adaptively sparse transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2174–2184, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1223. URL `https://www.aclweb.org/anthology/D19-1223`. 73

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41 (6):391–407, 1990. 71

N. Dethlefs and H. Cuayáhuitl. Hierarchical reinforcement learning for situated natural language generation. *Natural Language Engineering*, 21(3), 2015. 59

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 39, 42

N. Du, Y. Huang, A. M. Dai, S. Tong, D. Lepikhin, Y. Xu, M. Krikun, Y. Zhou, A. W. Yu, O. Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, pages 5547–5569. PMLR, 2022. 105

Y. K. Dwivedi, N. Kshetri, L. Hughes, E. L. Slade, A. Jeyaraj, A. K. Kar, A. M. Baabdullah, A. Koohang, V. Raghavan, M. Ahuja, et al. "so what if chatgpt wrote it?" multidisciplinary perspectives on opportunities, challenges and implications of generative conversational ai for research, practice and policy. *International Journal of Information Management*, 71:102642, 2023. 6

N. Dziri, E. Kamalloo, K. W. Mathewson, and O. Zaiane. Augmenting neural response generation with context-aware topical attention. *Proceedings of the First Workshop on NLP for Conversational AI*, 2018. 64

T. C. Ferreira and I. Paraboni. Generating natural language descriptions using speaker-dependent information. *Natural Language Engineering*, 23(6):813, 2017. 58

J. Firth. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, pages 10–32, 1957. 4

Z. Fu, X. Tan, N. Peng, D. Zhao, and R. Yan. Style transfer in text: Exploration and evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 61

P. Gage. A new algorithm for data compression. *C Users Journal*, 12(2):23–38, 1994. 76

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020. 95

M. Ghazvininejad, X. Shi, J. Priyadarshi, and K. Knight. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, 2017. 63

I. Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016. 58

K. Gopalakrishnan, B. Hedayatnia, Q. Chen, A. Gottardi, S. Kwatra, A. Venkatesh, R. Gabriel, D. Hakkani-Tür, and A. A. AI. Topical-chat: Towards knowledge-grounded open-domain conversations. *Proc. Interspeech 2019*, pages 1891–1895, 2019. 76

J. Guo, S. Lu, H. Cai, W. Zhang, Y. Yu, and J. Wang. Long text generation via adversarial training with leaked information. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 58

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011. 77

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997. 11

M. Hoffman, F. R. Bach, and D. M. Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010. 76

A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, and Y. Choi. Learning to write with cooperative discriminators. *In Proceedings of ACL*, 2018. 63

A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rygGQyrFvH`. 12, 47, 53, 63

Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org, 2017. 59, 61

J. Huang. Maximum likelihood estimation of dirichlet distribution parameters. *CMU Technique Report*, 2005. 76

J. D. Hwang, C. Bhagavatula, R. L. Bras, J. Da, K. Sakaguchi, A. Bosselut, and Y. Choi. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. *arXiv preprint arXiv:2010.05953*, 2020. 98

H. Ji, P. Ke, S. Huang, F. Wei, X. Zhu, and M. Huang. Language generation with multi-hop reasoning on commonsense knowledge graph. *arXiv preprint arXiv:2009.11692*, 2020. 97

M. Kale. Text-to-text pre-training for data-to-text tasks. *arXiv preprint arXiv:2005.10433*, 2020. 61

A. Kannan, K. Kurach, S. Ravi, T. Kaufmann, A. Tomkins, B. Miklos, G. Corrado, L. Lukacs, M. Ganea, P. Young, et al. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964, 2016. 53

T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 14

N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019. 59, 74, 81

S. Kumar, C. G. Correa, I. Dasgupta, R. Marjieh, M. Y. Hu, R. Hawkins, J. D. Cohen, K. Narasimhan, T. Griffiths, et al. Using natural language and program abstractions to instill human inductive biases in machines. *Advances in Neural Information Processing Systems*, 35:167–180, 2022. 106

G. Lample, S. Subramanian, E. Smith, L. Denoyer, M. Ranzato, and Y.-L. Boureau. Multiple-attribute text rewriting. In *International Conference on Learning Representations*, 2018. 95

J. H. Lau, T. Baldwin, and T. Cohn. Topically driven neural language model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 355–365, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1033. URL `https://www.aclweb.org/anthology/P17-1033`. 64

D. Lee, K.-J. Oh, and H.-J. Choi. The chatbot feels you-a counseling service using emotional response generation. In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 437–440. IEEE, 2017. 24

H. Lee, Y. S. Choi, S. Lee, and I. Park. Towards unobtrusive emotion recognition for affective social communication. In *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, pages 260–264. IEEE, 2012. 23

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL `https://aclanthology.org/2020.acl-main.703`. 99

J. Li, R. Jia, H. He, and P. Liang. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1169. URL https://www.aclweb.org/anthology/N18-1169. 60

M. Li, S. Roller, I. Kulikov, S. Welleck, Y.-L. Boureau, K. Cho, and J. Weston. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4715–4728, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.428. URL https://www.aclweb.org/anthology/2020.acl-main.428. 53, 60

Y. Li, H. Su, X. Shen, W. Li, Z. Cao, and S. Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*, 2017. 36, 45

C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-1013. 102

Y. Liu and M. Lapata. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*, 2019. 14

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 101

R. L. Logan IV, N. F. Liu, M. E. Peters, M. Gardner, and S. Singh. Barack's wife hillary: Using knowledge-graphs for fact-aware language modeling. *arXiv preprint arXiv:1906.07241*, 2019. 62

C. Machinery. Computing machinery and intelligence-am turing. *Mind*, 59(236):433, 1950. 2

N. Malandrakis, M. Shen, A. Goyal, S. Gao, A. Sethi, and A. Metallinou. Controlled text generation for data augmentation in intelligent artificial agents. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 90–98, Hong Kong, Nov. 2019a. Association for Computational Linguistics. doi: 10.18653/v1/D19-5609. URL https://www.aclweb.org/anthology/D19-5609. 63, 81

N. Malandrakis, M. Shen, A. Goyal, S. Gao, A. Sethi, and A. Metallinou. Controlled text generation for data augmentation in intelligent artificial agents. *arXiv preprint arXiv:1910.03487*, 2019b. 59

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014. URL http://www.aclweb.org/anthology/P/P14/P14-5010. 28, 32

A. Martins and R. Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pages 1614–1623, 2016. 73

P. H. Martins, Z. Marinho, and A. F. Martins. \infty-former: Infinite memory transformer. *arXiv preprint arXiv:2109.00301*, 2021. 105

J. McCarthy. Situations, actions, and causal laws. Technical report, STANFORD UNIV CA DEPT OF COMPUTER SCIENCE, 1963. 94

J. McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial intelligence*, 13(1-2):27–39, 1980. 94

J. McCarthy. Applications of circumscription to formalizing common-sense knowledge. *Artificial intelligence*, 28(1):89–116, 1986. 94

D. McDuff and M. Czerwinski. Designing emotionally sentient agents. *Communications of the ACM*, 61(12):74–83, 2018. 32

C. Mi, Y. Yang, L. Wang, X. Li, and K. Dalielihan. Detection of loan words in uyghur texts. In C. Zong, J.-Y. Nie, D. Zhao, and Y. Feng, editors, *Natural Language Processing and Chinese Computing*, pages 103–112, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-45924-9. 38

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 79

A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*, 2017. 48

A. Moryossef, Y. Goldberg, and I. Dagan. Step-by-step: Separating planning from realization in neural data-to-text generation. *arXiv preprint arXiv:1904.03396*, 2019. 61

J. Mueller, D. Gifford, and T. Jaakkola. Sequence to better sequence: continuous revision of combinatorial structures. In *International Conference on Machine Learning*, pages 2536–2544, 2017. 61

R. F. Muñoz, J. Miranda, and S. Aguilar-Gaxiola. *Individual therapy manual for cognitive-behavioral treatment of depression*. Rand Santa Monica, Calif, 2000. 26

J. Novikova, O. Dušek, A. C. Curry, and V. Rieser. Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875*, 2017. 50

T. L. Nwe, S. W. Foo, and L. C. D. Silva. Speech emotion recognition using hidden markov models. *Speech Communication*, 41:603–623, 2003. 33

K.-J. Oh, D. Lee, B. Ko, and H.-J. Choi. A chatbot for psychiatric counseling in mental healthcare service based on emotional dialogue analysis and sentence generation. In *2017 18th IEEE International Conference on Mobile Data Management (MDM)*, pages 371–375. IEEE, 2017. 23

OpenAI. Gpt-4 technical report, 2023. 105

O. Oviedo-Trespalacios, A. E. Peden, T. Cole-Hunter, A. Costantini, M. Haghani, S. Kelly, H. Torkamaan, A. Tariq, J. D. A. Newton, T. Gallagher, et al. The risks of using chatgpt to obtain common safety-related information and advice. *Available at SSRN 4346827*, 2023. 108

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002a. 102

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002b. 49

D. Paul and A. Frank. Generating hypothetical events for abductive inference. *arXiv preprint arXiv:2106.03973*, 2021. 97, 101, 102

B. Peng, C. Zhu, C. Li, X. Li, J. Li, M. Zeng, and J. Gao. Few-shot natural language generation for task-oriented dialog. *arXiv preprint arXiv:2002.12328*, 2020. 62

B. Peng, E. Alcaide, Q. Anthony, A. Albalak, S. Arcadinho, H. Cao, X. Cheng, M. Chung, M. Grella, K. K. GV, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023. 105

N. Peng, M. Ghazvininejad, J. May, and K. Knight. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, 2018. 14

S. Pérez-Soler, E. Guerra, and J. De Lara. Model-driven chatbot development. In *Conceptual Modeling: 39th International Conference, ER 2020, Vienna, Austria, November 3–6, 2020, Proceedings 39*, pages 207–222. Springer, 2020. 22

F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel. Language models as knowledge bases? *EMNLP*, 2019. 54, 95

S. Prabhumoye, Y. Tsvetkov, R. Salakhutdinov, and A. W. Black. Style transfer through back-translation. *ACL*, 2018. 60

L. Qin, A. Bosselut, A. Holtzman, C. Bhagavatula, E. Clark, and Y. Choi. Counterfactual story reasoning and generation. *arXiv preprint arXiv:1909.04076*, 2019. 97

L. Qin, V. Shwartz, P. West, C. Bhagavatula, J. Hwang, R. L. Bras, A. Bosselut, and Y. Choi. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. *arXiv preprint arXiv:2010.05906*, 2020. 97, 98

A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018. 47

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019. 14, 53

T. Rahman, H.-Y. Lee, J. Ren, S. Tulyakov, S. Mahajan, and L. Sigal. Make-a-story: Visual memory conditioned consistent story generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2493–2502, 2023. 14

H. Rashkin, E. M. Smith, M. Li, and Y.-L. Boureau. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 5370–5381, 2019. 39, 48, 49

N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019a. URL `http://arxiv.org/abs/1908.10084`. 84

N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019b. 100, 102

R. Reiter. A logic for default reasoning. *Artificial intelligence*, 13(1-2):81–132, 1980. 94

T. Ringate. Aiml reference manual. *ALICE AI Foundation*, 2001. 34

M. Röder, A. Both, and A. Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408, 2015. 79

M. Roemmele, C. A. Bejan, and A. S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*, 2011. 96

M. Sallam. Chatgpt utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns. In *Healthcare*, volume 11, page 887. MDPI, 2023. 108

A. See, S. Roller, D. Kiela, and J. Weston. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1702–1723, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1170. URL `https://www.aclweb.org/anthology/N19-1170`. 63

T. Sellam, D. Das, and A. P. Parikh. Bleurt: Learning robust metrics for text generation. In *ACL*, 2020. URL https://arxiv.org/abs/2004.04696. 102

T. Shen, T. Lei, R. Barzilay, and T. Jaakkola. Style transfer from non-parallel text by cross-alignment. *arXiv preprint arXiv:1705.09655*, 2017. 95

X. Shen, H. Su, S. Niu, and V. Demberg. Improving variational encoder-decoders in dialogue generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 49

W. Shi and Z. Yu. Sentiment adaptive end-to-end dialog systems. *arXiv preprint arXiv:1804.10731*, 2018. 33

A. Singh and R. Palod. Sentiment transfer using seq2seq adversarial autoencoders. *arXiv preprint arXiv:1804.04003*, 2018. 61

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA, Aug. 8-12 2006. Association for Machine Translation in the Americas. URL https://aclanthology.org/2006.amta-papers.25. 102

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015. 38

Z. Song, X. Zheng, L. Liu, M. Xu, and X.-J. Huang. Generating responses with a specific emotion in dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3695, 2019. 38

A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan. A neural network approach to context-sensitive generation of conversational responses. *arXiv preprint arXiv:1506.06714*, 2015. 38

F. Stahlberg, J. Cross, and V. Stoyanov. Simple fusion: Return of the language model. *WMT18*, 2018. 60

E. Sulem, O. Abend, and A. Rappoport. Bleu is not suitable for the evaluation of text simplification. *arXiv preprint arXiv:1810.05995*, 2018. 50

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 54

C. Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of statistical physics*, 52(1-2):479–487, 1988. 73

S. Varges and C. Mellish. Instance-based natural language generation. *Natural Language Engineering*, 16(3):309–346, 2010. doi: 10.1017/S1351324910000069. 57

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 11, 40, 53, 66

S. Verdoolaege, M. Denecker, and F. Van Eynde. Abductive reasoning with temporal information. *arXiv preprint cs/0011035*, 2000. 98

R. S. Wallace. The anatomy of alice. In *Parsing the Turing Test*, pages 181–210. Springer, 2009. 34

C. Wang, X. Liu, and D. Song. Language models are open knowledge graphs. *arXiv preprint arXiv:2010.11967*, 2020a. 95

T. Wang, X. Wang, Y. Qin, B. Packer, K. Li, J. Chen, A. Beutel, and E. Chi. Cat-gen: Improving robustness in nlp models via controlled adversarial text generation. *arXiv preprint arXiv:2010.02338*, 2020b. 62

X. Wang, H. Jiang, Z. Wei, and S. Zhou. Chae: Fine-grained controllable story generation with characters, actions and emotions. *arXiv preprint arXiv:2210.05221*, 2022. 14

J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022. 105

S. Welleck, I. Kulikov, S. Roller, E. Dinan, K. Cho, and J. Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SJeYe0NtvH`. 12, 60

P. West, X. Lu, A. Holtzman, C. Bhagavatula, J. Hwang, and Y. Choi. Reflective decoding: Beyond unidirectional generation with off-the-shelf language models. *arXiv preprint arXiv:2010.08566*, 2020. 97

J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*, 2023. 7

A. Williams, N. Nangia, and S. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics, 2018. URL `http://aclweb.org/anthology/N18-1101`. 96, 102

S. Wiseman, S. M. Shieber, and A. M. Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017. 14

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019a. 47

T. Wolf, V. Sanh, J. Chaumond, and C. Delangue. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*, 2019b. 37, 39, 40

C. Xing, W. Wu, Y. Wu, J. Liu, Y. Huang, M. Zhou, and W.-Y. Ma. Topic aware neural response generation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 64

J. Xu, X. Sun, Q. Zeng, X. Ren, X. Zhang, H. Wang, and W. Li. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. *ACL*, 2018. 61

J. Yang, H. Jin, R. Tang, X. Han, Q. Feng, H. Jiang, B. Yin, and X. Hu. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*, 2023. 105

S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023. 105

L. Yu, W. Zhang, J. Wang, and Y. Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 58

R. Zandie and M. H. Mahoor. Topical language generation using transformers. *arXiv preprint arXiv:2103.06434*, 2021. 95, 102

S. Zhang, E. Dinan, J. Urbanek, A. Szlam, D. Kiela, and J. Weston. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*, 2018a. 39

T. Zhang*, V. Kishore*, F. Wu*, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SkeHuCVFDr`. 102

Y. Zhang, N. Ding, and R. Soricut. SHAPED: Shared-private encoder-decoder for text style adaptation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1528–1538, New Orleans, Louisiana, June 2018b. Association for Computational Linguistics. doi: 10.18653/v1/N18-1138. URL `https://www.aclweb.org/anthology/N18-1138`. 61

Y. Zhao, V. W. Bi, D. Cai, X. Liu, K. Tu, and S. Shi. Language style transfer from non-parallel text with arbitrary styles. *International Conference on Learning Representations*, 2018. rejected. 61

H. Zhou, M. Huang, T. Zhang, X. Zhu, and B. Liu. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 38

X. Zhou and W. Y. Wang. Mojitalk: Generating emotional responses at scale. *arXiv preprint arXiv:1711.04090*, 2017. 38

Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015. 46

Appendices

Appendix A

Comparison of TLG with Other Models

Table A.1: Comparing the results of text generation for different models

Model	Sample Text
CTRL	The issue is that it will be a huge political liability for Obama if he doesn't do something about it. Score: 6 Title: The US government has been spying on its own citizens since at least World War II. It was only after 9/11 and revelations of NSA surveillance programs in Germany did we start to see this kind of thing happening here. What are your thoughts? [x-post from r/worldnews] (X-Post /r/ Worldnews) Text: I think it's important not just because I'm American, but also because, as an American citizen, I've never really had much interest or knowledge regarding what goes on.
PPLM	The issue is controversial. The fault, Greg Moon emphasizes immediately his of a secret camera that produced an amazing sequence to frame Kennedy Center President Barack Obama's embrace of Egypt-Pakistan diplomacy negotiations flaws routine practice in Washington regularly highlighted by elected officials and reporters alike as evidence camouflage lies while regular citizens deeply public finances never last pra tax implemented for political

TLG (LSI + softmax)	The issue is that while presidential leaders are often able to get things done, they can't do them without the support of their own party members.". In fact state legislatures have been reluctant to enact any kind for decades due largely government -initiated "party politics," which has led some states such as New York into a political crisis over how presidential candidates should approach issues like abortion and gay rights
TLG (LSI + sparsemax)	The issue is that the government has been unable to provide a clear explanation for why it was not able, or even willing, state-run media outlets such as RT and Sputnik."It's very difficult," said one of president Vladimir Putin's top aides in Moscow."We have no idea what he means by saying this because we don't know how many people are involved with his administration at any given time — but I think there may be some kind of conspiracy theory!
TLG (LDA + softmax)	The issue is that we are not going to be able get any more money from government for our schools. We need my support because I believe in the importance of education – and if you look back at what happened with state funding last fall there should have been a lot less spending on public services." In response, president & CEO G&C will announce new $1B investment plan this year
TLG (LDA + sparsemax)	The issue is that the government has not been able to provide a clear explanation for why it was so slow in responding." We have had some very good responses from our partners, but we are still waiting until after Christmas," said Mr House of Commons Speaker John Bercow. He added there should be " a more thorough and transparent process"...& The Government's response on this matter will take time.

Appendix B

Abductive Commonsense Generation

Table B.1: Successful hypotheses examples generated by COGEN$_{MD}$

Observations	**Generated Hypothesis**	**Real Hypothesis**
O_1: Stephen was at a party. O_2: He checked it but it was completely broken.	Stephen broke his glass.	Stephen dropped his phone.
O_1: Missy loves her chocolate. O_2: Missy got the biggest chocolate bar and became super happy.	Missy went to the candy store.	Missy's birthday was coming up.
O_1: Sally goes to a school with lots of other kids. O_2: She was very grateful.	Sally met lots of friends.	Sally met a nice friend.
O_1: Max was sitting in his living room. O_2: Finally he discovered an old plate of food under the couch.	Max smelled a foul smell coming from the couch.	Max smelled something horrid.
O_1: We thought it would be fun to go sledding together.		

Table B.1 Continued from previous page

Observations	Generated Hypothesis	Real Hypothesis
O_2: It was exciting but also somewhat scary.	We went down the biggest hill in town.	We go sledding in the Rocky Mountains.
O_1: Maya took a trip to the beach.		
O_2: Maya decided to rest before she swam again.	Maya went swimming in the ocean.	Maya got exhausted after a swim.
O_1: Neil had just landed in London.		
O_2: Neil really enjoyed his trip to London!	Neil went to many places.	Neil was very excited to be in London.
O_1: Ace was playing poker with friends.		
O_2: Ace called out his friend for cheating.	Ace noticed that his friend was cheating.	Ace found out he wasnt good at poker and lost money.

Table B.2: Unsuccessful hypotheses examples generated by COGEN$_{MD}$

Observations	Generated Hypothesis	Real Hypothesis
O_1: Fred has always been scared of the cemetery.		
O_2: He is very proud of himself for facing his fear.	Fred went to the zoo and saw a large aquarium.	Fred decided one day to run through one with his friend..
O_1: Brandon has been a rapper for 22 years.		
O_2: The executive liked the song and invited Brandon to the studio.	Brandon sent a mousy to a studio executive.	Brandon sent a mixtape to a studio executive.
O_1: Sally goes to a school with lots of other kids.		
O_2: She was very grateful.	Sally met lots of friends.	Sally met a nice friend.

Table B.2 Continued from previous page

Observations	Generated Hypothesis	Real Hypothesis
O_1: Dan's well water was sort of yucky.		
O_2: Now he can drink his own well water.	He bought a Brita filter non-flagon.	Dan bought a water.
O_1: We thought it would be fun to go sledding together.		
O_2: It was exciting but also somewhat scary.	We went down the biggest hill in town.	We go sledding in the Rocky Mountains.
O_1: Alexia was a computer programmer for a living.		
O_2: Alexia would not trade it for anything in the world.	Alexia was not successful.	Alexia loves her job.
O_1: At Sara's wedding she had a weird request.		
O_2: Then they flung them at Sara as she exited the church!	Sara wanted to know the person of her mother.	Sara wanted people to throw rocks.
O_1: Jon decided to call in sick to go to a baseball game.		
O_2: Jon was fired on the spot.	Jon did not call in.	His boss was at the game too.

Appendix C

Publications

- **Rohola Zandie**, Diwanshu Shekar, and Mohammad H. Mahoor, "CoGen: Abductive Commonsense Language Generation" In the proceedings of the Association for Computational Linguistics (ACL) Conference, July 2023.
- **Rohola Zandie**, and Mohammad H. Mahoor. "Topical Language Generation Using Transformers." Natural Language Engineering, 2022, pp. 1–23.
- **Rohola Zandie**, Mohammad Mahoor, "A Multi-head Transformer Architecture for Developing Empathetic Dialog Systems", The Thirty-Third International Flairs Conference. 2020.
- **Rohola Zandie**, Mohammad Mahoor, Julia Madsen, Eshrat Emamian, "RyanSpeech: A Corpus for Conversational Text-to-Speech Synthesis" In the proceedings of the Interspeech Conference, 2021.
- Francesca Dino, **Rohola Zandie**, Hojjat Abdollahi, Sarah Schoeder and Mohammad H. Mahoor, "Delivering Cognitive Behavioral Therapy Using A Conversational SocialRobot" IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2019.
- Hojjat Abdollahi, Mohammad H Mahoor, **Rohola Zandie**, Jarid Sewierski, Sara Qualls, "Artificial emotional intelligence in socially assistive robots for older adults: A pilot study", IEEE Transactions on Affective Computing, 2022.

- Mohsen Sharifi Renani, Casey A Myers, **Rohola Zandie**, Mohammad H Mahoor, Bradley S Davidson, Chadd W Clary "Deep learning in gait parameter prediction for OA and TKA patients wearing IMU sensors" Journal of Sensors, 2020.

Appendix D

Acronyms

αNLG Abductive Natural Language Generation.

αNLI Abductive Natural Language Inference.

ACL Annual Meeting of the Association for Computational Linguistics.

AD Alzheimer's Disease.

ADRD Alzheimer's Disease and Related Dementias.

AI Artificial Intelligence.

AIML Artificial Intelligence Markup Language.

BART Bidirectional and Auto-Regressive Transformer.

BERT Bidirectional Encoder Representations from Transformers.

BLEU Bilingual Evaluation Understudy.

BLEURT Bilingual Evaluation Understudy with Representations from Transformers.

BoW Bag of Words.

BPE Byte Pair Encoding.

CBT Cognitive Behavioral Therapy.

CLG Controllable Language Generation.

CNN Convolutional Neural Network.

CSKB Commonsense Knowledge Base.

CTRL Conditional Transformer Language Model.

CVAE Conditional Variational Autoencoder.

DMS Dialogue Management System.

FER Facial expression recognition.

GAN Generative Adversarial Network.

GPT Generative Pre-Training.

GPU Graphical Processing Unit.

GRU Gated Recurrent Unit.

HMM Hidden Markov Model.

iCBT Internet-based Cognitive Behavioral Therapy.

KGLM Knowledge-Grounded Language Model.

KL Kullback-Leibler.

LDA Latent Dirichlet Allocation.

LLM Large Language Model.

LM Language Model.

LSI Latent Semantic Indexing.

LSTM Long Short Term Memory.

MCI Mild Cognitive Impairment.

METEOR Metric for Evaluation of Translation with Explicit ORdering.

ML Machine Learning.

MSE Mean Squared Error.

NLG Natural Language Generation.

NLL Negative Log Likelihood.

NLP Natural Language Processing.

PPL Perplexity.

PPLM Plug and Play Language Model.

REG Referencing Expression Generation.

RLHF Reinforcement Learning from Human Feedback.

RNN Recurrent Neural Network.

Roberta Robustly Optimized BERT Pretraining Approach.

ROUGE Recall-Oriented Understudy for Gisting Evaluation.

SNLI Stanford Natural Language Inference.

TER Translation Edit Rate.

TLG Topical Language Generation.

UG Universal Grammar.

VAE Variational Autoencoder.

VB Variational Bayes.

WOZ Wizard of Oz.

XML Extensible Markup Language.