Université de Montréal

A Personality Aware Recommendation System

par

Fahed Elourajini

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en informatique

Le 6 août 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

A Personality Aware Recommendation System

présenté par

Fahed Elourajini

a été évalué par un jury composé des personnes suivantes :

Michel Boyer
(président-rapporteur)

Esma Aïmeur
(directeur de recherche)

Bang Liu
(membre du jury)

Résumé

Les systèmes de recommandation conversationnels (CRSs) sont des systèmes qui fournissent des recommandations personnalisées par le biais d'une session de dialogue en langage naturel avec les utilisateurs. Contrairement aux systèmes de recommandation traditionnels qui ne prennent comme vérité de base que les préférences anciennes des utilisateurs, les CRS impliquent aussi les préférences actuelles des utilisateurs durant la conversation. Des recherches récentes montrent que la compréhension de la signification contextuelle des préférences des utilisateurs et des dialogues peut améliorer de manière significative les performances du système de recommandation. Des chercheurs ont également montré un lien fort entre les traits de personnalité des utilisateurs et les systèmes de recommandation. La personnalité et les préférences sont des variables essentielles en sciences sociales. Elles décrivent les différences entre les personnes, que ce soit au niveau individuel ou collectif. Les approches récentes de recommandation basées sur la personnalité sont des systèmes non conversationnels. Par conséquent, il est extrêmement important de détecter et d'utiliser les traits de personnalité des individus dans les systèmes conversationnels afin d'assurer une performance de recommandation et de dialogue plus personnalisée. Pour ce faire, ce travail propose un système de recommandation conversationnel sensible à la personnalité qui est basé sur des modules qui assurent une session de dialogue et recommandation personnalisée en utilisant les traits de personnalité des utilisateurs. Nous proposons également une nouvelle approche de détection de la personnalité, qui est un modèle de langage spécifique au contexte pour détecter les traits des individus en utilisant leurs données publiées sur les réseaux sociaux. Les résultats montrent que notre système proposé a surpassé les approches existantes dans différentes mesures.

Mots-clés: Prédiction de la personnalité, Traitement du langage naturel, Apprentissage par transfert, Transformateurs, Systèmes de recommandation de conversation.

Abstract

A Conversational Recommendation System (CRS) is a system that provides personalized recommendations through a session of natural language dialogue turns with users. Unlike traditional one-shot recommendation systems, which only assume the user's previous preferences as the ground truth, CRS uses both previous and current user preferences. Recent research shows that understanding the contextual meaning of user preferences and dialogue turns can significantly improve recommendation performance. It also shows a strong link between users' personality traits and recommendation systems. Personality and preferences are essential variables in computational sociology and social science. They describe the differences between people, both at the individual and collective level. Recent personality-based recommendation approaches are traditional one-shot systems, or "non conversational systems". Therefore, there is a significant need to detect and employ individuals' personality traits within the CRS paradigm to ensure a better and more personalized dialogue recommendation performance.

Driven by the aforementioned facts, this study proposes a modularized, personality-aware CRS that ensures a personalized dialogue recommendation session using the users' personality traits. We also propose a novel personality detection approach, which is a context-specific language model for detecting individuals' personality traits using their social media data. The goal is to create a personality-aware and topic-guided CRS model that performs better than the standard CRS models. Experimental results show that our personality-aware conversation recommendation system has outperformed state-of-the-art approaches in different considered metrics on the topic-guided conversation recommendation dataset.

Keywords: Personality prediction, natural language processing, transfer learning, transformers, prompting, conversation recommendation systems.

Contents

List of tables

List of figures

List of abbreviations

CRS	Conversational Recommendation System
NLU	Natural Language Understanding
DST	The Dialogue State Tracker
RE	Recommendation Engine
NLG	Natural Language Generation
NLP	Natural Language Processing
NL	Natural Language
PL	Prompt-based Learning
DL	Deep Learning
ML	Machine Learning

LM	Language Models
IE	Information Extraction
PD	Personality Detection
DS	Dialogue Systems
CS	Conversational Systems
DST	Dialogue State Tracking
DTP	Dialogue Topic Tracking
PPLM	Plug and Play Language Model
AI	Artificial Intelligence
QA	Question-Answering
SA	Sentiment Analysis
MT	Machine Translation
IR	Information Retrieval

ATS	Automatic Text Summarization
POS	Part Of Speech tagging
NER	Named Entity Recognition
STSR	Sequence To Sequence Recommendation
Big5	The Five-Factor personality
MBTI	Myers Briggs Type Indicator
I	Introversion
E	Extroversion
N	Intuition
S	Sensing
T	Thinking
F	Feeling
J	Judging

P	Perceiving
EXT	Extroversion
NEU	Neuroticism
AGR	Agreeableness
CSN	Conscientiousness
OPN	Openness
GNN	Graph-based Neural Network
GAT	Graph-Attention
TF-IDF	Term Frequency-Inverse Document Frequency
DARPA	Defence Advanced Research Projects Agency
PPerMo	A Prompt-based and Personality-aware Modularized CRS
GRU	Gated Recurrent Units
SAUP	System is Active User is Passive

SAUE	System is Active user Engaged
SAUA	System is Active user is Active
UASP	User is active, the system is passive
BERT	Bidirectional Encoder Representations from Transformers
ALBERT	A Lite BERT for Self-supervised Learning of Language Representations
MLM	Masked Language Modeling
NSP	Next Sentence Prediction
ELECTRA	Efficiently Learning an Encoder that classifies Token Replacement Accurately
PEPLER	Personalized Prompt Learning for Explainable Recommendation
SVM	Support Vector Machine
LSTM	Long Short-Term Memory
CNN	Convolutional Neural Network

P-r-C	Pearson Correlation Coefficient
MLP	Multi Layer Perceptrons
RNN	Recurrent Neural Network
AWS-EP	A Multi-task learning approach for individuals personality prediction
PMLM	Pseudo-Masked Language Models for Unified Language Model Pre-Training
RTD	Replaced Token Detection
API	Application Programming Interface
DBT-SR	DistilBERT with Transfer learning trick for Sequential Recommendation
Elec-SP	ELECTRA for State Prediction
PPPG-DialoGPT	A Personality and Preference-aware Prompt-based Goal-oriented DialoGPT model

Acknowledgments

I would like to take this opportunity to thank all of those who contributed to achieving this work.

I would like to express my deepest gratitude to my incredible supervisor, Professor Esma Aïmeur, for her valuable guidance and unlimited support. I feel very fortunate to be associated and supervised by such an inspirational, caring, and helpful mentor. Without her enormous support, continuous inspiration, and valuable guidance, it would have been impossible for me to accomplish this thesis on personalized conversational recommendation systems.

My sincere gratitude goes as well to the jury members, Professor Michel Boyer and Professor Bang Liu, for sparing their precious time to review and evaluate my thesis.

I would like to thank my fellow lab mate Rim for guiding me while I'm writing my research work. I want to thank Firas, and Mohamed my roommates, for being there for me whenever I needed them.

I'm also grateful to the TakeLab members for providing us with their personality-based Pandora dataset. Without their kindness, this research step would never be accomplished.

Finally, my very profound gratitude goes to my parents and all my family members for believing in me and providing me with the support and encouragement that I needed to achieve my goals.

Chapter 1

Introduction

This chapter presents the general context of our research. It discusses the necessity for and the motivation behind the realization of this thesis. Finally, it outlines the research objectives, contribution, and organization of our thesis dissertation.

1.1. Problem Definition and Motivation

Over the decades, generating user-specific and *personalized* recommendation responses has been a crucial and complex task, especially for automatic recommendation systems. Recommendation systems are types of software applications that help users find items of interest. Having a recommendation system with adequate intelligence and topic *knowledge awareness* to effectively provide users with personalized items of interest can be very challenging. This is due to the diversity of users' behaviours and the variation of topics or preferences that a user might have or employ within a recommendation session.

In the mid-70s, recommendation systems evolved to become an independent research domain of their own. Since then, a lot of research has been done in this field. To this date, various promising recommendation paradigms have been proposed in the literature. Different techniques such as collaborative filtering, content-based filtering, knowledge-based systems, and hybrid-based approaches were the backbone of most traditional recommendation systems [**1, 2, 3**]. Despite their effectiveness at modeling the users' previous interactions, traditional systems are still incapable of providing personalized recommendations [**4, 5**]. Known as *one-shot* recommendation systems, they focus on tracking the user's behaviour over time and then providing a set of recommendations in a one-shot process. These approaches assume that the users are already aware of their current preferences and know exactly the item they are looking for, which is not always the case. Users might change their preferences over time, and they might learn about the item that they prefer only by interacting with the recommendation system [**5**]. Therefore, by using only the user's

previous monitored behaviour, one-shot recommendation systems are incapable of providing an effective personalized item recommendation.
To overcome these drawbacks, a new paradigm called *Conversational Recommendation System* (CRS) [**4**] was proposed in the literature. Unlike traditional one-shot systems, CRSs employ both previous and current user preferences during the recommendation session. They also support both users' *multi-turn* and *task-oriented* dialogues. This makes the CRS capable of understanding users' current preferences while taking into consideration their previous behaviour and interactions.
To ensure a solid natural dialogue quality and a more personalized item recommendation, this paradigm employs five different components: the *Natural Language Understanding* (NLU), the *Dialogue State Manager/Tracker* (DST), the *Recommendation Engine* (RE), the *User Model* "External Knowledge" and the *Natural Language Generation* (NLG) components, with each component having its specific role. The NLU component is responsible for understanding the meaning of user utterances. The DST component is responsible for managing the state of the dialogue (for example, deciding whether the system needs to provide a recommendation or if it should keep talking about a specific topic). The Recommendation Engine component is responsible for predicting the most convenient item for the user. The NLG component is responsible for generating a natural and meaningful response to the user while taking into consideration the current state (topic) of the conversation. The User Model component (also known as the knowledge component) includes the user-specific information (such as personality traits, profile, previous interactions, demographic information, etc.). These details help ensure more personalized recommendation sessions.

With the recent advancement in *Natural Language Processing* (NLP) and *Deep Learning* (DL) technologies, the interest in CRS has increased significantly over the past few years. Different promising approaches have been proposed in the literature to implement the CRS as a complete package known as *End-to-End* CRS or to implement each CRS component separately, known as *Modularized* CRS. A recent line of research focuses on using *pre-trained transformer* models by fine-tuning them to produce specific and personalized recommended items [**6, 7**]. Despite the promising results provided by these models on the topic-oriented conversational recommendation task [**8, 9**], these models are originally trained on large, unstructured, open-domain, textual datasets (books, movie scripts, open-domain internet information, etc.). Therefore, they do not encode the dialogue structure knowledge within their parameters. Moreover, training or fine-tuning these types of models has been shown to be computationally expensive [**10**]. Also, CRS approaches published to this date do not employ individual *personality traits* within their conversation recommendation sessions, which limits their capabilities for providing more effective personalized recommendations [**5**].

Personality traits refer to the difference among individuals' characteristic patterns, such as how they think, feel, and behave. Understanding people's preferences and their personality trait factors can significantly enhance the quality of the recommendation performance [**5**]. Different traditional recommendation approaches have applied personality trait factors within their recommendation process [**11, 12, 13, 14, 15**]. Although these systems use different recommendation techniques, most of the proposed personality-aware recommendation approaches are one-shot systems, meaning that modern CRS research does not employ personality traits within its multi-turn dialogue recommendation sessions. They generally build the dialogue session by considering different types of user-related information, such as their demographic information and/or their previous interactions. Although these approaches can personalize the recommendation process to some degree, we believe that by not employing the user's personality traits, they are still incapable of effectively building personalized dialogue sessions, as recent research has demonstrated a strong link between individuals' personality traits and personalized recommended movies [**16**].

If the user tends to be more curious about the world and eager to learn new things, we believe that the CRS should initiate a dialogue session by discussing things that interest them (for example, by talking about how exciting and mysterious life could be, to finally recommend a motivational movie about world exploration). In this way, the user may feel that he is talking to another human speaker that shares with him/her the same personality and preferences, which can significantly improve the user's trust for the recommended item and make it easy for them to accept it.

For example, if someone suggests to you a movie to watch, you might respect their taste and like their suggestion. However, if this person is shy and unsocial, in contrast to you being a social individual that enjoys new experiences, the movie might not be to your taste and thus, it is preferable if you get the recommendation from someone who shares your interests.

We strongly believe that making the CRS aware of users' personalities and acting as if it has similar traits to them will significantly improve both the user's experience and the system's recommendation performance. Therefore, the aforementioned issues are what motivated us to address the following points in this thesis:

- How to effectively learn the individuals' personality trait factors from their social media posts and employ them in a CRS framework.

- Designing a modularized conversational recommendation system that benefits from the users' personality traits.

- Evaluating the performance of each component in our modularized CRS.

- Reporting the impact of employing the user personality trait factors on each CRS component.

1.2. Research Objectives and The Main Contributions

The primary purpose of this thesis is to provide a personalized topic-guided conversational movie recommendation system that considers the individual's personality traits during the recommendation process. This system helps its users get convenient and personalized movie recommendations based on their historical interactions, profiles, personality traits and current preferences. More specifically, the contributions and objectives of this thesis are as follows:

- Propose a *weight-shared* and *multi-task* personality prediction approach that effectively predicts the user's personality from a conversation session.

- Design a *modularized*, personality-aware CRS framework by employing the user's personality traits on each module.
  To build the modularized CRS, we organized our objectives as follows:

  - Initially, propose a novel sequential recommendation module to predict the next movie to watch using a *transfer learning* trick. This module will be used with the topic prediction component to improve the CRS performance.

  - Next, develop a topic prediction module that predicts the current dialogue state within a conversation session. This component will be used by the dialogue generation module to produce more topic-aware responses.

  - After that, design a *prompt-based*, goal-oriented response generation model that uses the previous topics and individuals' personality traits as prompting controls during the response generation process.

  - Finally, experimenting and evaluating the designed approaches on different metrics and improving state-of-the-art models, as well as combining all of the designed components into a single module-based CRS.

Motivated by the importance of personality traits on recommendation systems' performance and of the goal-oriented CRS to reach effective recommendations, by the significant results provided by the recent *Language Models* (LM) and by the prompt-based learning paradigms, we propose the first prompt-based, personality-aware, topic-guided conversation recommendation system which we call **PPerMo**, "**P**rompt-based and **Per**sonality-aware **Mo**dularized CRS".

1.3. Thesis Organization

This thesis is divided into eight chapters, including the first chapter which is the introduction.

- Chapter 2 discusses the recent research done for personality prediction and conversational recommendation systems as well as prompt-based learning approaches, along with the limitations of different existing approaches.

- Chapter 3 explains our proposed PPerMo CRS framework in detail. It highlights the main components within our proposed solution as well as their working pipeline.

- Chapter 4 highlights the personality prediction model architecture, its experimental setup, and the obtained results on different datasets compared to other state-of-the-art models.

- Chapter 5 presents our proposed Recommendation Engine module the "sequential next movie-to-watch recommendation model", as well as the experimental setup and the obtained results.

- Chapter 6 discusses our Dialogue State Manager component architecture as well as the training and evaluation results.

- Chapter 7 presents our proposed prompt-based, goal-oriented dialogue recommendation system the "Dialogue Generation" module. It highlights how we managed to combine the personality features, topic, and preference information using prompting templates as input data for a pre-trained language model.

- Finally, we conclude this thesis in Chapter 8 by discussing the prospects of this research.

Chapter 2

Background and Literature Review

NLP is an artificial intelligence field that involves understanding, interpreting, manipulating and generating human spoken languages (English, French, Chinese, etc.). Different NLP fields such as Information Extraction (IE), Personality Detection (PD), Conversational Recommendation System (CRS), Dialogue State Tracking (DST), Dialogue Systems (DS) and Prompt-based Learning (PL) have grown significantly in recent. This chapter presents the IE, PD, CRS, DST, DS, and PL fields, as well as the recent research done in fields related to this thesis.

2.1. Information Extraction and Personality Detection

IE refers to the use of computational methods to highlight and filter relevant insights within an information source and convert them into a computer-based representation for storage, processing, and retrieval purposes [**17**]. Detecting relevant insights from a natural language data source is considered to be one of the hardest NLP tasks due to the complexity and ambiguity that a natural language can have on different levels. For example, in human natural languages, we can have the same meaning for different sentences as well as having different meanings for the same sentence. The sentence "At last, a computer that understands me like my mother" could mean that the computer understands me as my mother does, or it also could mean that the computer that understands me tends to like and appreciate my mother. This type of ambiguity makes the extraction of relevant insights from natural language information sources very challenging.

Moreover, different approaches are proposed in order to enhance the NLP's capabilities of understanding the sentences' ambiguity and extracting meaningful information from natural languages [**18, 19, 20**]. Extracting specific user-related information such as individuals' *preferences* or *personalities* from natural language sources is still an active, challenging research area. There are so many different characteristics involved in modeling a person's personality or preferences. Some features such as the individual's behaviour, feelings, sociability and

thinking patterns can significantly improve the information retrieval task. These patterns are hard to learn and very expensive to get. Moreover, different individuals can share the same characteristics which makes distinguishing them and extracting their personal information a very complex task to do.
Despite these complexities, different user-related sensitive IE approaches have been proposed in the literature in the past few years [**21, 22**]. In this chapter, the focus is mainly on discussing IE techniques that are related to understanding and extracting individuals' personality traits information, as this thesis's goal is to investigate the impact of using the user's personality factors within a conversational recommendation system. The next section discusses personality traits as a general concept, as well as the different IE approaches proposed within this field.

2.1.1. Personality Traits

Personality traits refer to the difference among individuals' characteristic patterns, such as how they think, feel, and behave. With the recent advances of both social media (such as Facebook and Twitter) and e-commerce platforms (for example, Amazon and eBay) [**23**], people have become more open to expressing their thoughts, emotions and complaints, whether within direct chat interfaces where they can discuss directly with another speaker about a specific topic or through comment and review interfaces where they can post their opinions and thoughts directly.
Understanding people's core personality traits and knowing what they are good at can be very important in a wide variety of situations, such as ameliorating their social relationships, improving their thinking, developing their daily interaction capabilities, etc. A lot of important information can be obtained just by knowing someone's personality. Therefore, an urgent need to develop models that automatically predict individuals' personality traits has been introduced in the literature in the past few years [**24, 25, 26**].

Over the decades, psychologists have designed different personality trait detection systems. The most popular ones are the *Five-Factor* personality (Big Five or Big5) and the *Myers Briggs Type Indicator* (MBTI) systems [**27**]. MBTI is a simple system to use when it comes to highlighting the individual's personality traits. However, when it comes to accuracy and reliability, the Big Five model is much more reliable and commonly used compared to MBTI.

- **MBTI (Myers Briggs Type Indicator):**
  The Myers Briggs Type Indicator is a personality type system that divides everyone into 16 distinct personality types across 4 axes. These axes are known as *Introversion* (I)/*Extroversion* (E), *Intuition* (N)/*Sensing* (S), *Thinking* (T)/*Feeling* (F) and *Judging* (J)/*Perceiving* (P). Figure 2.1 highlights the different MBTI traits.

Combining the 4 dimensions, this model provides 16 different categories for user

Fig. 2.1. MBTI test personality properties [**28**].

personalities (ISTJ "Responsible/Executor", ISFJ "Dedicated/Stewards", INFJ "Insightful/Motivators", INTJ "Visionary/Strategist", ISTP "Nimble/Pragmatics", ISFP "Practical/Custodians", INFP "Inspired/Crusaders", INTP "Expansive/Analyzers", ESTP "Dynamic/Mavericks", ESFP "Enthusiastic/Improvisers", ENFP "Impassioned/Catalysts", ENTP "Innovative/Explorers", ESTJ "Efficient/Drivers", ESFJ "Committed/Builders", ENFJ "Engaging/Mobilizers", EMTJ "Strategic/Directors").

In this personality system, a user can belong to just one category out of the 16 personalities, where each personality has its strengths and weaknesses [**29**].

- **Big Five model (Five-Factor model):** The Big Five personality test measures the human personality along five distinct and independent dimensions, *Extroversion* (EXT), *Neuroticism* (NEU), *Agreeableness* (AGR), *Conscientiousness* (CSN) and

Openness (OPN). During this test, a real value is applied for each dimension to indicate to which degree an individual has a specific personality property. For example, if an individual has 0.6 EXT, 0.4 EST, 0.8 AGR, 0.1 CSN, and 0.9 OPN, this means that this individual's personality is rated at 60% Extroversion, 40% Neuroticism, 80% Agreeableness, 10% Conscientiousness and 90% Openness. Figure 2.2 highlights the different Big Five personality traits.

The meaning of these personality properties are independent:

Fig. 2.2. The Big Five test personality properties [**30**].

- **Extroversion (EXT):** Extroversion refers to excitability, sociability, talkativeness, assertiveness and high amounts of emotional expressiveness. People that have a high degree of belonging to this personality property are outgoing and tend to gain energy in social situations. Also, being around other people helps them feel energized and excited [**31**].
- **Neuroticism (NEU) "Emotional Stability":** Neuroticism refers to sadness, moodiness and emotional instability. People that have a high degree of belonging to this personality property tend to experience mood swings, anxiety, irritability and sadness [**31**].

- **Agreeableness (AGR):** Agreeableness refers to trust, altruism, kindness, affection, and other related behaviours. People that have a high degree of belonging to this personality property are more cooperative than usual [**31**].
- **Conscientiousness (CON):** Conscientiousness refers to high levels of thoughtfulness, good impulse control and goal-directed behaviours. People that have a high degree of belonging to this personality property tend to plan ahead, think about how their behaviour affects others, and are mindful of deadlines [**31**].
- **Openness (OPN):** Openness refers to imagination and insight. People that have a high degree of belonging to this personality property are more curious about the world and eager to learn new things and enjoy new experiences [**31**].

Relying on these personality tests, different automatic prediction approaches have been proposed in the literature in the past few years. The next section discusses the personality trait detection approaches in detail.

2.1.2. Personality Trait Detection Approaches

With the advancement in machine learning research and the recent significant data amount availability, the ability to detect the user's personality, emotions, and preferences is now higher than ever. Despite the fact that the performance of these learning models has proven not to be accurate enough to allow for precise, traits-based people distinction, researchers argue that predictions can still be accurate on average [**32**].

To predict individuals' personalities, researchers tried to use different types of data sources, such as the users' social relationships, their regular behaviour, regular activities, communication behaviour, etc. Wu Youyou *et al.* [**26**] found that *digital footprint* information is better at measuring personality traits than other information sources such as friends, family, colleagues, etc. Many researchers have successfully applied traditional learning algorithms for personality traits detection using digital footprint information [**33, 23**]. Mark Smallcombe [**34**] argues that in modern times the amount of unstructured data (qualitative data, social media data, textual data etc.) is much larger than that of structured data (quantitative and handcrafted feature data). Moreover, Waldemar *et al.* [**35**] highlight the importance of unstructured data for human behaviour identification and detection due to the significant insights that can be mined from this type of data.

Giulio *et al.* [**36**] were the first to apply textual data for personality detection. They used a *Support Vector Machine* (SVM) model to conduct personality detection on top of textual-based features instead of using manual-crafted features and pre-defined rules.

Although using statistical learned patterns instead of pre-defined rules made the personality prediction task more flexible, using the SVM model did not provide significant personality detection results.

Following this research and with the recent advancement in automatic and deep learning fields, Yen *et al.* [**37**] used both *Long Short-Term Memory* (LSTM) and *Convolution Neural Network* (CNN) approaches to extract individuals' personalities from textual data. They used the memory gating mechanism within the LSTM network to keep track of and encode different insights within the textual source of data. They then used a CNN architecture to predict the different personality traits and provide the encoded insights. Despite the performance improvement that this approach provides compared to traditional techniques, it is still incapable of understanding the full context of a data source. This is due to the encoding capabilities of the LSTM architecture, which can only capture information in a one-side sequential order (from *left-to-right* or *right-to-left*). To overcome this issue, a new line of research attempted to use the *attention mechanism* and *transformer* models to encode information from both sides simultaneously [**38, 10**].

In their "Pandora Talks: Personality and Demographics on Reddit" paper [**39**], Gjurkovi *et al.* used a *Bidirectional Encoder Representations from Transformers* (BERT) [**7**] model to set a benchmark for their huge *Pandora* dataset [**39**], which includes three different personality tests: OCEAN (which refers to the Big Five model categories), MBTI and Enneagram tests. The authors of this paper developed six regression models (to predict both age and Big Five traits) and eight classification models (to predict the four MBTI features, gender, region, and Enneagram test features).

Experiments were conducted using traditional machine learning approaches such as linear/logistic regression as well as deep learning approaches such as *Multi-Layer Perceptrons* (MLP). In each model, the textual comments were encoded using 1024-dimensional vectors derived using BERT, which produced a new benchmark for both regression and classification tasks for this dataset, using macro F1-score and *Pearson Correlation Coefficient* (P-r-C) metrics.

The Pandora approach is the largest and first personality-based project in the research field that is trained on over 17 million *Reddit* comments written by over 10k users, annotated with both MBTI and Big Five factors. However, this approach still needs significant improvements in terms of the flexibility level. As to predict the different personality traits, this work trains over 9 different models. Each model is responsible for predicting only one personality trait. This increases the computational expense and the complexity level of their proposed solution, as well as reducing its flexibility.

Following this work, Tao Yang *et al.* [**40**] combined a *Graph-based Neural Network* (GNN) with a BERT transformer embedding model to detect the individual's personality

traits. This system uses a graph network to inject structural psycholinguistic insights as a novel approach to exploit domain knowledge within the personality prediction task. Their work consists of constructing a specific tripartite graph for each user, where each one is represented by three heterogeneous types of nodes: post, word, and category nodes. The post node represents the different posts of the users. The word node highlights the different words contained both in the users' posts and the *Linguistic Inquiry and Word Count* (LIWC) dictionary, which is a dictionary tool for psycholinguistic analysis. The category node aims to encode the psychologically relevant categories of the users' words. The edges between the different nodes are determined by the dependencies between category and word nodes as well as between post and word nodes. To improve the effectiveness of the interactions between the different nodes and to reduce the computational training cost, the authors of this paper used a *Graph-Attention* (GAT) network, which employs the attention mechanism to allow only message transmission between neighbouring parties. Using BERT to generate all nodes' initial representations and the GAT network to enhance these representations, the authors kept the averaged post node encoding as the final representation for predicting the individuals' personality. Their experiments show that their proposed model outperforms the existing state-of-the-art baselines by 3.47 and 2.10 points on the average F1-score. Despite the GAT network being used to reduce the model's computational cost, it is still computationally expensive and time-consuming due to the implementation of the BERT transformer model [**10**]. Also, this model only tackles the MBTI personality test which limits its performance accuracy as the Big Five personality test has been proven to be more accurate compared to MBTI [**27, 41**].

To further enhance the effectiveness of personality prediction models, Yang Li *et al.* proposed a new "Multitask Learning for Emotion and Personality Detection" [**42**] model. They combined the BERT transformer model and a three CNN layer model, allowing information sharing between the different layers to predict users' personalities and emotions using two different datasets (one for personality prediction and the other for *emotion prediction*). Using the multi-learning technique and the information-sharing mechanism, the authors argue that their work overcomes different state-of-the-art models on different metrics such as accuracy, macro-precision, macro-recall, and the macro-F1 metric. The contribution of their proposed framework consists of using a classification multi-task neural network with weight sharing on two different tasks (personality and emotion prediction). Although their proposed personality prediction approach outperforms state-of-the-art models on different benchmark datasets, this approach is incapable of providing efficient prediction results. The fact that the authors used BERT as their main model to understand the contextual information within user comments and posts as well as training three CNN

layers on top of it made this approach more complex and computationally expensive.

Inspired by all of the previous research and by the significant performance improvements that both transformers and the multi-task learning approaches provided, we decided to investigate the effect of using a multi-task MLP learning approach on top of a fine-tuned *Efficiently Learning an Encoder that Classifies Token Replacements Accurately* (ELECTRA) transformer model [**43**] and compare its performance to the already existing state-of-the-art baselines. We also investigate the impact of sharing weights between the MBTI and the Big Five personality tests. Furthermore, we looked at the similarity between these two personality models. More details on our proposed personality prediction approach are discussed in chapter 4.
Now that we discussed the different existing personality prediction approaches that are related to our work, in the next section we discuss the different existing recommendation system approaches, especially the conversational ones, as our thesis's main contribution combines both personality prediction and conversational recommendation systems (as shown in figure 3.1).

2.2. Conversational Recommendation Systems

As discussed in [**4**], there is no widely-established definition of what a conversational recommendation system is or what components it employs. However, most researchers agree that a common definition for CRS is that they are types of multi-turn dialogue software that support recommendation goals for a specific user [**4**].

Traditional recommendation systems often assume a one-shot interaction paradigm to predict the convenient item of interest for a specific user [**44**]. They consider the user's past observed behaviour as the ground truth to build and understand the current user's preferences. Whereas this may be true in some cases, it's generally not the case. Users' current preferences can change over time. For example, in movie recommendation systems, a user could have a history of watching action movies. However, it may be that at this moment they want to watch a comedy movie. In this case, a one-shot recommendation system cannot provide and fulfill the user's current desire. Moreover, when users have no previous information to be considered by the recommendation system, also known as the "*Cold start problem*", one-shot recommendation approaches are incapable of recommending convenient items for them. Furthermore, the users might not even know their current preferences and may construct them only during the decision process by investigating the available options. In some cases, the user might even learn about the domain and the available options only during the interaction with the recommender agent. This is not the

case with one-shot systems.

Conversational recommender systems address many of these challenges [**4**]. They propose a different recommendation paradigm and support a richer set of interactions. They also support *task-oriented* and sometimes *open-domain* multi-turn dialogue with their users. During the dialogue turns, they can construct a detailed analysis of the current user's preferences by providing item suggestion explanations and by processing the user's feedback on the suggestions. In the past few years, a variety of CRS designing approaches have been proposed in the literature. These approaches can be categorized into three different main architectures: *End-to-End*, *Data Flow*, and *Modularized* architectures.

- **End-to-End architectures:** These types of architectures rely on End-to-End *automatic learning* approaches that are trained on large corpora of recorded natural language dialogues. Despite the significant advances in NLP End-to-End techniques, it is still unclear if the final generated recommendation responses can be considered meaningful or not [**45**]. Different studies argue that End-to-End learning approaches still cannot be fully trusted to be used in practice [**4, 45**]. A recent study shows that during interaction experiences with End-to-End CRS, at least one-third of the system responses were not meaningful for the annotators [**45**].

- **Data Flow architectures:** In this architecture, the dialogue state is defined as a data flow graph. These types of architectures rely heavily on using graph approaches as a searching mechanism to reach the convenient recommendation. Despite the significant improvements that graph learning approaches bring to the recommendation field, these types of techniques are still incapable of providing natural human-like conversation. Most graph-based CRS focus only on retrieving the convenient item for the user and adding it to a pre-defined response [**46, 47**]. Also, such types of CRS approaches use a predefined conversation structure and rely on manually engineered knowledge, which can influence the user experience [**48, 45**].

- **Modularized architectures:** As discussed in [**4**], most recent existing CRS projects are based on this type of architecture, where the conversational agent could be formulated as different functional modules. These function modules can be divided into four major components: Natural Language understanding/generation, Dialogue state management, Recommendation, and Explanation modules (the following sections 2.2.1, 2.2.3, 2.2.4, and 2.2.5 discuss these modules in detail). Figure 2.3 highlights the general architecture of CRS, which is discussed by Dietmar Jannach *et al.* in their recent 2022 conversational recommender systems survey. [**4**]

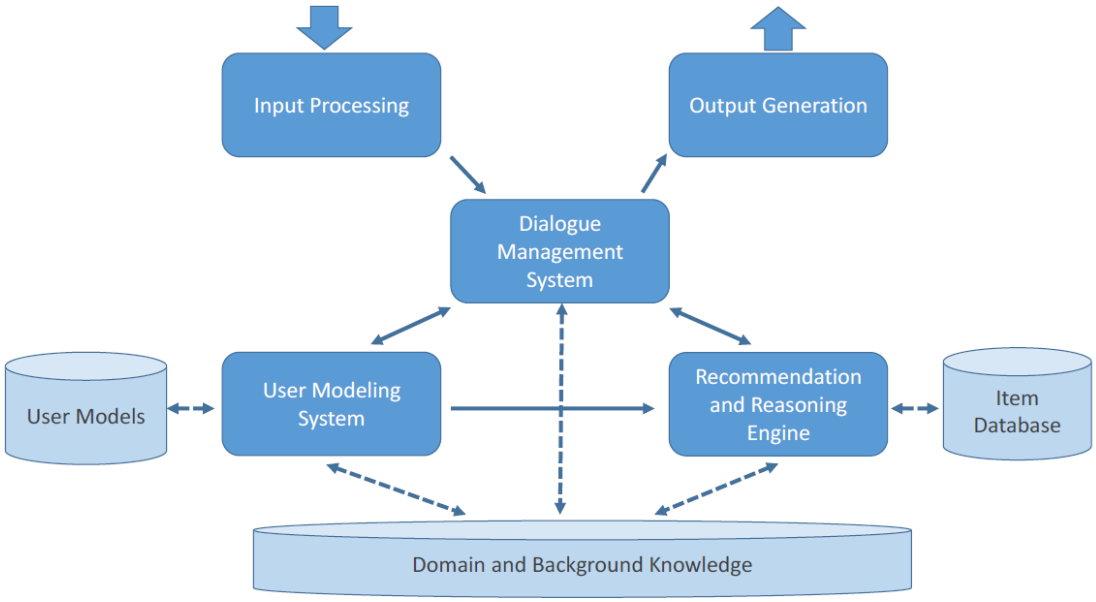

Fig. 2.3. Modularized CRS components [**4**].

The first conversational recommendation system paper, called "Adaptive Place Advisor", was published 20 years ago by Goker and Thompson [**49**]. The architecture design of the CRS in this paper is very similar to modern conversational recommendation systems [**4**]. Figure 2.4 highlights the different components of the Adaptive Place Advisor system and their interactions.

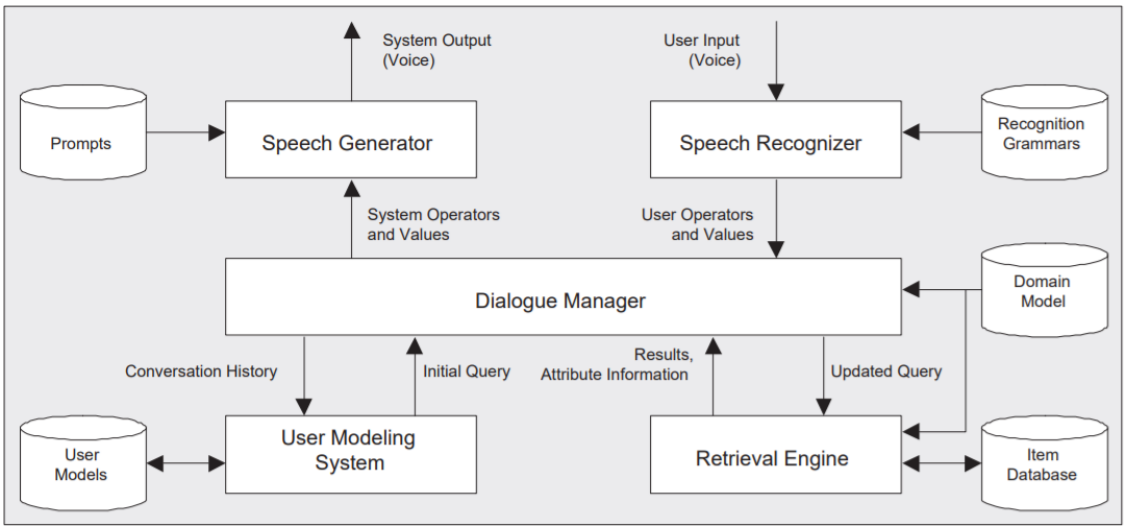

Fig. 2.4. Components of the Adaptive Place Advisor and their interactions [**49**].

In recent years the number of published papers on CRS has increased significantly. A new research direction has been presented in the literature which combines the use of machine

learning and deep learning approaches within the CRS modules. The next sections discuss research related to each CRS component.

2.2.1. NLU-Related Work

Natural Language Understanding is the process of gathering important knowledge and filtering meaningful information from user inputs. NLU uses different approaches and techniques to change natural human language into structured reasoning. To facilitate and improve the understanding of natural human language, different traditional NLU concepts and tools have been proposed in the literature, such as *Tokenization*, *Lemmatization*, *Stemming*, *Part-of-Speech Tagging*, *Word Vectorization*, etc. With the recent advancements in deep learning and attention-based architectures, which reduce the use of some traditional NLU techniques, it is important to discuss these traditional methods, as some of them are still used in modern NLP architectures, particularly the tokenization technique which is still used by almost every NLP model.

- **Tokenization:**

  Tokenization is the process of splitting a given text into words or sub-words which are usually called tokens. Different types of tokenization algorithms can be applied depending on the task's final goal, such as *word-based*, *character-based*, and *sub-words-based* tokenization.
  Word-based tokenization focuses on splitting the text into words using a splitting element. Character-based tokenization splits the text into characters. While this tokenization approach is very simple and would efficiently reduce both memory and time complexity, using this technique makes it very hard for the model to learn meaningful input representations.
  Sub-words-based tokenization is a hybrid combination between word-level and character-level tokenization techniques. It is based on the idea that frequently used words should not be split into smaller sub-words, whereas rare words should be decomposed into meaningful sub-words. For example, the word "annoying" is a common word, therefore, it should not be split. However, the word "annoyingly" would need to be split into "annoying" and "ly". This will allow the model to have a reasonable vocabulary size while being able to learn meaningful context-independent representations, as in our example where the model will learn that the word "annoyingly" is formed using the word "annoying" but with slightly different meanings. In addition, this tokenization approach enables the model to process words that it has never seen before by decomposing them into known sub-words [**50**]. For example, if the word "centralization" is unknown by the model, the sub-word-based tokenizer

will split it into a known root token and a second sub-word that represents additional information for it. In our example, "central" is the root word and "ization" is the second sub-word. This splitting technique helps the model learn that the words with the same root token as "central", like "centralized" and "centralizing", are similar in meaning. This will also help the model learn that "centralization" and "modernization" are made up of different root words but have the same suffix and are used in the same syntactic situations. This tokenization technique is very popular in modern NLU deep learning techniques, especially for the transformer and attention-based models.

- **Word Vectorization and Embedding:**

Word Vectorization, also known as word vector representations, is one of the most popular NLP techniques in modern NLU approaches [**51**]. It represents the words within a text in the form of feature vectors [**52**]. Figure 2.5 highlights an example of the word vectorization process.

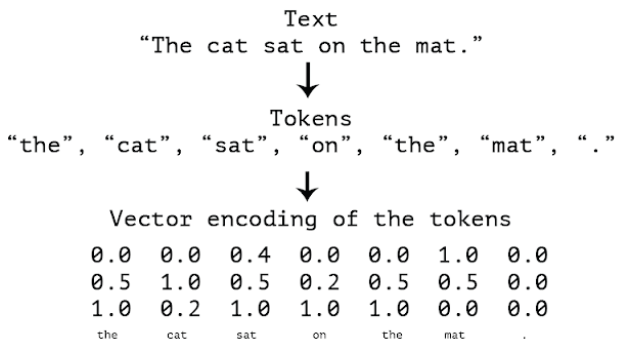

Fig. 2.5. Example of word vectorization process [**53**].

The values within the vector depend on the used approaches. Different approaches have been proposed in the literature to create vector representations. Traditional techniques such as the Term *Frequency-Inverse Document Frequency* (TF-IDF) use statistical methods to compute the vector values, which are represented by taking into consideration the frequency of a word within both the document and the whole corpus. Despite the effectiveness of some traditional representation approaches, these methods are incapable of understanding the real context within a sentence. That's why new approaches have been proposed in the past few years. Deep learning-based representation techniques such as *Word2Vec* [**54**] and *Glove* [**55**] have been considered state-of-the-art models for word embedding in the past few years. However, such techniques are not capable of providing different representations for the same

word when it is employed in different contexts. Therefore, recent work suggest employing attention-based and transformer approaches to understand the contextual information. Unlike Glove and Word2Vec's approaches, attention-based techniques are capable of learning more convenient vector representations that encode a variety contextual information. Given the context, attention-based models can provide different representations for the same word. The next section discusses these approaches and highlights the recent state-of-the-art transformer-based models for the NLU task.

2.2.2. Transformers for Natural Language Understanding

Transformer models have been dominating word embedding and contextual information understanding for the last five years. Different papers have been proposed in the literature since the "Attention Is All You Need" paper has been published in 2017 [**38**]. The BERT model [**7**] was one of the first transformer models to outperform many state-of-the-art approaches in different NLP tasks such as QA (Question-Answering), text classification, sentimental analysis, etc. The idea of using very complex decoder component layers that give attention to the meaningful information within a sentence led to the creation of other transformer-based models such as *ALBERT* [**56**], *RoBERTa* [**57**], *DistilBERT* [**58**], *XLNET* [**59**] etc. Figure 2.6 highlights the general *Encoder-Decoder* transformer architecture.

All of the pre-trained transformer models are based on the Encoder-Decoder architecture. The only difference is how they implement this mechanism to encourage greater information understanding. BERT uses *Masked Language Modeling* (MLM) and *Next Sentence Prediction* (NSP) objectives using the attention mechanism to obtain both left-to-right and right-to-left information. BERT is considered to be an Encoder-based model architecture, where once given a sentence, this model encodes the words within the sentence and passes the encoded information as input for the downstream task (for example, a classification task using the "[CLS]" token).

DistilBERT is a new and lighter BERT-based version, where the BERT model is compressed using the *Teacher-student* learning approach. In this model, a simple neural network is trained to imitate a complex model (BERT) performance with significantly fewer trainable parameters.

Unlike the DistilBERT model, which improves the efficiency aspects of the BERT model, the XLNET model aims to improve the effectiveness performance compared to BERT. Using a *two-stream* self-attention (*Query stream*, and *Content stream*), this model is capable of understanding both sides' hidden relationships between different words within a natural language sentence. Despite the shining results provided by most transformer-based models, most of them rely on large datasets and are computationally expensive.

Unlike BERT, which uses the MLM objective during the training phase, ELECTRA

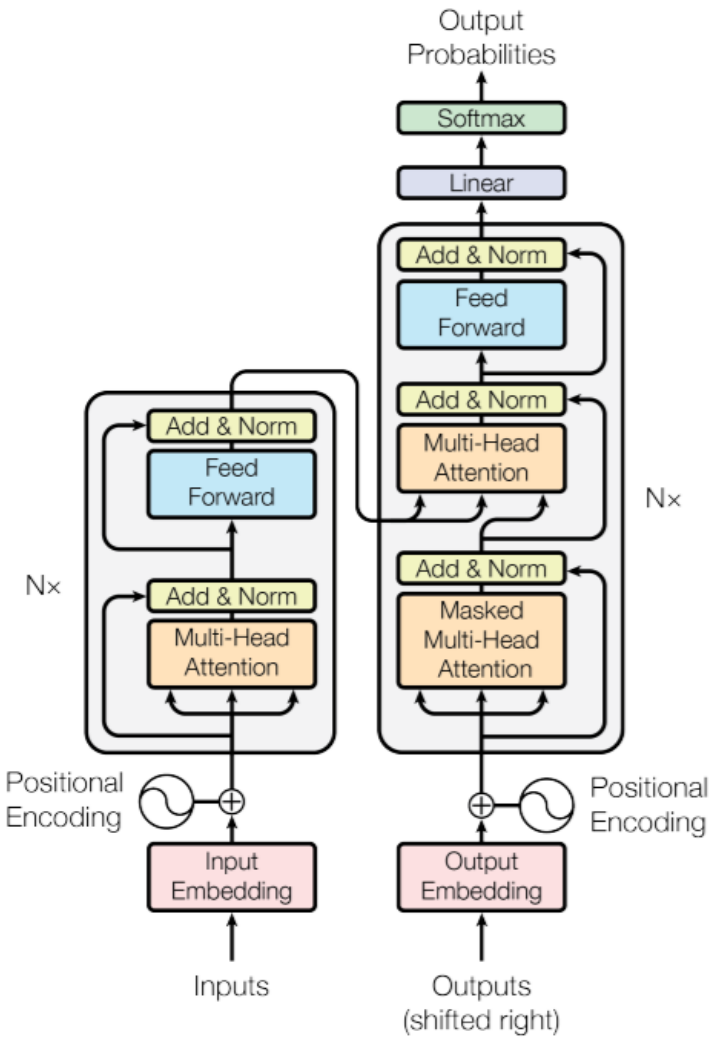

Fig. 2.6. The Encoder-Decoder transformer architecture [**38**].

[**43**] uses the replaced token detection mechanism. It combines both a *Generator* and a *Discriminator* component. The Generator is trained using an MLM approach to predict the masked token, and the Discriminator is trained to infer for each word in the sentence provided by the generator, whether it is the original word or if it has been replaced.
Therefore, instead of only seeing 15% masked words in the sentence and predicting the right tokens (in the case of the BERT model), this model will see all of the tokens. Using this technique, ELECTRA was capable of understanding better contextual insights in a very efficient way. Moreover, training the Discriminator with a binary classification loss to predict whether the word has been replaced or not helps the model reduce the computational expense. Clark *et al.* showed significant improvements in comparing ELECTRA with BERT, Roberta, and XLNET models on different NLP tasks. They argue that ELECTRA was capable of producing similar or better results with fewer computing resources compared

to other transformer models.

Now that we have discussed the most recent research on understanding the context within natural language sentences, which we will use for understanding the context of conversation utterances, the next section discusses the recent work published for the CRS Dialogue State Tracking (DST) component.

2.2.3. DST-Related Work

Human natural conversation scenarios always follow specific *topic changes* to reach the main goal of the dialogue. Goal-oriented multi-turn dialogue systems try to simulate natural human conversation behaviour by following specific topic threads as the conversation progresses. These systems necessitate the ability to capture different topics within a dialogue and to generate topic-aware responses. This task is known as *Dialogue State/Topic Tracking* (DST or DTP). Throughout this section, we consider both conversation topic tracking and dialogue state tracking to be the same thing. In our task setup (explained in chapter 6.2), we argue that the state of the conversation is defined by the different topics mentioned in it. Therefore, predicting the current topic of the dialogue is equivalent to tracking the state of the conversation.

DST has become one of the important steps to building coherent and engaging conversation systems that simulate natural human dialogues. The first topic detection technique was proposed in 1996 by the *US Government Defence Advanced Research Projects Agency* (DARPA) [**60**]. Since then, topic detection has been applied in many different contexts, such as helping political groups to predict election results [**61**], discovering natural disasters [**62, 63**], and helping companies to understand users' preferences, needs, and behaviours and thus improve their marketing strategy. In recent years, the topic detection task has become an essential step for building coherent and naturally engaging dialogue and conversation recommendation systems. In their recent paper, Zhang *et al.* [**64**] studied the topic-aware dialogue agent in counseling conversations, where the agent led the dialogue topic by deciding between moving to a new target or highlighting a situation within the current range. To keep track of the current topic and dialogue state, Tuan Lai *et al.* proposed a simple but effective BERT model for dialogue state tracking [**65**]. By formulating the dialogue state prediction task as the BERT NSP task, the authors managed to effectively track the current dialogue state within the current utterance. To make the prediction task efficient, the authors used a distillation technique to reduce the complexity of BERT by training a student network architecture to simulate the performance of a teacher network (BERT). Moreover, Hongru Wang *et al.* [**9**] used a three-stage iteration mechanism for the

topic-aware response generation task. First, they generated a response while conditioning the dialogue history utterances. Second, they predicted the conversation topic using a topic prediction component. Finally, they used the predicted topic to refine the first generated response and produce a more coherent one. For the topic prediction component, the authors of this paper explored both *GPT-2* [**6**] and BERT [**7**] pre-trained models, finding that BERT performed better than GPT-2. In Kun Zhou *et al.*'s recent study titled "Towards Topic-Guided Conversational Recommender Systems" [**8**], they employed a three-stage BERT encoder model combined with an MLP network to predict the dialogue topic. To do this they mainly used text data type and implemented a *conversation-BERT*, *topic-BERT*, and *profile-BERT* encoder models for encoding the historical utterances, historical topic sequence and user profile, respectively. They then used an MLP network with a softmax layer to predict the current topic.
Despite the good performance provided by both proposed approaches, these techniques are very computationally expensive as they use different transformer models (three BERT models, or two BERT models with a GPT-2 model) to encode different information.

Being inspired by all of the previous research and by the significant performance improvements that pre-trained and transformer models provide for understanding context, as well as by the significant computation reduction provided by the ELECTRA transformer model for the classification task, we investigate in this thesis the impact of using the ELECTRA transformer model on the topic prediction task [**43**]. Unlike the work of Tuan Manh Lai *et al.*, where they train a new model to simulate the performance of BERT, we aim to use the binary classification objective function of the Electra Discriminator component and fine-tune this pre-trained model to predict the utterance topic in an efficient way. Chapter 6 highlights our proposed architecture for the CRS Dialogue State Tracking component using ELECTRA, as well as its performance evaluation.
The next section highlights the Recommendation Engine (RE) component-related work.

2.2.4. The Recommendation Engine-Related Work

Characterizing users' preferences, attentions, desires, and interests accurately is a very important aspect in building an effective personalized recommendation system.
Sequential recommendations usually capture sequential insights using individuals' historical purchases and interactions. Previous research such as Shani *et al.*'s work [**66**] used the Markov chain and decision process to formulate the recommendation process as an optimization task. As a continuation of this work, Rendle *et al.* [**67**] used both Markov chain and matrix factorization approaches to model both individuals' general interest and their sequential behaviour. Moreover, different deep learning approaches were proposed to tackle the

sequential recommendation problem. *Recurrent neural network* (RNN) variation approaches (LSTM [**68**], and *Gated Recurrent Units* (GRU) [**69**]) were the first deep learning models to have been applied for user behaviour modeling tasks [**70, 71, 72, 73, 74, 75, 76**]. These approaches aim to model the user's behaviour into a single hidden representation, which is then used for a recommendation goal. A variant of these techniques is the Gru4Rec model proposed by Balázs Hidasi *et al.* [**73**], which uses a ranking loss with a window mechanism to predict the top next item to be recommended. Despite the architecture design of the RNN-based models, which naturally supports the sequential order, researchers never hesitated to explore and propose different types of deep learning architectures for the sequence order modeling task rather than using only the RNN variants, which may imply a serious problem, as the final prediction will only depend on the last network encoder node representation [**77, 78, 79, 80**]. Wang *et al.* proposed a convolution neural network architecture for sequence insight modeling. Using filters and convolution layers, the authors managed to learn sequential patterns.

Moreover, some research proposes employing an attention mechanism for modeling sequential information [**79**]. In their "Bert4Rec" framework, Fei Sun *et al.* [**81**] used a BERT [**7**] transformer architecture to model the sequential order of users' interaction in order to predict the next item that an individual may interact with. The authors applied the MLM learning approach to the historical interactions of each user to encourage BERT to learn the relation between the different interactions. During the inference step, they simply concatenated the user's previous interaction with a [MASK] token at the end of the sequence. BERT then tries to predict the next item to interact with by predicting the [MASK] token. Using this approach, the authors showed that their proposed model outperformed recent state-of-the-art models for the sequential recommendation task on different datasets. Despite the effectiveness of this model, this technique is very expensive as the authors trained BERT from scratch. Also, this approach only uses textual data and is incapable of incorporating numerical and tabular shape information. Moreover, it does not employ personalization aspects. There is no implementation of users' specific information during the prediction task. In addition, using the random masking approach, the model is capable of learning the relationship between the different movies, but it is not guaranteed that the model will learn the sequential natural structure of the movie-watching sequences.

We argue that knowing both previous and following items during the prediction process will limit the model's capability to learn the sequential structure of the watched movies. For example, let's say that we have four movies (@Movie1, @Movie2, @Movie3, and @Movie4). Suppose that we mask @Movie1 and @Movie4 and try to predict the correct masked token ([MASK], @Movie2, @Movie3, and [MASK]). @Movie1 and @Movie4 are strongly related to each other in a way where watching @Movie4 is only suggested if we watched @Movie1. In this case, using only @Movie2 and @Movie3 to predict @Movie4 and @Movie1, there is

no way for the model to predict the correct movie, @Movie4, because @Movie1 is masked. Therefore, the standard MLM objective used by Bert4Rec is limited in terms of understanding the natural sequential structure between items.
To overcome this problem we propose to use a transfer learning trick to make the recommendation model capable of understanding more relative information. Chapter 5 highlights in detail our proposed architecture as well as its performance and evaluation setup. The next section highlights the NLG component-related research.

2.2.5. NLG-Related Work

Natural language generation is an artificial intelligence process that transforms different types of data into languages that are coherent and understandable by humans. This is done using statistical approaches that analyze a large amount of data and use it to produce natural human sentences. The analyzed data can be in different shapes (textual, tabular, images, etc.). The application of NLG has significantly improved in the past few years to reach different tasks such as storytelling [**82**], machine translation [**83**], poetry generation [**84**], text summarizing, dialogue systems [**85, 86, 87**], dialogue-based recommendation systems [**8, 9**], etc.
The first *dialogue-based* NLG system is called Eliza [**88**] and was published in 1966. Eliza generates language through a set of rules. The use of explicit and predefined rules made the Eliza system incapable of generalizing on unseen data and of producing diverse responses. Since then, different approaches have been proposed in the literature to enhance NLG generalization capabilities such as language modeling approaches.

Language models are probabilistic models that are capable of predicting the most likely word given the preceding words [**89**]. N-grams are one of the most popular statistical approaches for language modeling [**90**]. Mainly relying on the Markov assumption (a word is only dependent on its previous n-words), this technique uses a chunk of consecutive n-words to predict the probability of each word in the vocabulary being the next word given the chunk of preceding tokens. Despite the promising performance of this approach in different scenarios, this technique is incapable of considering the whole context within a sentence as it relies on a fixed size of previous words. Also, this technique is computationally expensive, especially for large n-values, as it needs to store all of the previous n-grams in the memory for each step.
In 2010, Mikolov *et al.* [**91**] argued that using statistical approaches to model languages has limited capabilities and cannot lead to significant performance results. Using RNNs, the authors achieved state-of-the-art performance for the NLG task. These results were the main motivation for neural network approaches to become more popular for

modeling sequential data like text [**89**]. Moreover, with the significant improvements in sequence-to-sequence and auto-regressive neural network architectures, as well as major advancements in computational resources and large corpora and data availability, neural networks became the first choice for most NLG researchers. The capability of these networks to learn representation with different levels of abstractions made them capable of achieving state-of-the-art generalization results [**92, 93**].
Bengio *et al.* [**94**] were the first to investigate the potential of sequential neural networks on language modeling tasks. In their proposed approach, they relied on a fixed window size of words to predict the next probable word. Despite the promising results of this revolutionary and novel language model approach "at the time", this network was incapable of encoding a high level of information as it only depends on a pre-defined number of words. To overcome this issue, new types of sequence-to-sequence architectures were proposed in the literature such as RNN-based models [**91, 68, 95**]. Unlike the previous approaches, these techniques are capable of encoding a larger size of inputs. Using the gating mechanism, these approaches can encode long sequence contexts which significantly improved the language generation performance. However, these approaches are one-directional generation approaches, meaning that they can only encode data from one side, either from left to right or from right to left. Although this technique can effectively generate human-like language, sometimes it cannot produce meaningful or grammatically correct sentences, as in some cases, generating the correct words needs an overview of both previous and following words.

To overcome this issue, new attention-based language modeling approaches were proposed in the literature. In their recent paper [**8**], Kun Zhou *et al.* used a pre-trained GPT-2 model to generate topic-guided responses in the movie domain. The authors considered two different scenarios for dialogue generation. The first is for non-recommendation scenarios, where they trained the GPT-2 model to generate response conditioning on the historical utterances and predicted topics. To accomplish this, they just concatenated both types of information into a single input using the token [SEP] as a separator. The second is for recommendation scenarios, which consists of generating response by conditioning on the recommended movie. They concatenated the recommended movie to the historical utterances to form a new input for the GPT-2 decoder component.
Unlike the work by Kun Zhou *et al.* [**8**], which uses two different models, one for the topic prediction and one for the dialogue generation, Hongru Wang *et al.* [**9**] proposed a new End-to-End network to achieve both topic prediction and response generation simultaneously. Using two different GPT-2 models for dialogue generation and a BERT model for the topic prediction task, the authors proposed a refined learning approach for topic-aware response generation. First, a GPT-2 model generates a response using only the previous utterances (without any knowledge of the different topics within the utterances).

Then, a BERT model predicts the current topic of the previous utterance. Using the prediction of the BERT model, a second GPT-2 model generates a new response. However, unlike the first GPT-2 generation model, this model generates a response while taking into consideration both the topic predicted by BERT and the first GPT-2's generated response. Using this approach, the first response is refined using the predicted topic to reach a new topic-aware generated response.
Despite the improvement of both topic prediction and oriented response generation models' performance on the TG-Redial dataset [**8**], this approach lacks the integration of the recommendation component. The main purpose of this architecture is to generate a topic-oriented response without focusing on the recommendation task. Also, in order to control the response generation process, the focus is only on using the topic information as control features, which is not sufficient to generate personalized utterances, as personalizing a response may depend on different types of features such as the user's preferences, their personalities etc. Moreover, we argue that fine-tuning three large pre-trained models is very computationally expensive. Therefore, efficient LM-based dialogue generation paradigms were needed.

In the past few months, a large amount of controllable text generation research has been published in the literature. Different works are investigating the use of *prompt-based* learning to control and encourage pre-trained language models to generate texts more efficiently. Figure 2.7 highlights the significant spike of published work about the prompt-based learning paradigm in the last two years.

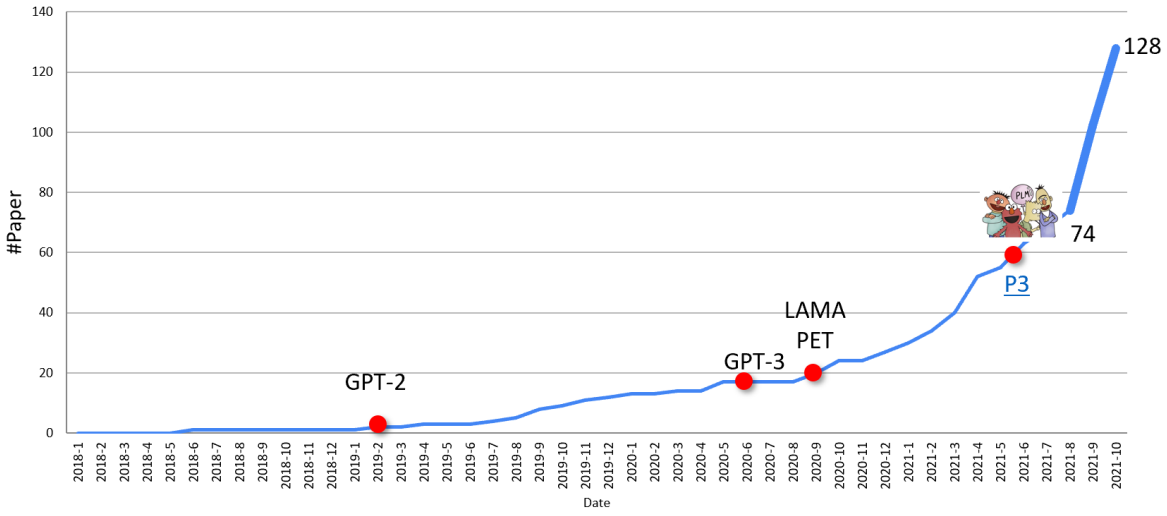

Fig. 2.7. The trend of prompt-based Research [**96**].

We can see that the first prompt-based learning approach was published in 2018, and that for the following two years, this topic did not get remarkable attention in the research

field. However, in the past few months, it has become extremely popular and an enormous amount of work has been published regarding it. Therefore, in the next sections, we discuss the controllable text generation task in general and highlight the prompt-based learning paradigm specifically.

2.2.6. Controllable Text Generation

Controllable text generation is the process of guiding a decoder model to generate personalized responses while taking into consideration different features and constraints. The constraints used to control the response generation can be in different shapes. For example, an utterance topic can be considered to be a constraint. The user model, such as its preferences and profile, can also be considered as a constraint to generate more specific and personalized responses. Also, the user's specific attributes, such as their demographic information, can significantly improve the personalization of the generated response. Combining the controllable generation approach with state-of-the-art pre-trained models has achieved state-of-the-art results on different benchmark datasets.
In 2020, Sumanth Dathathri *et al.* proposed a new language model approach called "*Plug and Play Language Models: A Simple Approach to Controlled Text Generation*" (PPLM) [**97**], which uses a pre-trained language model with the combination of an attribute classifier that guides text generation. Moreover, Eric Smith *et al.* [**98**] used 200 different text style control codes to encourage the language model to generate specific style-based responses. Behnam Hedayatnia *et al.* [**99**] proposed a new policy-based response generation model, which first generates a response policy, and then uses it to ensure a more controllable response.
Different control attributes have been used in the past few years to control the response generation process. Recently, a new study published at the ACL 2022 conference, "*Psych-E*: Configurable Response Generation using Personality Traits and Pragmatics" (which has been published anonymously and is still under review), tackles the controllable response generation task using individuals' Big Five personality traits. Using both Big Five personality traits and current utterance intent/topic as control codes, the authors provide better policy modeling for the response generator model, which results in a set of configurable parameters that can be modified to generate diverse personal responses using an encoder-decoder architecture. The encoder was responsible for encoding the planning step (the different contexts, policies, and facts within the conversation). Three different encoders were proposed to encode the facts, policy, and context within a conversation. Then, using the encoded information, a planning step was employed. Within the planning step, different classifiers were used to predict individuals' personality traits and the intent control codes, as well as to select the relevant facts within the conversation. Conditioned

on the encoder output information (the facts within a conversation, the response policy, the user's personality traits and their intents), a decoder component is responsible for generating a more personalized response utterance. The authors of this paper trained the entire system as an End-to-End architecture to try to minimize the weighted sum of the binary cross-entropy loss for the intent prediction, the binary cross-entropy loss for the fact selection, the language modeling cross-entropy loss, and the trait prediction cross-entropy loss. For the experimental setup, the authors used both *BART* [**100**] and *BlenderBot* [**101**] pre-trained models for the classification and response generation tasks respectively.

Despite the good performance provided by this approach, this model is only trained for dialogue purposes, not for recommendation goals. Moreover, this architecture can be categorized within a *multi-objective* learning paradigm approach, where we have different loss functions and where the model tries to perform effectively on all of the tasks. The use of different objectives within the same model can be a drawback for these types of architectures, as it significantly increases the model's complexity by having more parameters for each objective loss.
To overcome the objective learning paradigm drawbacks, a new learning paradigm named *prompt-based learning* was proposed in the literature. It is important to highlight that the work proposed in the above paper is very similar to our work in terms of using pre-trained language models and individuals' personalities in personalizing the response generation process. However, our work differs in terms of how we employed the generation control factors, as we have focused on using the prompt-learning paradigm to efficiently produce more personalized responses, compared to the above papers' work, which uses the objective learning paradigm.

In the next section, we discuss the core idea behind prompt-based learning and highlight recent related literature that employs this new approach for the recommendation process.

2.2.7. Prompt-Based Learning for NLG

Prompt-based learning is a *self-supervised* learning paradigm where a learned or a manually crafted *template* is added to create a new representation for the original input text and prompt the knowledge of a pre-trained LM. Given the new text form, the main objective of the language model became to model the probability of the input text $P(x)$ directly, unlike traditional learning paradigms (supervised/semi-supervised), where pre-trained models take x as input and predict an output y as $P(y|x)$.
By only designing convenient prompts, one pre-trained model with its pre-acquired knowledge can be used for different tasks without any implementation of other networks on

top of it. There is a major difference between previous paradigms such as *feature-based*, *architecture-based* and *objective-based* learning approaches and this new learning paradigm. Instead of adapting and designing different model architectures and objectives to fit the existing data, the prompt-based learning paradigm adapts the data to fit an existing model. More details about the prompt-learning working mechanism are defined in appendix section A.1.6.

Prompt-based learning has been successfully applied to many applications, such as *image captioning* [**102**]or *text summarizing* [**103**] etc. Wenpeng Yin *et al.* [**104**] used a manually designed prompt to predict the topic of a given document as a prompt-based *classification task*. They designed the prompting template as "the topic of this document is [Z]". This prompt was then fed to a pre-trained masked language modeling technique to fill in the slots. Yulong Chen *et al.* [**105**] used a prompt-based learning approach to *extract word relation* information from the text by modifying the entity mentioned in the template using a special marker, "[E]". The job of the model was to predict the masked entity. Timo Schick *et al.* [**106**] proposed the use of prompting approaches for *dataset construction* and *data augmentation* tasks. For example, to construct a dataset containing pairs of sentences that are semantically similar, the authors used the template "Write two sentences that mean the same thing. [X][Z]". Using this template, the model learned how to generate sentences that share the same meaning as the input text. Moreover, for the *text generation* task, different approaches were proposed in the literature that used prompt engineering to generate controllable responses. Alec Radford *et al.* [**6**] used the prompt template "translate to French, [X], [Z]" to control the pre-trained model and encourage it to translate English text to French.

Following the work of Radford *et al.*, Lei Li *et al.* [**107**] proposed a prompt-based learning model named *Personalized Prompt Learning for Explainable Recommendation* (PEPLER) to generate explanation sentences for the recommendation systems. Using user and item IDs as prompts, the authors designed two training strategies to explain the recommended items. The first strategy is called sequential tuning. Within this strategy, the author separated the training process into two stages.

The first stage is responsible for fine-tuning the continuous prompts (ID vectors) where the parameters of the LM are frozen. This is done so that both the continuous prompt and LM representation can be accomplished in the same space. For the second stage, they updated both the model and prompt parameters depending on the performance of the pre-trained LM for generating a convenient explanation.

For the second learning strategy, the authors made use of the rating prediction task to increase the effectiveness of the explanation task. They argue that the explanation task is highly related to the recommendation one. Therefore, increasing the recommendation task

performance will significantly improve the explanation performance. To accomplish this, the authors used the users' rating information as additional information to the prompt during the explanation learning process.
Despite the promising recommendation performance provided by the PEPLER model, this approach focuses on the explanation task rather than the recommendation task. During the experiments and results, the authors only reported the explanation task performance and did not report the recommendation performance. Also, the prompts templates were designed to fulfill the explanation purpose and not the recommendation one.
Another recent study proposed by Damien *et al.* [**108**] also employed the prompt learning paradigm for the *recommendation task*. Using a manually designed prompt, the authors managed to formulate a zero-shot learning sequential movie recommendation task. By designing a prompt template as "$< m_1^{u1} > , < m_2^{u1} > , < m_3^{u1} > ... < m_i^{u1} >$", where $< m_i >$ defines the i^eth rated movie by the user u1 and by using the GPT-2 pre-trained model without any fine-tuning, the authors managed to outperform recent matrix factorization approaches on the MovieLens 1M dataset for a small number of users on the MAP@1 metric, which highlights the significant capabilities provided by the prompt-based learning paradigm.

To summarize, much research has taken place in the past few years to tackle the recommendation task problems. Some of them use the user's previous behaviour as ground truth to model user preferences. Another line of work proposes interaction-based systems to consider both previous and current user preferences and personalize the recommendation session. Recent work employs pre-trained LMs to prompt their pre-acquired knowledge and fulfill the recommendation task.
Table 2.1 presents recent recommendation systems that tend to be similar to our work, whether in terms of the used approach, the recommendation domain, the conversation, chitchat, prompting, and/or personality support.

It is important to mention that, to the limit of our knowledge, only recent one-shot recommendation paradigms have employed users' personality traits within the recommendation process [**5**]. We did not find any published CRS research that employs users' personality traits within the conversation recommendation session. Moreover, for the prompt-based recommendation system, we did not find any CRS work that uses this new paradigm to improve the recommendation results. The only CRS published work that we found using prompt learning is the one by Lei Li *et al.* [**107**]. However, the main objective of their work was using prompt learning to explain the recommendation and not for improving the recommendation itself.

Table 2.1. Summary of similar published work.

Paradigm	Approach	Task	Conversation	Chit-chat	Prompting	Personality	Year	Reference
One-shot	Collaborative Filtering	Movie recommendation	No	No	No	Yes	2015	[**11**]
One-shot	RNN	Job postings	No	No	No	No	2017	[**73**]
CRS	Graph model	Product recommendation	No	No	No	No	2018	[**46**]
CRS	Auto-Encoder RNN	Movie recommendation	Yes	Yes	No	No	2018	[**109**]
One-shot	BERT	Movie recommendation	No	No	No	No	2019	[**81**]
One-shot	Retrieval system	Travel recommendation	No	No	No	Yes	2020	[**12**]
CRS	Collaborative Content-based Filtering	E-commerce	Yes	No	No	No	2020	[**110**]
CRS	GRU/BERT Attention	Movie recommendation	Yes	Yes	No	No	2020	[**111**]
CRS	ConceptNet DebiatNet	Movie recommendation	Yes	Yes	No	No	2020	[**112**]
CRS	BERT GPT-2	Movie recommendation	Yes	Yes	No	No	2020	[**8**]
CRS	BERT GPT-2	Movie/Medicine recommendation	Yes	Yes	No	No	2021	[**9**]
One-shot	BERT	Movie recommendation	No	No	Yes	No	2021	[**108**]
CRS	GPT-2	Explainable recommendation	Yes	Yes	Yes	No	2022	[**107**]
CRS	DialoGPT DistilBERT ELECTRA	Movie recommendation	**Yes**	**Yes**	**Yes**	**Yes**	2022	Ours

Therefore, we present in this thesis **PPerMo**, the first Prompt-based and Personality-aware modularized CRS. The next chapter highlights the main architecture of our PPerMo CRS framework.

2.3. Conclusion

This chapter allowed us to clearly discuss the existing research that is related to our proposed approaches in this thesis, as well as to define our contribution in comparison to state-of-the-art architectures.
The next chapter discusses the global architecture of our proposed Prompt-based and Personality-aware Modularized CRS framework (PPerMo).

Chapter 3

The Proposed PPerMo Framework Architecture

This chapter discusses the general architecture of our **P**rompt-based and **Per**sonality-aware **Mo**dularized CRS framework (PPerMo). This chapter highlights the main contribution of our research, which consists of building a modularized, topic-guided and conversational movie recommendation system that considers the individual's personality traits in order to improve and personalize the recommendation process. This chapter is structured as follows: Section 1 consists of a global overview of our proposed architecture and a detailed discussion of its different modules. Section 2 presents the working pipeline of our solution.

3.1. The Modularized CRS Framework

In our research, we acknowledge the importance of modularized CRSs [**4**] (as shown in section 2.2) to design a module-based CRS framework by building different components independently, where each component is responsible for a given task (i.e., dialogue state management, recommendation, response generation, etc.). For the following chapters, we will explain in detail the design procedure and our contribution on each component independently. However, in this chapter, we focus on presenting the global overview of our proposed CRS framework as well as the dependency between its different modules.

3.1.1. The Global Overview

In a non-personality-aware CRS, a set of user dialogues is given to the system so that it can learn the natural human conversation behaviour.
Non-personality-based CRSs focus only on understanding the human dialogue structure and they generally consider all of the users as having similar personalities. Therefore, within a conversation session, they focus only on the dialogue context to distinguish a user from another and generate a convenient response and/or recommendation.

By considering that all users have similar personalities, the CRS may be unable to provide different users with specific and personal recommendations. For example, if Bob and Alice are looking to watch a movie to relieve stress, the CRS, not knowing the personalities of Bob and Alice, may recommend the same movie to both of them. However, being an open person that enjoys new experiences, Bob may prefer watching action movies, in contrast to Alice being an emotional person who usually prefers watching dramas to relieve stress. Therefore, users may prefer some movies over others depending on their personality traits. To overcome these issues, the CRS should consider individuals' personalities as features to build more personalized dialogue sessions. Taking into consideration that different users have different personalities improves the performance of the CRS and helps it provide specific recommendations for each user. For these reasons, we organized our thesis as a two-goal research project. For the *First goal*, we aim to **design a novel personality prediction module**, and for the *Second goal*, we focus on **developing a modularized conversational recommendation system**.

- **Personality Prediction Module:**

  This module is a text-based learning algorithm that uses users' social media posts and comments to predict their personality traits in a supervised way. Given the user's textual information (discussions, conversations, comments, etc.) our personality prediction model can infer the Big Five traits related to that user (more details about this module are discussed in chapter 4).

- **The Conversation Recommendation System:**
  Using the novel personality prediction approach discussed in the first goal, we focus on designing a modularized CRS framework and enhancing its performance during the recommendation process by employing the user's personality traits on each module.

To reach both goals, we created different components in our solution (the User Model, the Dialogue History module, the NLU, the Personality prediction model, the Dialogue State Manager, the Recommendation Engine, the Dialogue Generation model, etc.). Figure 3.1 highlights these components and discusses the interactions between them (our both main research goals' "main components" are highlighted in green).

Using the user conversation utterances, the personality prediction model predicts the user's personality trait factors to introduce more knowledge about the user model to the modularized CRS framework. The CRS then uses the information within the user model to

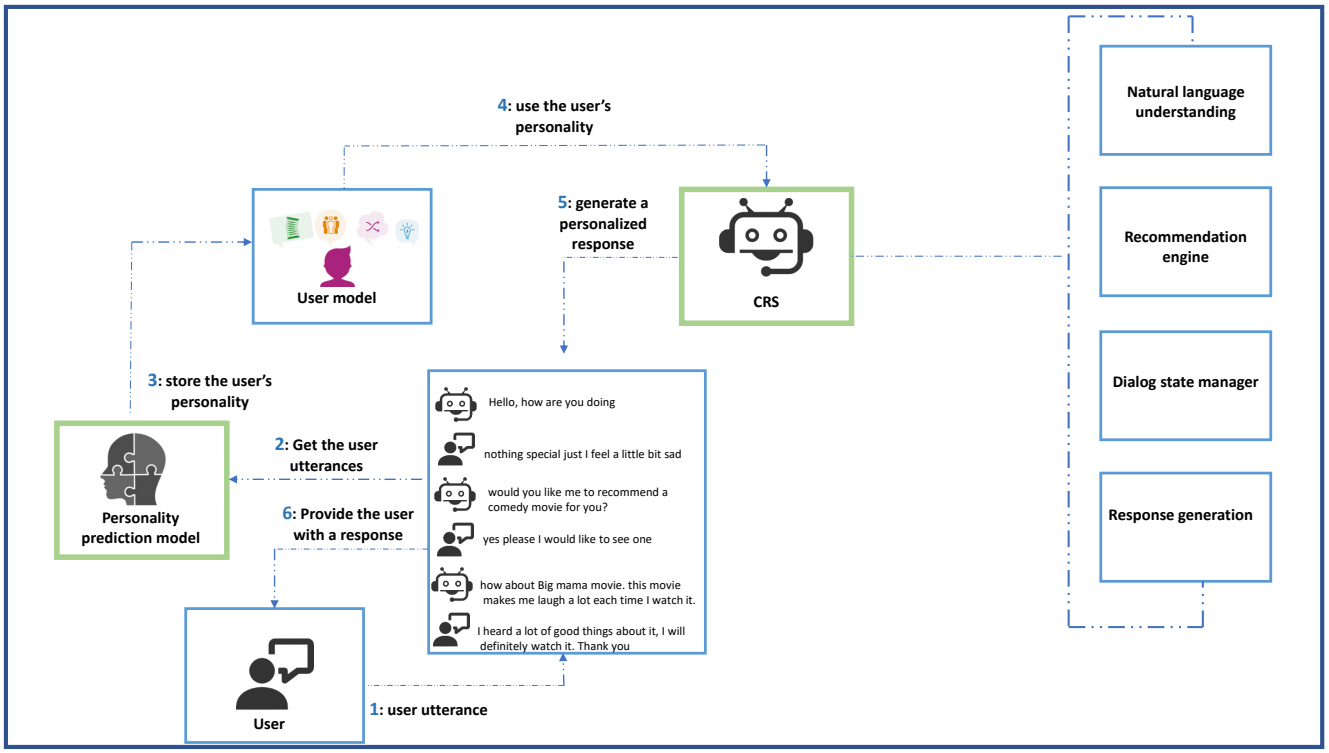

Fig. 3.1. General overview of our proposed system components.

create a coherent and personalized conversation recommendation session and finally provide the user with a convenient response.
After discussing the general overview of our thesis work, the following section highlights a detailed overview of each component within our modularized CRS framework.

3.1.2. The Objective of Each CRS Component

In order to build our personality-based CRS, we mainly rely on 8 components:

- **The User Model**:
  This component is responsible for storing users' specific related information, such as their *previous movie interaction history* (for example, watching action movies), *their personality traits* (for example, how much of an open or thinking person they are), *their preferences* (for example, if they like sports, adventures, spending time with friends, etc.). Figure 3.2 gives an example of the user model data structure.

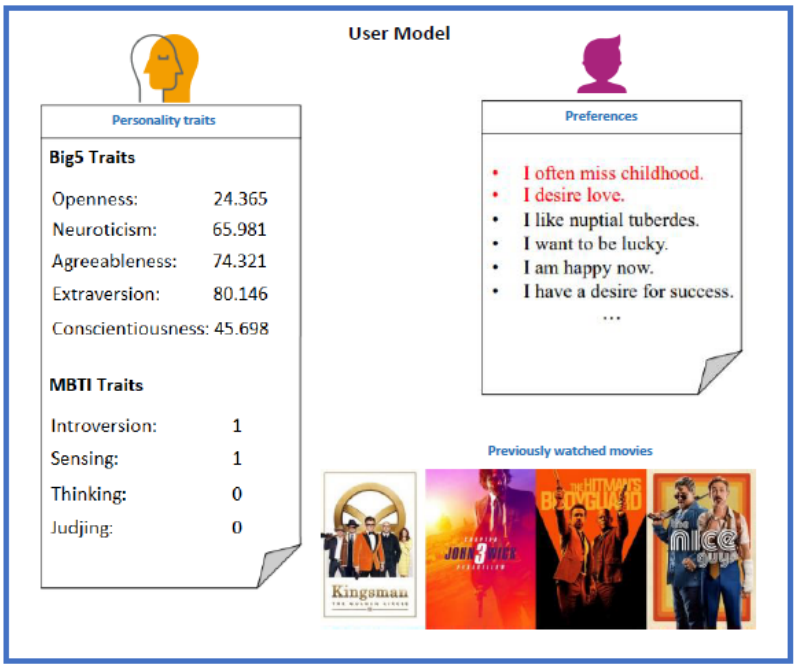

Fig. 3.2. The User Model example.

Tracking and storing the user model's information are the main keys to ensuring a personalized conversation and recommendation performance.

- **The Dialogue History Component**:
  Besides tracking the user model information, which helps to understand their *long-term* preferences, we also need to track and store their *short-term* preferences as well as their current emotions, which can be expressed within the conversation utterances. Figure 3.3 highlights an example of a user's dialogue history.

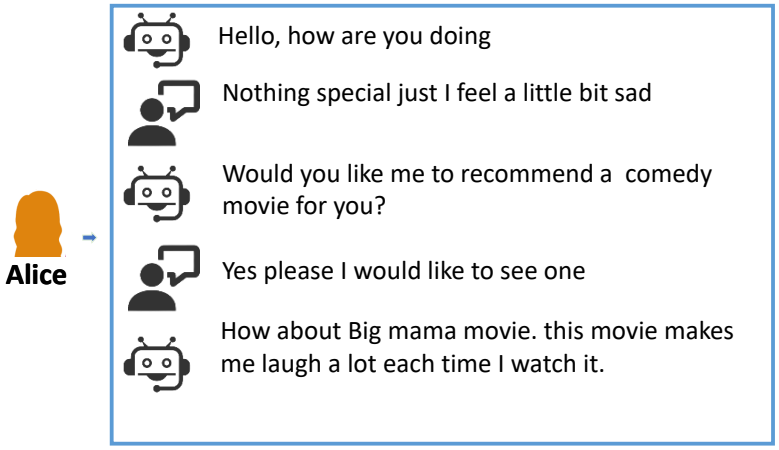

Fig. 3.3. An example of the Dialogue History component.

For example, if Alice expresses her emotions within a dialogue exchange by mentioning that she is not feeling well, the CRS needs to keep track of those expressions in order to build more natural and context-aware responses (for example, by suggesting a comedy movie). Knowing both long-term and short-term user preferences helps to insure more personalized recommendations and conversations. Therefore, in our system, both dialogue history and user model components are designed as data storage components that store *intra-based* and *inter-based* user-related information.

- **The NLU Component**:
  This component is responsible for understanding the contextual information within textual data input. For this component, we used the pre-built attention mechanism within the pre-trained transformer models. Therefore, we will not discuss this component in detail within the following chapters, as we just used pre-trained transformer models to understand the context within the conversation session.

- **The Personality Prediction Model**:
  As we have already mentioned above, one of the most important aspects of effectively modeling the users within the CRS is by using their personality traits. Figure 3.4 highlights an example of how users are distinguished from each other using the personality prediction system.

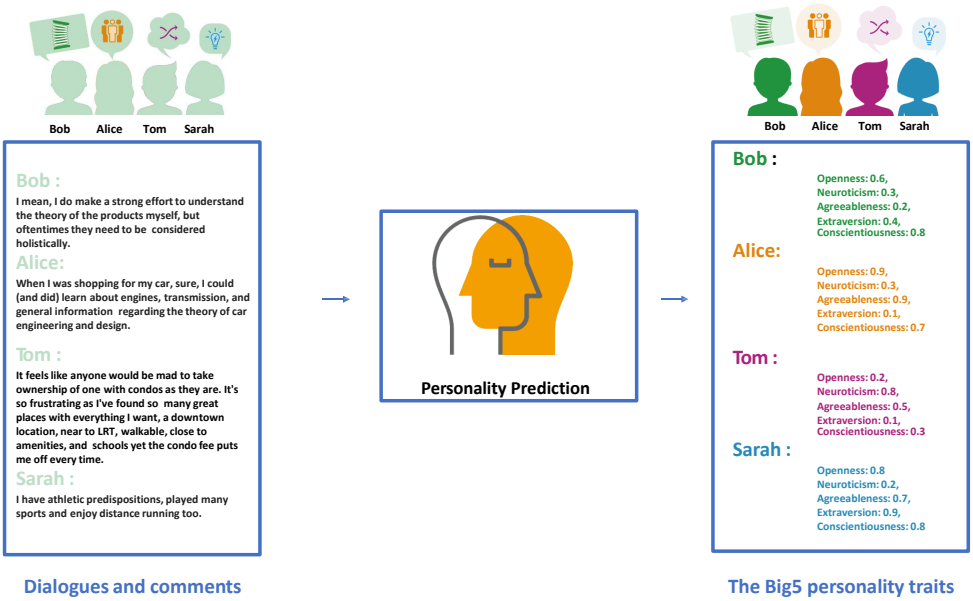

Fig. 3.4. The Personality Prediction Model example.

If Bob, Alice, Tom, and Sarah don't know what Big Five personality traits they have, it is important to design a predictive model that can predict their personalities using their conversation utterances and/or preferences. Therefore, given the users' dialogue, we differentiate them by predicting their personality traits to improve the recommendation session. This component is explained in section 3.2 and chapter 4 .

- **The Dialogue State Manager**:
  Given the previous and current utterances, as well as all of the topics discussed within the conversation session, this component is responsible for tracking the state of the dialogue by predicting the topic and intent of the current utterance. For example, within a dialogue session, Bob might talk about reading books, having coffee and relaxing. Therefore, the state tracking model may understand that Bob is looking for some activities to let go of stress. For that reason, it may suggest Bob to try certain sports. Figure 3.5 highlights a simple example of the input-output information for our dialogue state manager module, where reading books, having coffee and relaxing are the previously discussed topics.

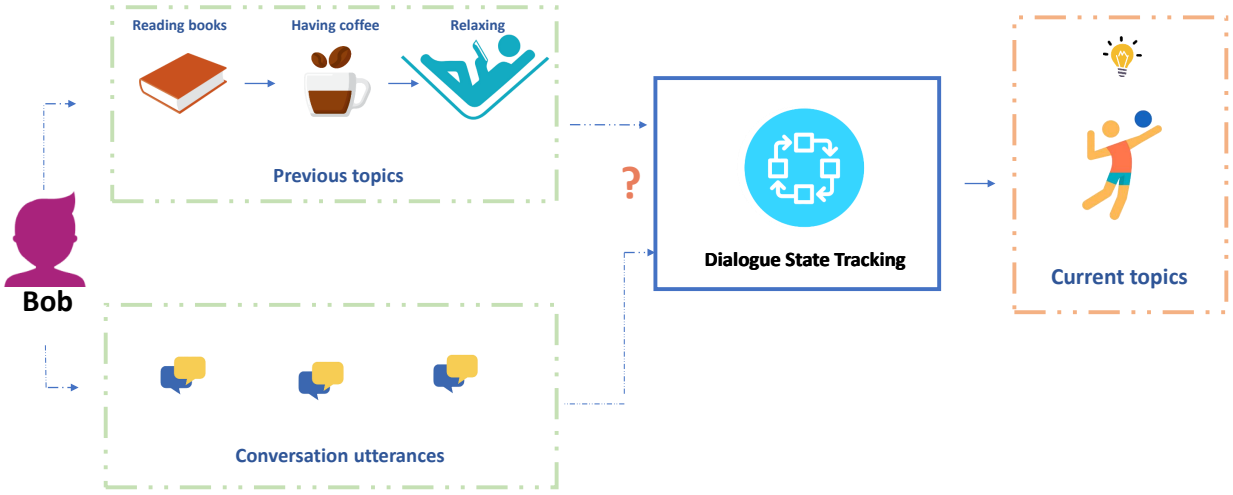

Fig. 3.5. The Dialogue State Manager model example.

By predicting both intent and topic information, this component can also decide whether to recommend a movie to users or to keep chatting with them under a

specific topic.

- **The Preference Generation Model**:
  This component is responsible for tracking the user's preferences. For example, within a conversation session, Bob may describe some new preferences that are not stored in his user model component (for example, enjoying the sunset, enjoying going out with his friends, etc.). Therefore, after the conversation ends and given all of Bob's previous utterances and preferences, this model generates the new preferences expressed by Bob (for example, enjoying having fun with his family) and stores them in his user model data storage. More detail about this component is explained in appendix section A.2.1.

- **The Sequential Movie Recommendation Model**:
  This component is responsible for tracking the user's previous watching history and recommending a movie depending on their long-term preferences. For example, if Bob previously watched "Iron Man", "Spider-Man", and "Hulk" movies, he is more likely to watch "The Avengers" as the next movie.
  It is important to highlight that we have two recommendation components in our global CRS architecture, one that recommends movies given the *long-term* dependency interactions, or "history of watching", which is the *Sequential Movie Recommendation* module and another one that recommends movies given the *short-term* dependency interactions based on the preferences expressed in the dialogue session, which is implemented in the *Dialogue Generation module*. Both long- and short-term recommendations are combined to predict the next movie for the user (more explanations are discussed in section 3.1.3). The next section highlights the Dialogue Generation module, or "NLG component".

- **The Dialogue Generation Model**:
  This component is responsible for generating a context-based and topic-aware response to the user given historical utterances, the conversation's current state (conversation topic) and the user model information. For example, knowing that Alice is a neuroticism person, in contrast to Bob being an open human being, helps our CRS framework to produce different and personalized responses. Figure 3.6 highlights the input-output information of our dialogue generation module.

  Despite the response generation role, this component is also capable of recommending movies within the generated response while taking into consideration the *short-term* preferences discussed within the conversation session. This is because the model

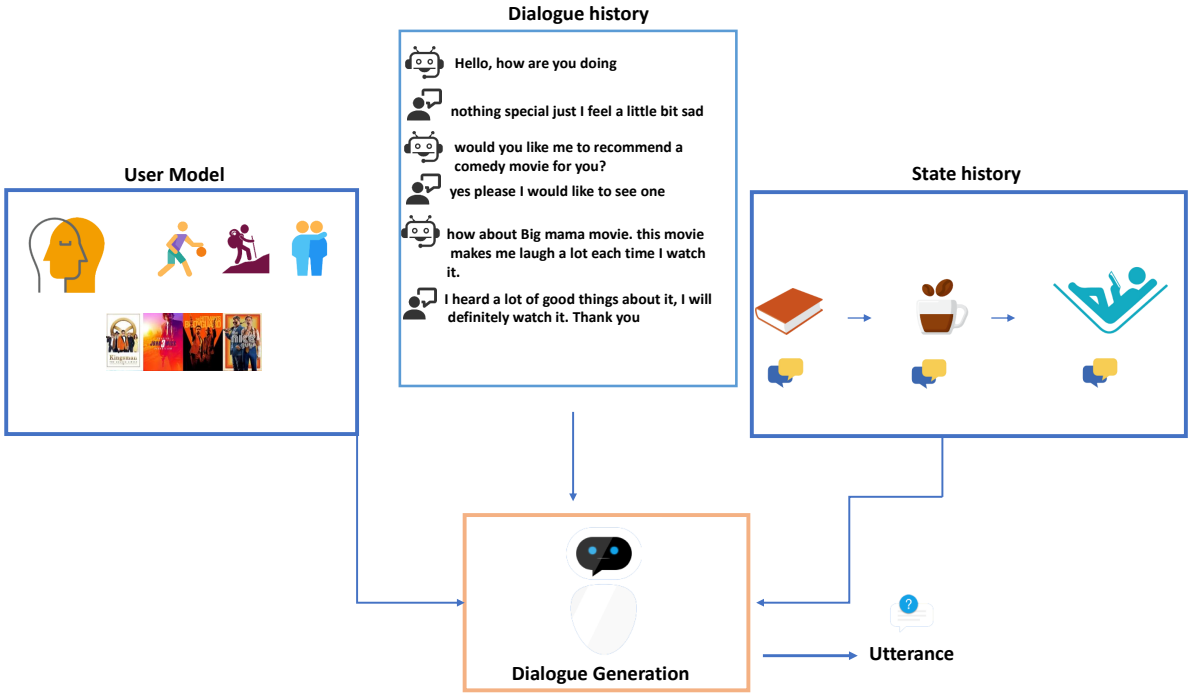

Fig. 3.6. The Dialogue Generation component.

was trained on a conversation corpus that already employs recommendations within the dialogue utterances. Thus, by generating words within the utterance, this component can also generate movies since it considers them as natural language words. More details are provided in chapter 7.

The reason behind designing two recommendation components (the sequential recommendation model which encodes long-term movie prediction and the NLG component which encodes short-term movie prediction) is to employ a movie prediction refining mechanism, which is discussed in the next section.

3.1.3. The Movie Prediction Refining Mechanism

Using the long-term-based movie prediction model, "sequential movie recommendation component 3.1.2", we aim to refine the short-term-based movie prediction and produce a final-shot item recommendation. Without the refining mechanism, only combining the long-term-preferences with previous long conversation utterances during the response generation process made the NLG component incapable of understanding important information and generate convenient responses, as the longer the input is, the harder for the model to understand the input's full context (more details are discussed in chapter 7).
Therefore, to benefit from the long-term movie preferences, instead of just combining them with the NLG input data, which might lead to poor recommendation performance, we used

a refining mechanism to produce more personalized movie recommendations. Figure 3.7 illustrates an example of the recommendation refining process.

Fig. 3.7. The movie prediction refining mechanism.

For example, if Alice mentions within the conversation session that she likes to watch comedies, the NLG component could generate a response that contains the "Big Mama" movie. Alice may refuse this recommendation and ask for a Vanessa Hudgens movie instead. Then, the NLG component may generate "The Princess Switch" (which is a Vanessa Hudgens comedy). This recommendation may satisfy Alice to some degree. However, if Alice's movie-watching history contains a lot of dancing and musical movies (for example, "Step Up 1", "Step Up 2" movies), then "High School Musical" (which is a Vanessa Hudgens comedic dance movie) may be a more convenient recommendation for Alice compared to "The Princess Switch". To predict the "High School Musical" suggestion, we managed to track the movies mentioned within the conversation session (highlighted in blue in figure 3.7), and when Alice is satisfied with a specific movie, we refined the last recommendation by proposing a new convenient one. To do this, we concatenated the movies mentioned within the conversation session with Alice's viewing history (highlighted in gray in figure 3.7) and used them as input for the sequential movie recommendation component to finally predict the refined recommendation ("High School Musical").

Now that we have explained the functionality of each component and how we make use of both long-term and short-term movie preferences to provide the user with a convenient

recommendation, in the next section we highlight the working pipeline of our proposed solution.

3.2. The Working Pipeline of Our Proposed Solution

As we have already mentioned in the previous section, our personality-based CRS framework contains 8 modules. Figure 3.8 highlights the relationship between these modules.

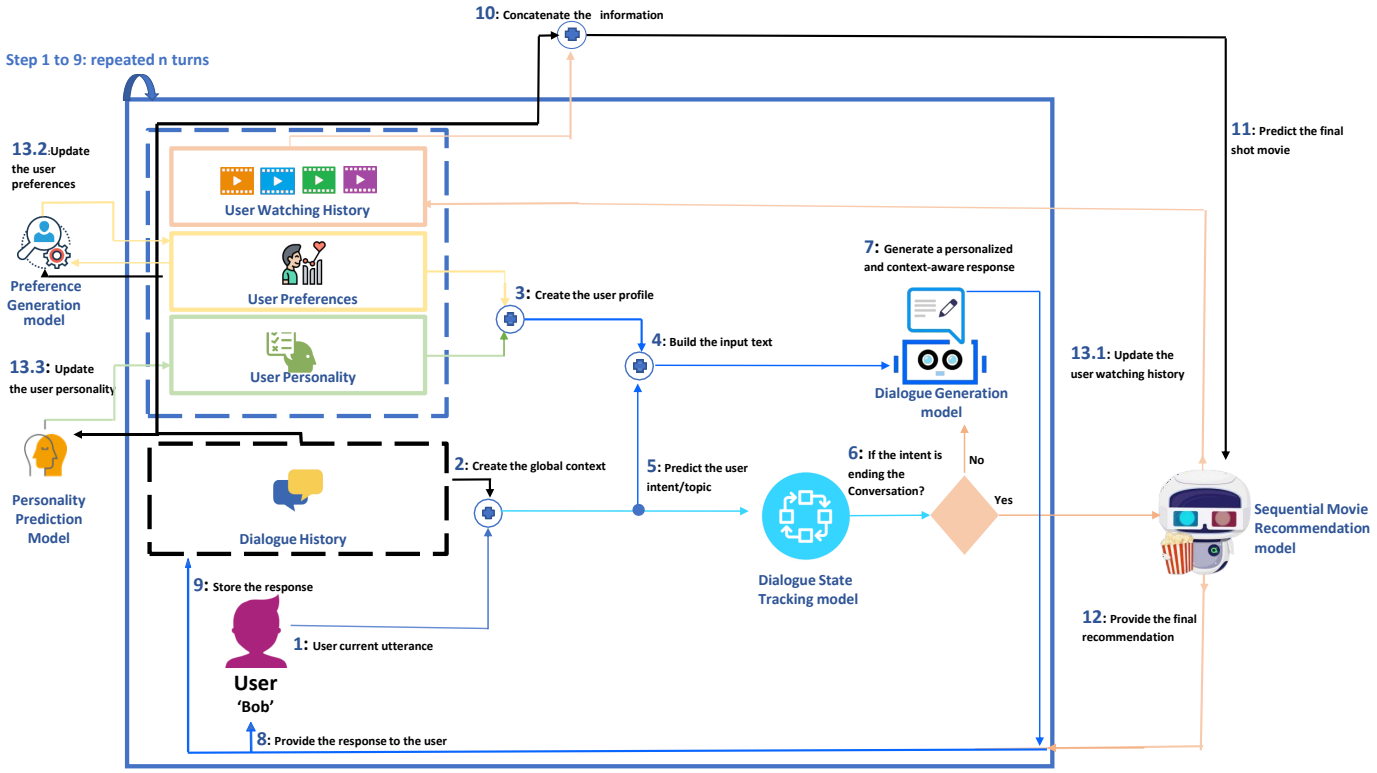

Fig. 3.8. Our proposed CRS architecture components.

To illustrate a CRS scenario, let us suppose that user Bob is currently in the middle of a conversation "at turn t" and that it is his turn to provide an utterance. The working mechanism of our proposed CRS can be divided into four levels: the Input Collection, the Dialogue Generation, the Movie Refining, and the Information Update phases. Therefore, a simple CRS scenario can be described as follows:

- **Input Collection:**

  - **Step 1:** First, the user provides an utterance U_t whether responding to the previous recommender utterance, expressing a new preference, etc.).

- **Step 2:** Our system then combines Bob's current utterance with the previous dialogue history to create a global context. This is done so that the generated response does not deviate from the general conversation context.

- **Step 3:** Moreover, our system takes into consideration Bob's personality and preferences to create his user model, which will be concatenated with the global conversation context in step 4. If Bob's preferences and personality are not found, also known as the "cold-start problem", the system will only consider the intra-information within the conversation to keep generating responses. After the conversation ends, it will build the user model and include the learned information.

- **Step 4:** To generate a more personalized response, the global context of the conversation and Bob's user model are combined into a single text to create the input information.

- **Step 5:** Given the global context information, our system tries to predict the current intent of the user (for example, whether he is looking for a recommendation, giving feedback, just talking, wants to stop the conversation, etc.). Besides intent detection, our system also predicts the current topic expressed by the user. For example, if Bob's intent is to talk we need to know whether he is talking about nature, friends, animals, etc. If Bob's intent is a recommendation, then the system needs to know what kind of movie it needs to recommend. This is done by executing the dialogue state tracking model.

- **Step 6:** After predicting the intent, we need to verify whether the user wants to end the conversation or not. If the user tends to end the conversation, then we jump directly to step 10, otherwise we jump to step 7. Let's assume for now that Bob wants to talk about books. Therefore, the intent predicted in step 5 is "talking" and the topic is "books".

• **Dialogue Generation**

- **Step 7:** If the user's intent is not to end the conversation, then it is up to the response generation model or "NLG component" to generate a personalized and intent-aware response. This is done by prompting the knowledge of a pre-trained dialogue generation model using hand-crafted templates. To not deviate from

the conversation's context, the NLG component considers all of the accumulated information (information of step 4 and the intent/topic information of step 5) during the response generation process.
Because Bob's intent provided from step 5 is not one of recommendation, the NLG component will not generate a response that contains a movie suggestion (this was ensured using the prompting templates explained in chapter 7, where the NLG component will suggest a movie only if it gets recommendation tokens as control). Instead, it will generate an utterance that talks about books.

- **Steps 8 and 9:** The generated response will be then provided to the user and stored in the dialogue history so that our system will consider it during the following conversation steps.

All steps from 1 to 9 will be repeated n-turns until the user decides to quit the conversation, which will be predicted in step 5 by the DST component. If Bob decided to end the conversation, then the system will execute the steps described in the next paragraphs.

- **Movie Refining**
  - **Step 10:** When the user expresses his willingness to quit the conversation, our system will then concatenate his movie history with the short-term interactions discussed during the conversation to create the movie interactions input.

  - **Step 11:** Given the new refined user interactions, the sequential movie recommendation model (explained in chapter 5) will predict a new personalized movie that may satisfy the user better than the short-term predicted movies, as in this step, we are combining both short-term and long-term interactions to benefit the most from user-watching behaviour.

  - **Step 12:** The refined recommendation is provided to the user within a natural language sentence that also includes the end of conversation context (as demonstrated in the last utterance of figure 3.7).

- **Information Update**

  - **Step 13:** Now that the conversation has ended, we need to update the user model, as the user might express new information about himself within the

conversation utterance. This is done in three steps:

- **Step 13.1:** Given the movies discussed within the conversation, the long-term interactions are updated by concatenating the short-term interaction to them, so that in the next conversation session, the model will be aware of the previously mentioned short-term movies.

- **Step 13.2:** Given the user's previous preferences and conversation utterances, the user's preferences will be updated. To accomplish this, we fine-tune a pre-trained summarization model (T5) [**113**] on a conversation and preferences pair of texts where the conversation utterances are given as input to the model and where the preferences are given as labels. Given this setup, the T5 model aims to summarize the text of the conversation and generate a profile-based text. More details about this model are defined in appendix section A.2.1. We did not discuss this module in our thesis chapters due to the limited number of pages that we have. Therefore, we decided to discuss the main components of recommendation and leave this component for the appendix section. The newly generated profile information will be concatenated with the previous profile text.

- **Step 13.3:** The final step in the user model updating process is refining the user's personality traits using conversation utterances. However, it is important to mention that we employ this process only once for each step (only during the user's first conversation interaction with our system). We argue that the personality of a user is an identity aspect that cannot easily be changed, especially for short-term periods [**114**]. Therefore, the user's personality traits are refined only after the first conversation in order to build more trustful personality information. At first, the personality is predicted from the user's textual profile information using the personality prediction module, then it is refined using the user conversation behaviour. (more details about the personality prediction model are defined in chapter 4). This personality information is stored in the user model so that the CRS can use it to personalize the following conversations.

By executing the previous steps to update the user model, the working pipeline of our CRS reaches its end.

3.3. Conclusion:

Throughout this chapter, we have highlighted the general overview of our proposed **PPerMo** "**P**rompt-based and **Per**sonality-aware **Mo**dularized" CRS framework. We have discussed in depth the role of each component within our solution. Finally, we have highlighted the general working pipeline of our proposed recommendation framework as well as the movie refining mechanism.

In the following chapters, we will discuss how we managed to design and evaluate each component in our modularized system. The next chapter discusses our first research goal, which is the Personality Prediction component.

Chapter 4

The Personality prediction module: AWS

This chapter discusses the personality prediction module. As we have already mentioned in the previous chapter, this component is used to infer user personality traits within a session of dialogues.
As discussed in the "ELECTRA: Pre-training Text Encoders as Discriminators rather Than Generators" paper [**43**], ELECTRA outperforms many state-of-the-art models on different NLP tasks due to its capability of encoding contextual data much faster than other existing architectures. Our personality prediction component targets the prediction of individual personality traits from textual data. Thus, detecting these traits with a Context-specific ELECTRA-based language model named *AWS-EP* "All Weights Shared ELECTRA for Personality prediction" which is A multi-task learning approach for individual personality prediction. Our proposed AWS-EP model is trained using the Pandora dataset [**39**], and evaluated on different other datasets (Pandora test subset, MBTI-Kaggle dataset [**115**], and MyPersonality dataset [**116**]).
The rest of this chapter is structured as follows: Section 1 discusses the motivation behind our proposed solution. Section 2 highlights the architecture of our proposed approach. Section 3 discusses the training and evaluation data as well as the pre-processing methods. Section 4 conducts the experiments and results compared to different baselines and state-of-the-art models. Finally, section 5 concludes this chapter and presents the next module chapter.

4.1. Motivation and Contribution

Throughout this section, we highlight our contribution and the motivation behind the first objective in our research study (as shown in section 1.2) which consists of proposing a novel multi-task personality prediction approach that uses the weight sharing mechanism to simultaneously predict different personality test systems.

Understanding and modeling a personality can be a very challenging task. There are so many different characteristics that are involved in modeling the persons' personalities, such as their behaviour, emotions, sociability, feelings, thinking patterns, etc. Also, every person has his own unique characteristics and preferences that make up his personality. Moreover, different personalities can share the same characteristic which makes distinguishing them harder. [**117**]. Various psychological research show that the words that people tend to use in daily life, and on social media platforms, highly reflect their personality, cognition, and emotions [**40**]. Therefore, in modern times, there is a massive interest to design automatic personality learning models that use the individual's social media posts to predict their personality traits. Also, different research have demonstrated a strong link between individuals' personalities and movies recommendation [**16**]. This highlights the importance of this component in our research pipeline, as we will be using it to enhance the effectiveness and the personalization performance of the different conversational recommendation components (discussed in chapter 3). We highly believe that having an effective recommendation process depends on the active user's personality in a way that similar users that share the same personality and preferences tend to like similar or the same movies.

To model the personality of people, psychologists have defined different personality tests. The most popular tests are the MBTI 2.1.1 and Big Five tests 2.2 [**27**]. To the best of our knowledge, all existing automated personality prediction approaches mainly focus on predicting these two personality test systems independently. This means that the existing models can predict only one personality test (whether modeling the MBTI or the Big Five system). To the extent of our knowledge, no research study predicts both tests simultaneously.
Therefore, to encode the diverted individuals' information and enhance the personality trait prediction effectiveness, we aim to predict both personality tests simultaneously. Using pre-trained models and multi-task learning approaches, we propose *the first automated learning model that simultaneously predicts both MBTI and Big Five personality trait factors*. Our model is a *multi-task learning model* (a regression task for predicting the Big Five traits, and a classification task for predicting the MBTI traits). By training our model on predicting both MBTI and Big Five personality factors together we aim to employ the MBTI's information to improve the Big Five's traits detection, as well as benefiting from the Big Five's information to improve the MBTI's traits detection. Therefore, using the weight sharing mechanism we discuss in this chapter a potential relationship between both MBTI and Big Five personality predictions. To summarize, our proposed AWS-EP (All Weights Shared ELECTRA for Personality prediction) model consists of a Multi-Layer Perceptron network (MLP) with double head layers (one for the regression task and the other is for the classification task) built on top of a fine-tuned ELECTRA transformer model to model both

MBTI and Big Five personality test systems. Therefore, we are not only fine-tuning the hyperparameters of the ELECTRA model, but we are also training new hyperparameters on top of them (the MLP network hyperparameters).
It is important to highlight that the idea of employing multi-task learning for social media-based personality detection is not new. Therefore, our contribution in this work is not creating a novel model architecture for the NLP (Natural Language Processing) field in general. The core novelty of this chapter is the implementation of different existing NLP mechanisms (pre-trained models, multi-task learning, and weight sharing) to create a novel solution for the personality trait prediction problem. We aim also to investigate the impact of using simultaneously different personality tests to improve the prediction of the individual's traits. Moreover, we aim to explore the Electra model performance on the personality trait detection task compared to the existing state-of-the-art models.
Our work is similar to Matej *et al.*'s research [**39**] in terms of predicting the Big Five personality traits as a regression task. However, instead of using 5 different learning models (one model for each personality trait), we propose one multi-task learning model that predicts all traits simultaneously. In addition, our work is similar to Yash *et al.*'s research [**42**] in terms of using multi-task learning for personality prediction. However, instead of using it to predict the individual's emotion and only the Big Five personality trait test as two classification problems, we use the multi-task approach to improve the results of both the Big Five and MBTI personality tests as a regression and a classification problems by predicting them simultaneously. In Yash *et al.*'s study the authors discuss a strong relation between predicting the user's emotions and its Big Five personality. They argue that using shared information to perform two classification predictions at once (one for the individual's emotions and the other for its Big Five personality traits) can improve the performance of the personality prediction task. In our work we acknowledge this assumption and we argue that using shared information to simultaneously predict different personality frameworks at once as a classification and a regression tasks can also enhance the personality prediction performance. Besides the AWS-EP model, we have also designed three other local baselines named *OC-EP* "Only Classification ELECTRA for Personality prediction", *OR-EP* "Only Regression ELECTRA for Personality prediction" and *EWS-EP* "ELECTRA Weights Shared ELECTRA for Personality prediction" to locally measure the performance of the AWS-EP model and the effect of using the weight sharing mechanism. Despite the local evaluation we also report the performance of our model compared to state-of-the-art baselines.

Experiments and results demonstrate that our model outperforms state-of-the-art baselines across multiple well-known personality data sets, it even outperforms new language model-based systems such as BERT. Also, experiments show that both MBTI and Big Five

personality predictions are dependent on each other to some degree. The model architecture, data description, and training/evaluation results are explained in the following sections. The next section highlights the AWS-EP and the different local baseline architectures.

4.2. Models' Architecture

Throughout this section, we define the different OC-EP, OR-EP, EWS-EP, and AWS-EP local baseline architecture. The four architectures are built on top of the ELECTRA model (appendix section A.3.1 highlights more details about the ELECTRA working mechanism). As we have already mentioned, in order to investigate the weight sharing performance for both the classification and regression personality prediction tasks, as well as locally evaluate our AWS-EP model, we created three different baseline models named OC-EP, OR-EP, and EWS-EP. We were curious to know if the single-task prediction models (predicting MBTI and Big Five personality tests independently) would perform better than the weight-shared multi-task models, and if sharing more weights between the two tasks can help increase the effectiveness of the prediction model.
To start with, we first discuss our proposed AWS-EP model architecture.

- **AWS-EP:**
  As we have already discussed, our proposed solution is designed to predict both MBTI (classification) and Big Five (regression) personality traits using a fine-tuned ELECTRA model, a multi-head MLP network and a weight sharing mechanism. Therefore, we have two output heads in our model. Figure 4.1 presents a general abstract architecture of our AWS-EP model.
  The two heads are defined in pink and blue colors in figure 4.1 (pink for the MBTI classification head and blue for the Big Five regression head). The shared weights between these two heads are coloured in gray. We can see that both prediction heads share all of the network weights (the pre-trained ELECTRA and MLP weights), exception for the last two layers' weights, where each layer's parameters are learned independently to fulfill the end task (classifying the MBTI traits or predicting the Big Five value traits).
  The output of the last Big Five head layer defines the numerical values (from 0 to 100) of each Big Five system personality factor (R_{y1}, R_{y2}, R_{y3}, R_{y4}). The output of the last MBTI head layer defines the probabilities of each category in the MBTI system, where C_{y1}, C_{y2}, C_{y3} and C_{y4} define the four MBTI personality factors. It is important to mention that as we are looking to get probability values as final outputs of the MBTI head prediction, we have applied a sigmoid function on top of the MBTI prediction layer head. The reason behind using the sigmoid function instead of the softmax function is that the softmax function is generally used when

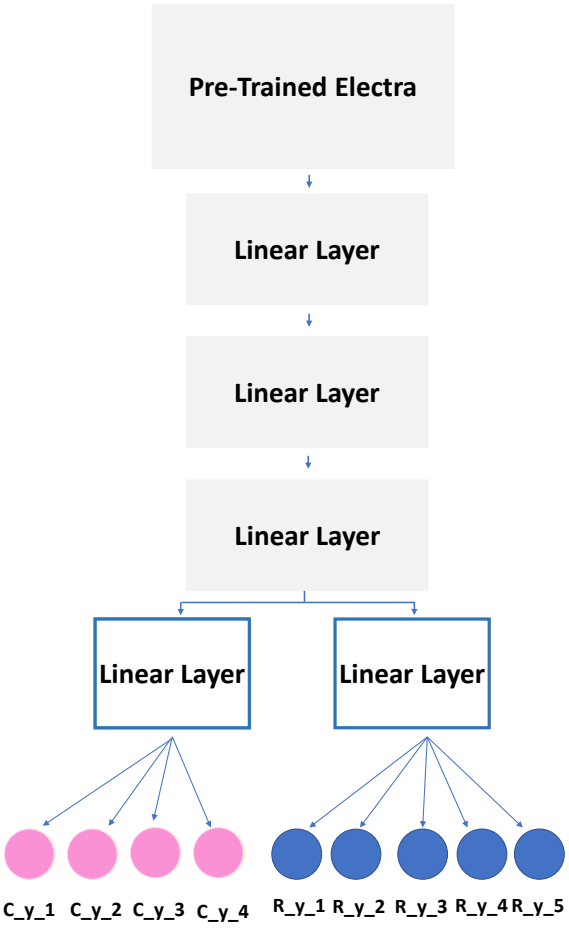

Fig. 4.1. AWS-EP Abstract Architecture.

we have a multi-classification task (for example, we need to choose only one class out of five). However, in our work, we have a multi-label task (out of five classes, we can choose one, two, three, or even all five). This model is trained using a combination of the MSE and BCE loss functions and also to minimize both $LOSS_{class}$ and $LOSS_{reg}$ losses.

$$LOSS_{class} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} Cy_{ij} \cdot \log(\hat{Cy_{ij}}) + (1 - Cy_{ij}) \cdot (1 - \log(\hat{Cy_{ij}})) \tag{4.2.1}$$

where N $\{1..n\}$ defines the data size, M defines the different classes $\{1..4\}$, Cy_{ij} defines the i^{eth} row and j^{eth} class original value $\{0,1\}$, and $\hat{Cy}_{ij}$ defines the i^{eth} row and j^{eth} class predicted value $\{0,1\}$

$$LOSS_{reg} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}(Ry_{ij} - \hat{Ry}_{ij})^2 \tag{4.2.2}$$

where N defines the data size, M defines the different labels {1,..,5}, Ry_{ij} defines the i^{eth} row and j^{eth} label original value {0,..,100}, and $\hat{Ry}_{ij}$ defines the i^{eth} row and j^{eth} predicted value {0..100}

The objective is to minimize both the $LOSS_{class}$ and the $LOSS_{reg}$ (equation 4.2.3), where X defines the training data and $\theta_{class,reg}$ defines the model learning parameters. In AWS-EP the model parameters are a combination of both regression and classification weights.

$$min_{\theta_{class,reg}} \sum_{x \in X} LOSS_{class}(x,\theta_{class,reg}) + LOSS_{reg}(x,\theta_{class,reg}) \tag{4.2.3}$$

Combining the two losses into one loss helps the model to focus on both tasks during the training phase (for example, if the model performs well on the classification task then the BCE loss will be small and if the model is providing poor performance for the regression task then the MSE loss will be high. Summing them together will produce a high loss. This will encourage the model to focus more on the task that produces high loss which is the MSE in this example). Figure A.4 in appendix section A.3.2 highlights a toy example of the working mechanism for our AWS-EP model.

- **OC-EP:**
  Unlike the AWS-EP model which is designed to predict both classification and regression tasks, the OC-EP baseline is designed only for the classification task (predicting the MBTI categories). It is independent of the regression task. Thus, there is no multi-task learning or weight sharing approaches defined in this model. It is a simple model that only performs a classification task.
  We designed this baseline to investigate whether the weight sharing and the multi-task learning defined in the AWS-EP model are necessary for increasing the classification effectiveness or not.
- **OR-EP:**
  This baseline is designed only for the regression task (predicting the Big Five categories). It is independent of the classification task. Similar to the OC-EP baseline, there is no multi-task learning or weight sharing approaches defined in this model. Therefore, it is a simple model that only performs a regression task. This baseline is designed to investigate whether the weight sharing and the multi-task learning defined in the AWS-EP model are necessary for increasing the regression

effectiveness or not.

- **EWS-EP:**
  Compared to both previous baselines, this model is the closest one to our AWS-EP model. This baseline employs both multitask-learning and weight sharing approaches and it is designed to predict both classification (MBTI) and regression (Big Five) tasks simultaneously. However, the difference between it and the AWS-EP model is that in this baseline both classification and regression heads only share a small proportion of similar weights. They share only the pre-trained ELECTRA model weights. They do not share the MLP network weights. Thus, each prediction head has its own MLP network weights (as shown in figure A.6 in appendix section).
  This baseline is designed to answer the question of whether a small weight sharing degree can be sufficient for the personality prediction task or if the more weight we share the more effective performance we get.

Now that we have defined the main architecture of our personality prediction model and the different baselines architectures. The next section discusses the data that we used to train our model as well as the different pre-processing steps.

4.3. Dataset and Data Preparation

To train our model, we used a recently published personality dataset known as Pandora [**39**]. Pandora dataset is currently the largest and richest personality prediction dataset. It annotates different personality trait tests such as the Big Five, MBTI, and Enneagram personality models. Therefore, in this section, we dive deeper into exploring this dataset by providing some statistical insights and highlighting the different information provided by the instances of this dataset.

4.3.1. Pandora Dataset

Pandora is the largest and the first dataset in the research field that contains over 17 million Reddit comments written by more than 10k users annotated with both MBTI and Big Five-Factors with users' demographical features (age, gender, and location). 1.6k users are labeled with the Big Five personality model with over 3M comments. It also comprises 9k users' annotations with the MBTI personality trait. Pandora was published in 2021 by the Text Analysis and Knowledge Engineering Lab (TakeLab). Since then different research showed the effectiveness of using this dataset to predict the user personality for downstream tasks [**39, 40, 42**].
Due to the massive insights within this dataset and its significant textual data amount, we decided to use it as our main training dataset. It is important to highlight that Pandora

is a private dataset and the authors employ different terms of use to protect the users within this dataset [**118**]. Some of the terms consist of not transferring or reproducing any part of the dataset, attempting to identify any user in the dataset, contacting any user in the dataset, displaying users' names and sensitive messages publicly, and reporting findings publicly unless it is at an aggregate level. The reason behind these obligations is to protect the users' information within this dataset.
Therefore, we are grateful to the Pandora dataset authors for providing us with this dataset. Without their kindness, this research step would never have been accomplished.
This dataset contains two main files.

- **The Author Profiles file**:
  This file contains the main attributes that define an individual such as their Reddit profile name, their MBTI personality trait, their Big Five personality traits, their country, their state, their gender, their number of posts or comments, etc. In this thesis, we mainly focus on using only the Big Five and MBTI personality profile information for privacy reasons (as discussed in appendix section A.3.3). Also, these are only the labels' attributes for the MBTI and Big Five personality models that we need to predict.

- **The All Comments Since 2015 file**:
  This file highlights the different comments posted by each user specified in the author profile file from 2015 to 2020. It contains over 17M rows with different attributes. Each row defines the author's name, their comment body/text, the UTC time for their post/comment, the language for their comment, the number of words within their comment, their link id, their subreddit id, their subreddit value, etc.
  In this thesis, we utilize only the body of the author's comment (the text posted by the user). As the main purpose of our personality prediction component is to predict the personality of a user only from its written comments and posts, we didn't use the other dataset attributes. By using only text to train our personality prediction model we can re-use this model in other different situations such as predicting the users' personalities from their dialogues. The following chapters discuss this idea in detail.

Using the authors' profiles and all comments since 2015 information we managed to merge all of the necessary knowledge for each user by relying on the author column within the two tables to create the final version of our training dataset.
Table 4.1 highlights different instance examples within our final filtered dataset.

Table 4.1. Example of the final training dataset instances.

Features		Regression Labels (Big Five)					Classification Labels (MBTI)			
Author	Body	AGR	OPN	CSN	EXT	NEU	I/E	N/S	T/F	P/J
12345jk12345	Are there any cheaper, decent quality black boots that match the style of Thursday Boot Co.	0.0	37.0	72.0	72.0	3.0	0	0	1	1
12345jk12345	The info in the video is relatively accurate but good lord that guy is annoying	0.0	37.0	72.0	72.0	3.0	0	0	1	1
12345jk12345	Appreciate it buddy. Which order of those pictures would you recommend?	0.0	37.0	72.0	72.0	3.0	0	0	1	1
-dyad-	I failed both ... I'm great at reading people irl	60.0	67.0	45.0	10.0	47.0	1	1	0	1
-dyad-	Indeed. So much unnecessary anger and spitefulness must make life difficult. What do they *think* goes on at Burning Man?	60.0	67.0	45.0	10.0	47.0	1	1	0	1
-dyad-	That's awesome that you had such a wonderful time! Sounds like it was a really special night you'll remember for a long time.	60.0	67.0	45.0	10.0	47.0	1	1	0	1

Besides using Pandora as our main training and evaluation dataset, we also used two other different datasets named My personality dataset [**116**] and MBTI Kaggle dataset [**115**], to evaluate the generalization performance of our personality prediction model.

4.3.2. MyPersonality Dataset

This dataset was collected in 2013 by Celli *et al.* [**116**]. It contains more than 250 different users with 10000 labeled Facebook statuses in total with the Big Five personality traits. It also combines network properties such as network size, density, transitivity, etc. As we have already discussed, for our AWS-EP model we need only to pass text posts and comments as inputs. Therefore, we ignored all of the other features and we kept only the post feature and the Big Five regression label values.
Table 4.2 highlights different instance examples for the filtered MyPersonality dataset.

Table 4.2. My personality dataset instances example.

Author ID	STATUS	EXT	NEU	AGR	CON	OPN
b7b7764cfa	I likes the sound of thunder.	2.65	3.00	3.15	3.25	4.40
b7b7764cfa	Is so sleepy it's not even funny that's she can't get to sleep.	2.65	3.00	3.15	3.25	4.40
b7b7764cfa	I likes how the day sounds in this new song.	2.65	3.00	3.15	3.25	4.40
deb899	Little things give you away.	2.15	2.15	4.10	2.90	4.60
deb899	Is wishing it was Saturday.	2.15	2.15	4.10	2.90	4.60
deb899	Is studying hard for the G.R.E	2.15	2.15	4.10	2.90	4.60

Where "STATUS" defines the text written by the user, and "EXT", "NEU", "AGR", "CON", and "OPN" highlight the Big Five personality traits.

4.3.3. MBTI-kaggle Personality Type Dataset

This data was collected by using the PersonalityCafe forum, as it displays a large selection of people and their MBTI personality type and what they have written. It contains over 8k rows of data, where each row represents a different person. For each person, we have the last 50 texts they posted separated by the "|||" character.

This dataset provides a large granularity summary for the MBTI personality traits labels (it stores the type of the personality example "INTJ"). To make it convenient for our MBTI model evaluation, we generated the four MBTI factors using the type attribute (see section 2.1.1 for more explanation of what each personality type refers to). Table 4.3 highlights the pre-processed version of the MBTI Kaggle dataset.

Table 4.3. MBTI-kaggle dataset pre-processed instances example.

Author ID	Posts	I/E	N/S	T/F	P/J
0	I'm finding the lack of me in these posts very alarming.||| Giving new meaning to "Game" theory.||| Real IQ test I score 127. Internet IQ tests are funny. I score 140s or higher.	0	1	1	1
1	Does being absolutely positive that you and your best friend could be an amazing couple count?||| Have you noticed how peculiar vegetation can be?||| This 5-year-old sentence is incredibly accurate and beautiful.|||	1	1	1	1
3	You're fired.||| Never mind. Just go on permanent vacation ||| Sometimes I just really like impoverished rap music.|||	0	1	1	0

Where "Posts" defines the text written by the user, and "I/E", "N/S", "T/F", and "P/J" highlight the MBTI personality traits.

4.4. Experiments and Results

Experiments and results are done using three different datasets. To investigate the performance of our proposed models, we used the Pandora dataset. This dataset combines both Big Five and MBTI features. To evaluate the different model's generalization performances, both MyPersonality [**116**] and MBTI-kaggle datasets are used. The MyPersonality dataset is used for the Big Five features validation, and the Myers-Briggs Personality Type dataset is used to validate the MBTI features.

4.4.1. Training Properties

The Pandora dataset is randomly partitioned into three parts during the training phase: training, validation, and test subsets. 20% of the data was considered a test set and 80%

was considered a training set. Then to create the validation data, we split the training set into two sub-parts. 20% were considered validation data, and the rest were kept to train the model, which is done using the scikit-learn library. The same data splitting process was done for all of the different experiments using a seed value of zero. The sentence words were embedded into a 256-length token vector, using the tokenizer of the pre-trained Electra-small model from PyTorch hugging face framework. The pre-trained model was fine-tuned on the Pandora training subset, and all models (OC-EP, OR-EP, EWS-EP, AWS-EP) were trained for 10 epochs. We also compared the current validation results with the least validation loss for each epoch and stored the model that gave us the least generalization loss. In our experiments, we reported the performance of a single run (10 epochs) for each model. Table 4.4 highlights the different hyperparameters used during the training phase.

Table 4.4. The model's hyperparameters.

Hyper-Parameter	**Value**
Epochs number	10
Optimizer	Adam optimizer
Learning rate	2e-5
Weight decay	0.01
Activation function	LeakyRelu
Dropout degree	0.4
Classification loss (CL)	BCE with logits loss
Regression loss (RL)	MSE loss
Global loss	(CL+RL)/2
Batch size	15
Trainable parameters	13542167

Different experiments were conducted to investigate the different generalization performances of the proposed baselines (OC-EP, OR-EP, EWS-EP, AWS-EP), and their performance was compared with state-of-the-art models. We used the google collaboratory pro version as our computing infrastructure (166.83 Gb hard drive capacity, 25.46 GB memory capacity, and a 1 Tesla P100-PCIE GPU), which allows us to use a 20h window session of these computational resources.

The choice of the batch size value, and the 10 epochs are due to the availability of the training computation resources that we had. As some models were incapable of being trained with more than 15 samples per batch and more than 10 epochs within a 20h window session. Also, as described in Mosbach *et al.* paper [**119**], the best way to fine-tune a pre-trained model is to choose a small learning rate with a bias correction training algorithm to avoid vanishing gradients. Therefore, we used the AadamW optimizer model with a 2e-5 learning rate and a 0.01 weight decay value.

4.4.2. Training Results

Training the four baselines using the same hyper-parameters led to different performance results on the training and the validation sets. Figure A.5 in appendix section highlights the different training and validation performance for each baseline.
Training results show that EWS-EP and AWS-EP models (multi-task models) have the highest trusted results in terms of generalization performance for the MBTI and Big Five traits. We can see that both are trying to reduce at the same time the training and validation loss during each epoch. This highlights the importance and the promising performance of the multi-task learning approach compared to the single-task learning approach results. We can see that the validation error is almost constant along the training epochs for the OC-EP. So during the learning phase, this model is trying to decrease the training loss while keeping the validation loss almost the same. Therefore, EWS-EP, and AWS-EP are better than the OC-EP on the classification generalization task.ing

4.4.3. Generalization Results

During the experiment phase, we were curious about having effective results for predicting the Big Five and MBTI personality traits and investigating to which degree these two tests are similar. We also were curious to know the effect of weight sharing on the model predictions. Table 4.5 highlights the generalization performance of our AWS-EP compared to the local baselines.

Table 4.5. The baselines performance on different metrics.

	Classification				**Regression**		
Models	**Accuracy**	**Precision**	**Recall**	**F1-score**	**MSE**	**R2_score**	**P_r_C**
OC-EP	0.738	0.738	**1.0**	0.844	-	-	-
OR-EP	-	-	-	-	2910.39	-2.82	0.32
EWS-EP	0.739	0.739	**1.0**	0.845	839.03	0.05	0.47
AWS-EP	**0.788**	**0.792**	0.94	**0.860**	**564.12**	**0.35**	**0.66**

Table 4.5 highlights the performance of the proposed baselines (OC-EP, OR-EP, EWS-EP, AWS-EP) on the unseen Pandora test subset. The OC-EP model provides a performance for the accuracy and F1-score metrics with 0.738 and 0.844, respectively. Results show that the OR-EP model provides a performance in MSE, r2-score with a 2910.39, and -2.82 respectively. By introducing a low level of weight sharing in the EWS-EP baseline, both classification and regression results improved. Moreover, allowing for more weight sharing between the MBTI and Big Five prediction tasks in the AWS-EP model significantly improved the regression and classification results. It is also clear that the regression head is the one that benefits the most from the weight sharing with more than 100% increase in

terms of the Pearson r correlation metric compared to the OR-EP model. We also report a five-fold decrease in MSE compared to the OR-EP model.
The experiments demonstrate that the more we allow the Big Five prediction head to know and share weights with the MBTI model, the better results the head provides. This demonstrates the high correlation between the Big Five and MBTI personality test systems. The results provided in table 4.5 show that the most effective model from the 4 baselines is the AWS-EP model. For this reason, we aim to investigate the performance of this approach further and evaluate its generalization performance compared to state-of-the-art models that were evaluated on the same dataset. Tables 4.6 and 4.7 provide more details on the AWS-EP model performance for each trait factor.

Table 4.6. The AWS-EP classification and regression performance on the Pandora benchmark dataset.

Classification performance

MBTI factors	accuracy	precision	recall
Introverted	0.7583	0.7629	0.9233
Intuitive	0.9131	0.9131	1.0
Thinking	0.7889	0.7939	0.9797
Preceiving	0.6916	0.7014	0.8625
Average	0.7880	0.7928	0.9414

Regression performance

P-r-C metric	**0.66**

Table 4.7 shows that our proposed AWS-EP model provides effective results for different classification and regression metrics (P-r-C, accuracy, precision, and recall). For the Big Five regression task, we achieved a 0.3971 units increase in the Pearson correlation metric compared to the regression results reported in the Pandora state-of-the-art paper [**39**]. Moreover, results show that our AWS-EP model outperformed state-of-the-art baselines on predicting the MBTI personality task on all of the different traits.

Experiments show a 0.2648 F1-score units increase for the Introverted factor compared to the PQ-Net baseline, a 0.4019 units increase for the Intuitive factor, a 0.2207 units increase for the Thinking factor, and a 0.1862 units increase for the Perceiving factor. Overall, we achieved a 0.2684 units F1 score average increase for all of the MBTI factors compared to the PQ-Net state-of-the-art model.
Despite the promising performance of our AWS-EP model on the Pandora dataset, we were curious to measure its generalization performance using different unseen personality datasets. The datasets we use in this experiment are the MBTI personality dataset from Kaggle [**115**],

Table 4.7. The AWS-EP Macro-F1 performance on each MBTI trait compared to state of the art models on the Pandora dataset

Models	Introverted	Intuitive	Thinking	Preceiving	Average
SVM	0.4474	0.4692	0.6462	0.5632	0.5315
XGBoost	0.4599	0.4893	0.6351	0.5555	0.5350
LSTM_{Glove}	0.4801	0.5201	0.6348	0.5621	0.5493
$\text{BERT}_{fine-tune}$	0.5660	0.4871	0.6470	0.5607	0.5652
AttRCNN	0.4855	0.5619	0.6439	0.5726	0.5660
SN-Attn	0.5698	0.5478	0.6095	0.5481	0.5688
PQ-Net	0.5707	0.5526	0.6564	0.5874	0.5918
AWS-EP	**0.8355**	**0.9545**	**0.8771**	**0.7736**	**0.8602**

and the MyPersonality dataset [**116**]. Table 4.8 demonstrates the generalization performance of the AWS-EP model for the F1 metric on unseen datasets.

As shown in table 4.8, evaluating our AWS-EP model performance on the MBTI Kaggle

Table 4.8. AWS-EP generalization performance on different unseen datasets.

	MBTI Kaggle
Metric	**F1**
TrigNet	0.7086
PQ-Net	0.7132
AWS-EP (Ours)	**0.8276**

dataset compared to the PQ-Net and the Trignet models that were evaluated on the same dataset shows that our proposed solution performs significantly well on the unseen dataset. Although it was only trained on the Pandora dataset, our AWS-EP model outperforms state-of-the-art MBTI Kaggle datasets baselines.

4.5. Conclusion

In this chapter, we discussed the first objective of our research which is the personality prediction component "as shown in section 1.2". We highlighted our contribution by simultaneously inferring different personality tests using a pre-trained transformer model and the weight sharing mechanism to finally improve the personality prediction results. We discussed also the effectiveness of using a multi-task learning approach on top of a pre-trained Electra model for the personality prediction task. Empirical results demonstrate that using shared weights between MBTI and Big Five personality tests outperforms state-of-the-art results for both systems on different metrics. Our results show a strong relationship between predicting both MBTI and Big Five personality tests simultaneously where sharing the same information between both tests significantly improve both prediction results.

In future work, more weight sharing, contextual information, and prediction heads will be considered. Moreover, we are curious to know the effect of demographic information such as (age, gender, country, etc.) on personality detection. Therefore, we aim to enhance the capability of our model by not only considering textual data for fine-tuning the transformer model but also combining it with the tabular data type (demographic information).

Now that we have discussed our research first goal, in the following chapters, we discuss the second goal of our research by highlighting the architectural design of each component within our CRS as well as discussing the remaining contributions highlighted in section 1.2 (the contributions for the recommendation engine, state tracker, and the dialogue generation modules). The next chapter highlights our proposed contribution for the recommendation engine component.

Chapter 5

The Sequential Movie Recommendation module: DBT-SR

This chapter highlights the sequential movie recommendation component, which aims to predict the next movie to watch for a specific user given its previous interactions. This chapter proposes a novel approach to efficiently and effectively train a MLM (Masked Language Modeling) task using transfer learning. We also aim to investigate the impact of using the individuals' personality traits for the next movie sequential recommendation task. This process is very important in our research pipeline since its success will guide us to reach more user satisfaction in terms of the recommended movies.

The rest of the chapter is structured as follows: Section 1 introduces the motivation and the contribution behind this work. Section 2 highlights the model description. Section 3 presents the datasets which we used to train our model as well as conducts the experiments and results compared to local baselines and existing state-of-the-art models. Finally, Section 4 concludes this chapter and presents the following one.

5.1. Motivation and Contribution

Throughout this section, we highlight in detail our contribution and the motivation behind the second objective of our research (as shown in section 1.2) which consists of building an efficient and effective personalized sequential movie recommendation approach using the transfer learning mechanism.

Despite the enormous benefits of personalized recommendation systems, designing an effective type of system still needs a lot of effort. The difficulty of understanding the relationship between the different items and their corresponding users is one of the main problems that made personalizing recommendation systems very hard. Moreover,

the complexity of users' behaviour and the diversity of their preferences made training sequential recommendation systems very expensive and time-consuming.
To solve these issues, different work used neural network approaches to create new representations that effectively encode the different user-item relationships. Previous research applied left-to-right sequential neural networks to encode user historical preferences and interactions, by creating embedded representations for a recommendation goal. These types of encoding approaches are known as unidirectional encoding techniques. Although unidirectional sequential models can encode important information about the user interaction context, these models does not encode all contextual information.
Recent research employ bi-directional and attention-based encoding techniques to benefit the most from the hidden information within the historical users' interactions [**81**]. Despite the effective improvement of these approaches compared to unidirectional models, these techniques still lack the personalization aspects, especially when it comes to using the individual personality traits to personalize an item recommendation. To the best of our knowledge, no existing study investigates the impact of using Big Five personality traits on the sequential recommendation of the next movie to watch problem [**5**].
To address these limitations, we propose a novel *causal-based* and *prefix-based* LM fine-tuning approach for the sequential recommendation goal. We call our proposed model **DBT-SR** "DistilBERT with Transfer learning trick for Sequential Recommendation". Our work uses a simple but effective transfer learning trick to simultaneously fine-tune a prefix-based and a causal-based language model to better understand the user's behaviour. In this work, we also investigate the impact of using the individuals' personality traits for the sequential recommendations problem. We used the AWS-EP model discussed in the previous chapter 4.2 to take advantage of users' personality traits by inferring them from their corresponding preferences information.
To the extent of our knowledge, this is the first work that uses the DistilBERT transformer model to encode both users' personality traits and their historical interactions for the next movie to watch recommendation task using a transfer learning trick. Also, as most proposed papers fine-tune the transformers models using only textual data, in this work, we aim to fine-tune the DistilBERT model with not only textual but also tabular data "numerical values". Moreover, to measure the performance of the DBT-SR model, we propose two other baselines named *DB-SR* (DistilBERT Sequential recommendation model), and *DBTP-SR* (DistilBERT with Transfer learning for Personality-aware Sequential Recommendation model).

Experiments and results show that our proposed transfer learning trick significantly improves the recommendation performance.
In this work, our main contributions compared to the existing research are summarized as

follows:

- Using DistilBERT for sequential recommendation downstream task.
- Proposing a simple but effective transfer learning approach for MLM tasks to better understand the users' historical behaviour.
- Employing and investigating the Big Five personality traits performance with an attention-based transformer model for the next movie to watch recommendation task.
- Fine-tuning the DistilBERT pre-trained model using both numerical and textual data input types.
- Reporting and discussing the evaluation results.

The following section discusses in detail the used transfer learning trick, as well as both *DB-SR* and *DBTP-SR* proposed baselines.

5.2. Models Description

The 3 baseline architectures proposed in this chapter are built on top of the DistilBERT pre-trained model (appendix section A.4.1 highlights more details about the DistilBERT approach). Using the Masked Language Modeling (MLM) objective, we aim to fine-tune the DistilBERT pre-trained weights to produce a more contextual representation of the users' movie-watching history, to finally recommend the convenient movie. To do that both efficiently and effectively, we propose a transfer learning trick to encourage the model to provide better generalization results.
Before we start highlighting the different model architectures, we first start explaining our proposed transfer learning trick which we called *Transfer learning for Causal and prefix-based language model for sequential recommendation approach.*

5.2.1. Transfer Learning for Causal and Prefix-based Language Model for Sequential Recommendation Approach

In standard causal learning (also known as MLM) we randomly start masking out a percentage of the words within a sentence, then the model is trained to predict the right words for the masked tokens using both sides' information about the unmasked tokens. In this approach, masking a large number of words can lead the model to not sufficiently understand the contextual information within the sentence, since too much information is hidden. Also, applying too little masking can be too expensive and could require more data to learn enough context as the model will condition on more information to predict the masked token. thus, the computation will be more expensive. Also, as having more information to condition on, the learning task became simple and the model can easily

predict the missing tokens as it has more contextual information (less information is hidden). Conditioning on too much information can easily lead to overfitting, where the model is capable of providing a significant performance on the training set but poor performance on the test set. Therefore, the most proper setup is to use a masking degree that helps both understand the hidden context and train the model efficiently. However, finding such a balance masking degree can be very hard and it requires a lot of experiments.
Moreover, existing MLM-based work generally assume that using the MLM (both side understanding) technique makes the LM capable of learning hidden relationships between the different tokens within a sentence. Though such an assumption has been proved to be true for NLP tasks, we argue that this is not the case for sequential recommendation tasks. By seeing both sides of movie interactions, it is not guaranteed that the recommendation system will significantly learn the sequential structure of the different movies. We argue that knowing both previous and following items during the learning process will introduce a portion of information loss about the sequential structure of the historical watched movies. As some times predicting the right movie, only depend on the preceding movies. Therefore, introducing information on the following items can deviate the model from recommending the right one.

To overcome these limitations we propose a simple but effective transfer learning approach that uses both causal and prefix learning objectives for a sequential recommendation task using two networks. We borrow from the "*Pseudo-Masked Language Models for Unified Language Model Pre-Training* (PMLM)" paper idea proposed by Hangbo Bao *et al.* [**120**], where they used a BERT-based causal and a partially auto-regressive learning approach to train a unified model that does both natural language understanding and generation. Using both causal and auto-regressive learning approaches, the authors showed that the unified learning model is effective and computationally efficient, as the encoded context information is reused by the two language modeling tasks. Therefore, they manage to avoid redundant computation.
Being inspired by this approach which was developed only for the NLP tasks, we propose a similar unified model approach that learns both causal and prefix tasks for the sequential recommendation objective. In this approach, we use two masking techniques, one for randomly masking the different movies within a sequence "for the causal-learning objective model" and one for masking the last token within the same sequence "for the prefix-learning objective model". Then we trained both models to learn from the MLM sequences using the weight sharing approach "Transfer learning". To the extent of our knowledge, this is the first work that uses a transfer learning-based approach with both causal and prefix LM tasks for a sequential movie recommendation goal.

Figure 5.1 highlights our novel proposed transfer leaning-based sequential recommendation approach.

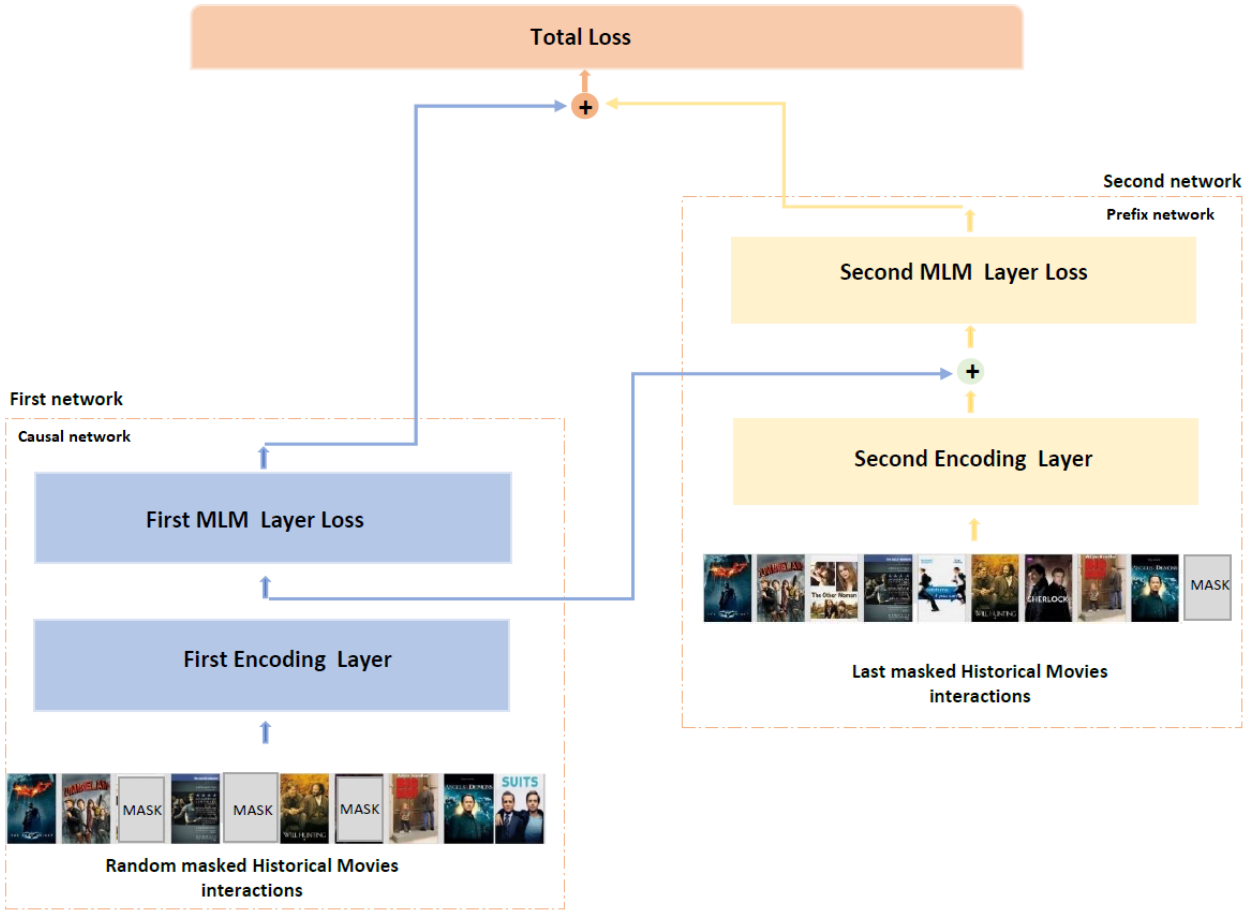

Fig. 5.1. Transfer learning for Causal and Prefix-based language model for sequential recommendation approach.

For an input movie sequence instance x, **the first network *"Causal network"*** takes as input a randomly masked version of the input x (similarly to the MLM training task of BERT), which we called x_{random}. This network is responsible for understanding and encoding the possible relationships between the different interactions. Therefore, it learns the *inter-relations* between the corrupted "masked" tokens. **The second network *"Prefix network"***, takes as input a new version of x which we called x_{last}, where we mask only the last movie within the same sequence of interactions while keeping all of the preceding movies visible. Therefore, it aims to learn the *intra-relations* between the masked movie and all of the previous ones. To do that we use the MLM approach, but we only mask the last token. Therefore, the model tries to predict this masked movie conditioning on all of the previous movie tokens.

We have already discussed that this type of learning setup "Using MLM with too little

masking" can be very expensive and it performs poorly on unseen datasets. Therefore, we apply the transfer learning technique.

Using the transfer learning technique, our prefix network will not be fine-tuned from the start. Instead, for each training step, we take the learned weights of the Causal network encoder and transfer them to the Prefix network decoder "Classification layer". Thus, it is the Prefix network objective to use this transferred information to update its classification layer weights to predict the last masked item.
By masking only the last movie, we encourage our learning model to understand the full previous context of the movie interaction history "intra-context". Thus, to ensure good generalization results, the last masked token will be predicted using the knowledge acquired "transferred" from the Causal network encoder. Therefore, with too little masked input we effectively increased the recommendation performance while stabilizing the computational expense by using transfer learning. Also, to more reduce the computation expense, we used the DistilBERT model instead of the BERT LM which was used by the PMLM work [**120**], as DistilBERT has been proved to be 40% less expensive than the original BERT learning approach.
We can also argue that our proposed solution borrows ideas from the prompting procedure but in a different way. As in the prompt-learning-based approach, the task is to use a pre-trained model and similarly design the input shape to the correspondent shape that the pre-trained model has been trained on. This is exactly what we are doing here, we are using the DistilBERT model and we are reshaping our input data to fulfill the MLM objective that DitsilBERT was trained on.

During the training phase, both causal and prefix learning approaches are trained simultaneously. The same sequence of interactions x will be masked into two forms "x_{random} and x_{last}" (one for the causal model and one for the prefix model). Then the encoded representation provided by the Causal network is transferred to the Prefix network. We designed the loss function as a combination of both the network prediction losses. Equation 5.2.1 defines the loss function used to train our DBT-SR model.

$$\begin{aligned} Loss_{total}(x,\theta_G) &= L_{causal}(x,\theta_{causal}) + L_{prefix}(x,\theta_{prefix}) \\ L_{causal}(x,\theta_{causal}) &= -\sum_{x\in D} \log \prod_{m\in M} p(x_m|X_{/M}) \\ L_{prefix}(x,\theta_{prefix}) &= E(\sum_{x\in D} -\log p(x^{masked}/X_{pre})) \end{aligned} \tag{5.2.1}$$

Where x is the original input $x = \{x_1, x_2, .., x_{|x|}\}$. M is the set of masks' position $M = \{m_1,..m_{|M|}\}$. x^{masked} is the masked movie. D defines the vocabulary token movie set.

X_{pre} is the previous movies within the interaction sequence tokens before the masked token. $X_{/M}$ is the set of unmasked tokens, θ_{causal} is the causal-based model set of parameters, θ_{prefix} is the prefix-based model set of parameters, and θ_G is the global model set of parameters.

By combining both losses, we encourage the causal model to learn the best representation that both understand the relations within the previous interactions (intra-relation) and learn the sequential structure of the movies sequence (inter-relation). At inference time we use the Causal network as our main prediction model, as during the training phase, this model has updated its parameters to make the Prefix network capable of predicting the last masked movie. Therefore, we argue that those learned parameters can be used directly with the Causal network to predict the next movie to watch without needing the second network part. Experiments and results show that the use of our proposed learning approach provides interesting results on both MRR@k and HR@k metrics compared to using only the standard MLM "random masked" or last masked item learning technique separately.

Now that we have defined the main idea behind the transfer learning trick that we propose within this chapter, in the next section we discuss the different architectures of our next movie to watch prediction baselines.

5.2.2. Baselines Architecture

To investigate the impact of including individuals' personality traits for the sequential recommendation task, as well as using the transfer learning approach compared to using the standard MLM approach, we created three different baseline models: DB-SR, DBT-SR, and DBTP-SR models. We were curious to know if including the users' personality traits would help predict more personalized results compared to non-personality-aware models, as well as knowing if our proposed transfer learning trick will enhance the performance of the normal MLM approach on the movie recommendation task. Figures 5.2, and 5.3 describe the main architecture for each baseline.

- **DB-SR: DistilBERT for Sequential Recommendation baseline**

  This is the simplest and the least complicated baseline. It is just a normal causal-based learning model. It only uses the user's historical interactions as inputs with an encoder layers "Movie DistilBERT Encoder". It also uses a standard MLM objective learning technique with a classification layers "Movie-DistilBERT Classification Layer" to predict the next movie to watch. It applies neither the transfer learning approach nor the personalization aspects (personality traits). During the training

phase, the masking degree is applied for each interaction sequences randomly. Then the encoder component is responsible for producing a contextual representation for the masked interaction sequence using the attention mechanism. Given the contextual representation, the DistilBERT classification layers are responsible for predicting the masked movies by minimizing the loss of each masked input. This model is trained using only the standard MLM loss (L_{causal} in 5.2.1) where P defines the softmax probability of the vocabulary tokens.

The objective of this architecture is to investigate whether adding additional information (such as personality traits or transfer learning information) will help improve the performance of the standard MLM (random masked language modeling).

- **DBT-SR: DistilBERT with Transfer learning for Sequential Recommendation baseline**

Instead of using only the standard MLM approach like the DB-SR model 5.2.2, this baseline uses our proposed Transfer learning for Causal and Prefix-based language model for the sequential recommendation approach 5.2.1. We were curious to know whether the use of this approach will provide more effective results compared to using the standard MLM learning approach "causal learning" 5.2.2. This baseline uses the same training data and the same features as the previous model. The only difference is that we are applying the transfer learning trick between two models instead of using one model 5.2.1. Figure 5.2 highlights the DBT-SR model architecture.

This model has two main components. **The first** component is the *Causal-DistilBERT network* which will take a randomly masked sequence of interactions as input, and update its parameters using the standard MLM loss "Causal loss" 5.2.1. **The second** network (*Prefix-DistilBERT*) takes the same sequence of interactions, but only the last element of the input sequence is masked. Then it uses the representation provided by the first network to predict the last masked token. This network updates its parameters using the prefix loss objective 5.2.1. Therefore, as a global loss, we want the final model to learn both the total loss defined in the previous section 5.2.1. In a way that the first Causal-DistilBERT network will update its parameters to learn how to effectively predict the last masked token

The objective of this baseline is to investigate whether our proposed transfer learning approach can improve the sequential recommendation performance compared to the normal causal learning approach 5.2.2. Experiments and results show

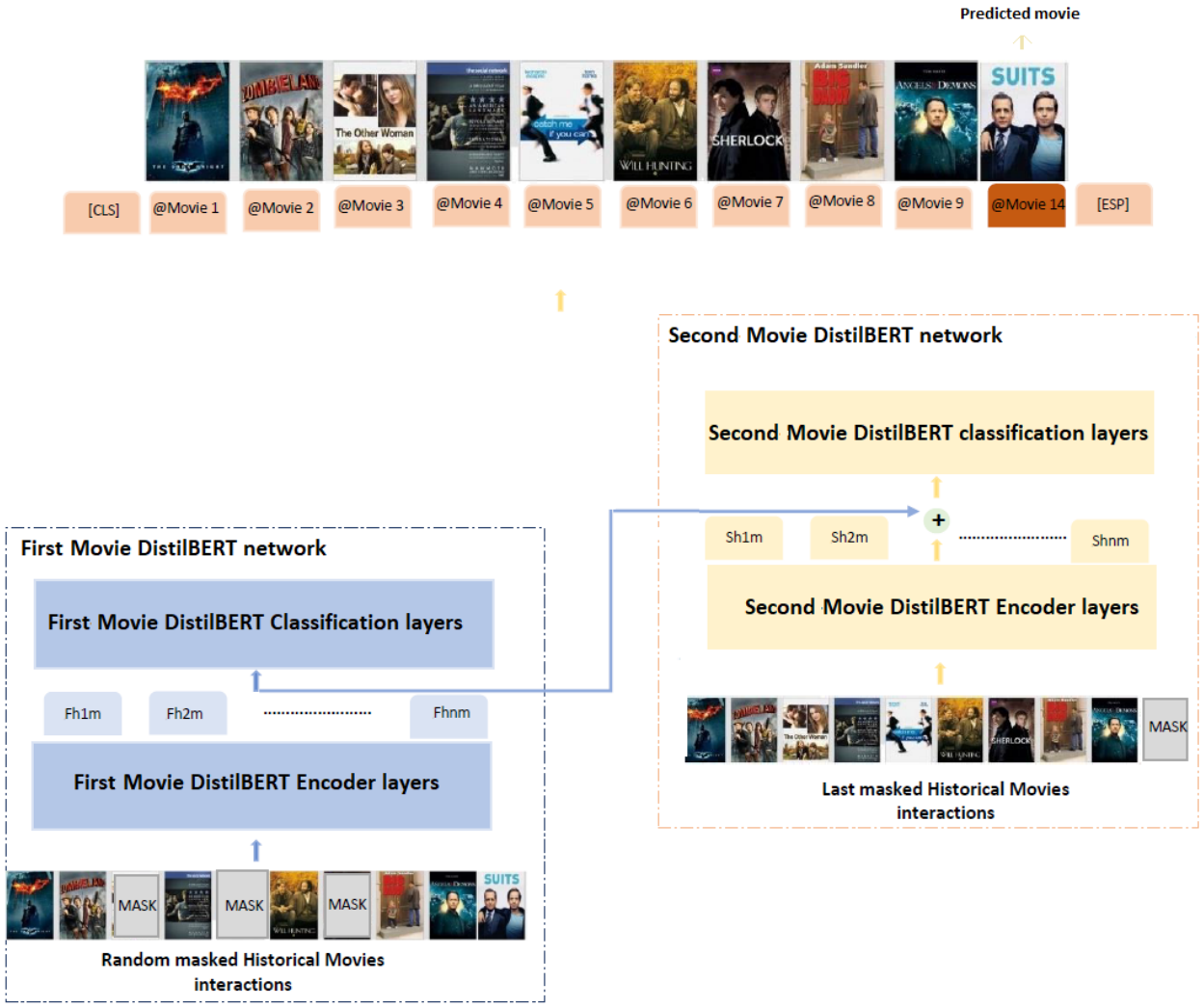

Fig. 5.2. DistilBERT with Transfer learning for Sequential Recommendation model baseline architecture.

that our proposed transfer learning-based approach outperforms the standard causal learning technique for the next item recommendation task on both HRR@k 5.3.1 and MRR@k metrics 5.3.2 for the TG-Redal dataset.

- **DBTP-SR: DistilBERT with Transfer learning Personality-aware for Sequential Recommendation baseline**
  The DBTP-SR baseline architecture is similar to the DBT-SR one. However, for this model, we add an additional component named *Big Five user Personality Encoder* to investigate the impact of employing the user personality traits during the recommendation process. Figure 5.3 highlights the main architecture of this baseline.

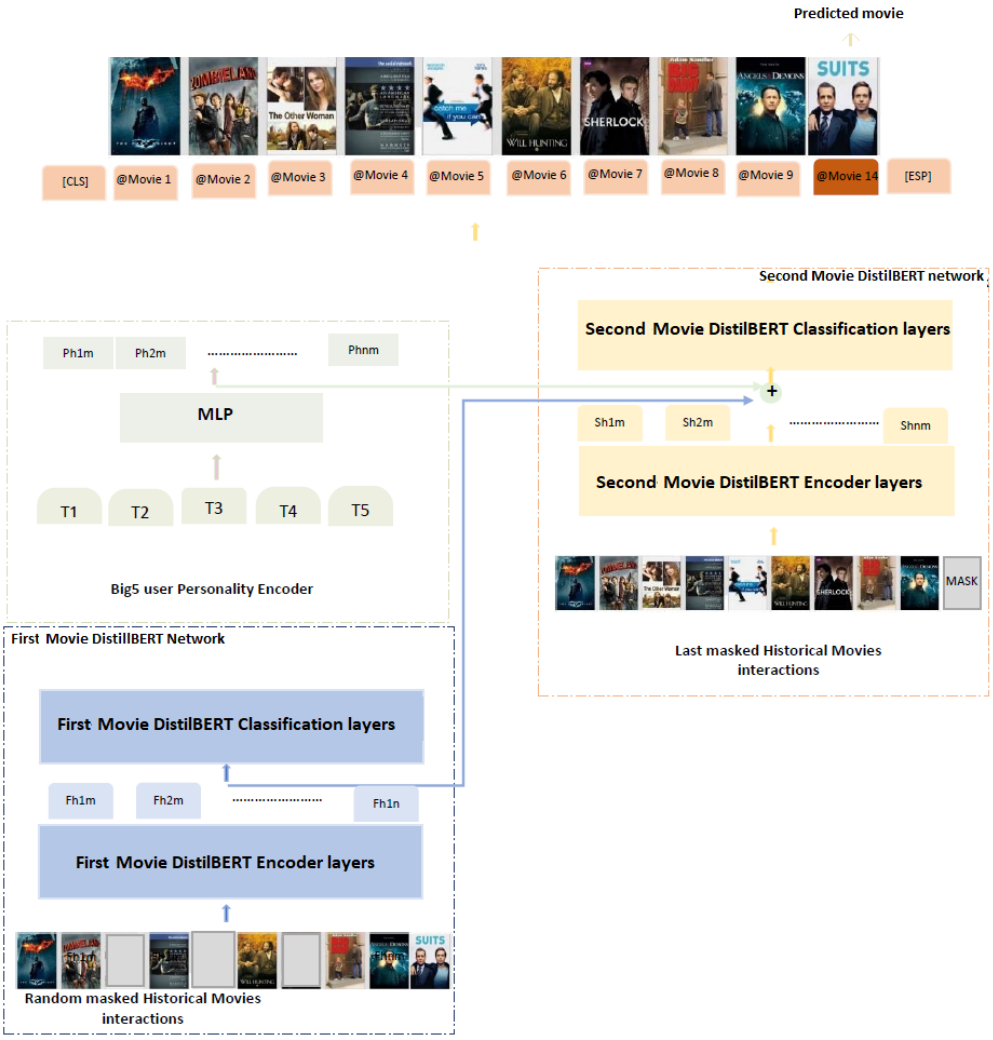

Fig. 5.3. DistilBERT with Transfer learning Personality-aware for Sequential Recommendation baseline architecture.

The user personality traits used in this component are the Big Five traits. We were curious to explore the impact of these personalities on the sequential next movie to watch recommendation problem. To do that the user personality is encoded using an MLP model. We used an MLP network with 5 neurons for the input layer where each neuron corresponds to one Big Five trait. This will result in a contextual representation of the user's personality. The last output layer of the MLP network is a 768 neurons layer size (similar to the shape of the DitilBERT classification layers neurons). This representation will then be combined with the interaction history representation provided by the "Causal-DistilBERT Encoder Layer". Therefore, the final result representation will encode both user personality and its interaction history.

It is important to highlight that standard transformer models' inputs are textual

inputs. However, this architecture uses both numerical (Big Five traits) and textual (movies sequence) data as input to the DistilBERT transformer. Therefore, in this architecture, we use a multi-type input DistilBERT transformer model along with the causal, prefix transfer-learning based approach. By doing that we benefit from both tabular "numerical" and textual data information.

The objective of this baseline is to investigate whether employing the Big Five personality traits can improve the sequential recommendation performance compared to previous baselines.

Now that we have discussed the different baselines architectures, the next section highlights the different models' evaluation results.

5.3. Experiment and Results

This section discusses the used training dataset as well as both training and evaluation performance for each baseline.

5.3.1. Experiment Dataset

Experiments and results are done using the TG-Redial dataset [**8**]. We used this dataset to investigate the performance of our proposed models. In its original form, the TG-Redial dataset was designed for topic-guided dialogue systems. It consists of 10000 dialogues between an information seeker and a recommendation provider in the movie domain. Besides the dialogue information, this dataset contains specific details for the recommendation seekers such as their profile "preferences" and the list of their previous interaction movies. Table 5.1 highlights global statistics of the TG-Redial dataset.

Table 5.1. Data statistics of our TG-Redial dataset [**8**].

	Total		**Average**
Users	1,482	#Words per Utterance	19.0
Dialogues	10,000	# Topics per Dialogue	19.0
Utterances	129,392	# Movies per Dialogue	3
Movies	33,834	# Profile Sentences per User	10
Topics	2,571	# Watched Movies per User	202.7

This dataset consists of 1,482 different users and 33,834 movies, with an average of 10 profile sentences per user and 202.7 watched movies per user. Therefore, we filtered this dataset to keep only the users' profiles and their history of movies interactions to use them during the training phase.
Using this dataset we faced two main problems:

- The First problem is that this dataset is not annotated with users' Big Five personality traits. Therefore, to solve this problem we used our AWS-EP personality prediction model described in section 4.2 to infer the users' personalities from their sentences.
- The second problem is that the TG-Redial dataset is a Chinese dataset. Therefore, to translate this dataset into English language we used the google cloud translation *Application Programming Interface* (API).

Table 5.2 highlights an example of the filtered TG-Redial dataset. The historical interactions columns highlight the different movies IDs that the user previously interacted with.

Table 5.2. The training data examples.

User ID	User Profile	Personality Traits	Historical interactions
1	I like comedy very much I like famous actors I' am a happy person I like to make friends	Openness =33.12 Neuroticism =50.80 Agreeableness =63.12 Extraversion =12.14 Conscientiousness=20.72	@1437013,@123548, @1258967,@325641, @3654782,@245687,
2	I like love very much I like good scripts I' am easily moved I like music very much	Openness =87.10 Neuroticism =24.60 Agreeableness =19.15 Extraversion =90.1 Conscientiousness=66.13	@2547898,@214598, @3458452,@123554, @1236548,@2136547,

During the training process, the dataset was partitioned into three parts, training, validation, and testing subsets. The training set comprises 8328 unique data rows where each row defines a set of preference utterances (profile), the 5 personality trait values (agreeableness, openness, conscientiousness, extraversion and neuroticism) and a set of previously watched movies. The training, validation and testing subsets statistics, are defined in the following table 5.3.

Table 5.3. Experiment dataset statistics.

	Training subset	Testing subset	Validation subset
Unique rows	8324	733	733
Unique interaction	8324	733	733
Unique profile	1482	733	733
Unique personality	86	85	85
Unique movie	28515	15547	15673

In addition, table 5.4 highlights the model training hyperparameters,

Table 5.4. The training hyperparameters.

Hyper-Parameter	Value
Epochs number	20
Optimizer	Adam optimizer
Learning rate	2e-5
Weight decay	0.01
Training loss (CL)	Cross-Entropy loss
Batch size	5
Trainable parameters	186007752

The reasons for choosing the epochs number, the optimizer algorithm, the learning rate, the weight decay and the batch size hyperparameters are similar to those discussed in section 4.4.1. The next section highlights the training and evaluation results for each baseline.

5.3.2. Training and Evaluation Results

Different experiments are done to investigate the different generalization performances of the proposed baselines (DB-SR, DBT-SR, and DBTP-SR).

To evaluate the sequential recommendation baselines, we applied the leave-one-out evaluation procedure. This is a common task that is used widely for the next item recommendation problem evaluation [**121, 78, 81**]. For each user in the test set, we mask out the last interacted item and feed all of the previous items to the prediction model. We then evaluate the predictions using the *Hit Ratio* (HR@k) and the *Mean Reciprocal Rank* (MRR@k) metrics.

$$HR = \frac{|U^L_{hit}|}{|U_{all}|} \tag{5.3.1}$$

$$MRR = \frac{1}{|U_{all}|} \sum_{u=1}^{|U_{all}|} RR(u) \tag{5.3.2}$$

$$RR(u) = \sum_{i<L} \frac{relevance_i}{rank_i} \tag{5.3.3}$$

Where $RR(u)$ is the reciprocal rank of user u, which is defined by the sum of relevance score of top L items weighted by reciprocal rank. MRR is simply the mean of all users U_{all} in the test dataset.

We start evaluating our system using HR@k, and MRR@k metrics. To guarantee a fair comparison with state-of-the-art models, k was chosen to be k={1,3,5,10}. In our evaluation setup, We define the hit and the relevance as having the original masked item in the k predicted items. Table 5.5 highlights the performance of the different baselines using this evaluation setup. It is important to mention that for this chapter and all of the following chapters evaluation sections, the dash character is put when the original baseline of the paper does not report the value of that metric within their evaluation results.

Table 5.5. The different baselines performance on the hold-out "test" set.

	MRR@1	**MRR@3**	**MRR@5**	**MRR@10**	**HR@1**	**HR@3**	**HR@5**	**HR@10**
Popularity [**8**]	-	-	-	0.0011	-	-	-	-
ReDial [**8**]	-	-	-	0.0005	-	-	-	-
KBRD [**8**]	-	-	-	0.0040	-	-	-	-
GRU4Rec [**8**]	-	-	-	0.0014	-	-	-	-
SASRec [**8**]	-	-	-	0.0050	-	-	-	-
TextCNN [**8**]	-	-	-	0.0119	-	-	-	-
BERT [**8**]	-	-	-	0.0182	-	-	-	-
TG-Redial [**8**]	-	-	-	0.024	-	-	-	-
DB-SR	0.06957	0.08185	0.0858	0.0878	0.0695	0.0995	**0.1173**	0.1309
DBTP-SR	0.05729	0.0723	0.07653	0.07838	0.05729	0.0914	0.10914	0.1227
DBT-SR	**0.0736**	**0.08481**	**0.0884**	**0.0912**	**0.0736**	**0.09959**	0.1159	**0.1364**

Table 5.5 show that the DBT-SR model outperforms TG-Redial model on the MRR@10. The MRR@10 ranking value was increased form 0.024 units to 0.0912 units. This highlights the importance of employing the transfer learning procedure with LMs to achieve better ranking results. We only report the MRR@10 results, as within the original paper the authors reported both MRR@10 and MRR@50. However, using MRR@50 we were incapable of evaluating our model due to computation issues, as the available resources were insufficient to generate 50 tokens in a row. In addition, the performance of our proposed baselines on the different ranking metrics is shown in figure 5.4.

Figure 5.4 shows that using our proposed causal/prefix transfer learning-based approach, the DBT-SR model outperformed the standard MLM model "DB-SR" baseline on all MRR@k and HR@k metrics, exception for the HR@5 metric. This highlights the importance of using the transfer learning procedure to improve the recommendation performance. The promising results provided by the DBT-SR model highlights the effectiveness of our proposed causal/prefix approach compared to using the standard causal learning technique.

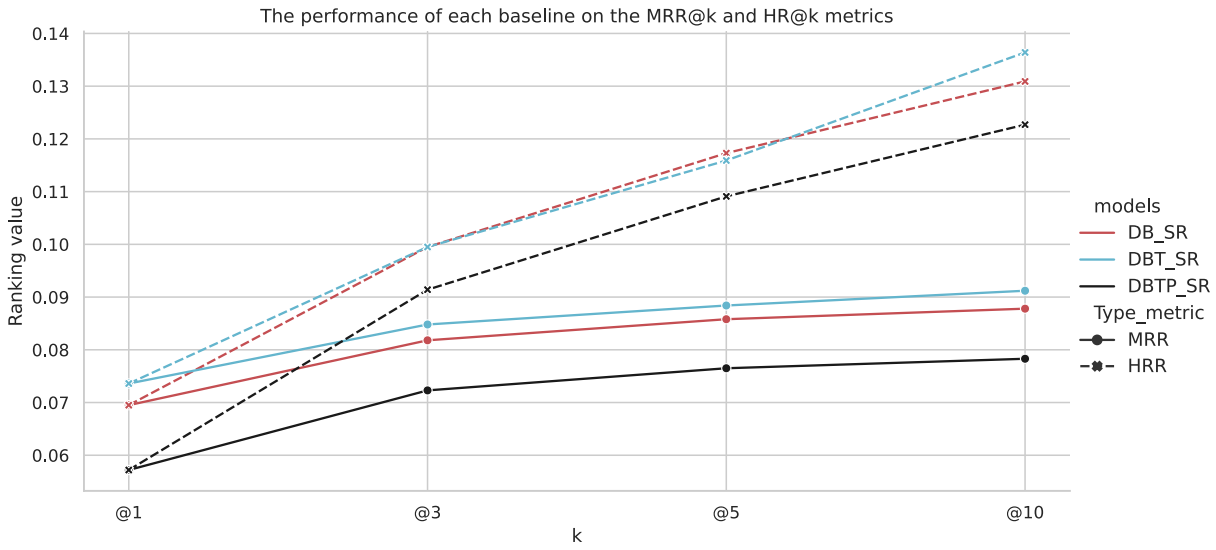

Fig. 5.4. The performance of each baseline on the MRR@k and HR@k metrics.

Moreover, Figure 5.4 shows that for the TG-Redial dataset sequential movie recommendation task, personality traits were incapable of improving the recommendation results. Such behaviour could be explained in different ways. We can argue that these poor results are related to the number of training epochs that we choose. Employing the personality traits means that we are employing very complex information to the model. Understanding this information may need more training epochs to build the relationship between the recommended movies and the correspondent personality traits. Another point of view to discuss these poor results is related to the gap between the MLP network "Big Five user Personality Encoder in figure 5.3" and the DistilBERT representation spaces. By combining the MLP-Big Five network representation vector and the DistilBERT representation vector, it is not guaranteed that both vectors will be within similar representation spaces. Thus, combining them prevents the model from producing good recommendation performances.

To summarize, experiments show that the causal/prefix transfer learning approach outperformed the state-of-the-art baselines on different metrics. Whereas including the personality traits within a range of 20 training step were incapable of providing effective performance for the TG-Redial sequential movie recommendation task. Although our DBT-SR model outperforms the TG-Redial state-of-the-art model, we can argue that the obtained results do not provide a fair comparison, as both models tackle the TG-Redial dataset movie recommendation problem but from different perspectives. The TG-Redial baseline tackles the session-based movie recommendation problem, Whereas our proposed model tackles the sequential-based movie recommendation task. However, as the TG-Redial dataset is still a

new dataset, to date there is no published study that tackles this dataset from a sequential recommendation perspective. Therefore, we compared it with the TG-Redial model.

5.4. Conclusion

In this chapter, we discussed the second contribution of our research "as shown in section 1.2" which consists of building a novel sequential movie recommendation model. We highlighted the effectiveness of using a transfer learning trick on top of a pre-trained DistilBERT model for the next movie to watch recommendation task. We also highlighted the impact of using individual personality traits for improving the sequential recommendation process. Empirical results show that using our proposed transfer learning trick outperformed state-of-the-art baselines for different MRR@k and HRR@k metrics. Also, during the evaluation process, we found that employing the users' personality traits does not improve the sequential recommendation process for the TG-Redial dataset. We argue that such a performance may be related to different reasons such as the training epoch numbers, and the gap between the representation spaces. Chapter 8 discusses different directions to improve our proposed sequential movie recommendation approach.

Now that we have discussed both our research's first and second objectives (discussed in chapters 4 and 5 respectively), in the following chapter, we discuss our third objective "as shown in section 1.2" by highlighting our proposed contribution for the Dialogue State/Topic Tracking component.

Chapter 6

The Dialogue State Tracking module: Elec-SP

In this chapter we introduce a simple but effective Dialogue State Tracking (DST) approach named *Elec-SP* "ELECTRA for State Prediction" built on top of a pre-trained ELECTRA transformer model to keep track of the topics discussed within a conversation session. We designed the conversation state tracking task as an intent, topic pair prediction problem, where we aim to keep track of the user's current intent within an utterance as well as the current conversation topic. This process is very important in the research pipeline, since knowing the different topics within the conversation helps the CRS to produce more topic-aware responses and recommendations that do not deviate from the general context of the conversation.
The rest of the chapter is structured as follows: Section 1 discusses the motivations behind our proposed solution. Section 2 highlights the task description and the architecture of our solution. Section 3 discusses the training and evaluation data as well as the pre-processing methods. Section 4 conducts the baselines experiments and results compared to state-of-the-art models. Finally, section 5 concludes the chapter and presents the next chapter.

6.1. Motivation and Contribution

Throughout this section, we highlight in detail our contribution and the motivation behind the third objective of our research (as shown in section 1.2), which consists of building an effective dialogue state tracking component.
Detecting dialogue states is very important in order to build coherent and engaging conversation recommendation systems that simulate natural human dialogues. Task-oriented CRSs also known as Goal-oriented CRSs, have gained a lot of attention in the past few years. These systems are capable of setting natural conversations to help users achieve simple tasks based on their intent (such as requesting movies recommendation, medicine suggestions, flight booking, etc.).
By knowing the current dialog state, these systems can effectively decide the next action to

take (for example, whether to recommend a movie or ask for more clarifications).
Learning the current dialogue state from natural language data sources to generate coherent and human-like responses has been one of the most challenging tasks in the NLP-based recommendation field. Modern dialogue response generation approaches usually formulate this task as a sequence-to-sequence problem, which may lead to out-of-context or repetitive responses generation [**122**]. Moreover, the natural human discussion may contain different topics for both open-domain and goal-oriented dialogues. Therefore, it is essential to design a model that effectively predicts the current state of a conversation session.

Motivated by the importance of dialogue state tracking in improving the conversational recommendation system performance [**4**], in this chapter, we propose a simple but effective learning approach to keep track of the current state within a dialogue session. We design the conversation state tracking as a multi-label classification problem where we aim to predict the topic and intent tokens from the state vocabulary. To do this, we fine-tuned the pre-trained ELECTRA model using a classification MLP network. It is important to highlight that by training the classification network on top of the Electra model, we are not only fine-tuning the hyperparameters of the pre-trained model, but we are also training the MLP network hyperparameters. We also investigate the impact of employing individual personality traits to keep track of the conversation current state, as well as the use of the pre-trained ELECTRA model compared to other pre-trained models such as BERT.
Experiments and results show that our proposed solution outperforms state-of-the-art topic prediction models on the TG-Redial dataset. To the extent of our knowledge, this is the first work that employs the ELECTRA model for the conversation state tracking task on the TG-Redial dataset.

It is important to highlight that our main contribution to this CRS component is not proposing a new learning architecture. However, the contribution of this chapter is reformulating the input data into a textual structural input by introducing special tokens and additional insights to improve the prediction task. We also investigate the impact of employing the Big Five personality traits for this task. The next section discusses our contribution task description as well as our proposed solution architecture.

6.2. Task Description and Model Architecture

Throughout this section, we discuss the formulation of the dialogue state tracking problem and our proposed solution. Our architecture is built on top of the ELECTRA model. More information about the ELECTRA model is provided in appendix section A.3.1.

6.2.1. Task Setup

In a dialogue-based system, a conversation topic can be represented by using a one-word token (for example, "Earth" when the user expresses an utterance about Earth) or a set of words (for example, "Tom", "Cruise" when the user expresses an utterance about the actor Tom Cruise). Also, the user's current intent within a conversation can be represented by a set of words (for example, "request", "recommendation" when the users express within the utterance that they want to watch a movie) or a single word (for example, "talk" when they just want to chit-chat). Therefore, in order to model both the user's intent and the conversation topic, we designed the dialogue state prediction problem as a supervised multi-label classification task where topic and intent words are considered as ground truth labels, and dialogue utterances are considered as system input features.
The reason behind considering ground-truth labels as a union of both intent and topic tokens is due to our assumption that both types of information are important in order to improve the performance of our system. For example, if the user provides an utterance such as "I enjoy watching the sunset", it is obvious that the utterance topic is "sunset". Therefore, a good state prediction model will be capable of predicting "sunset" as the inferred state. However, just knowing the topic information does not help our system choose the best action to take, as it does not understand whether the user wants to talk about a sunset, or if they are requesting a movie about a sunset. To improve the CRS action decision, we argue that tracking the user's intent while also tracking the conversation topic can significantly improve the decision-making of our system. Therefore, we designed our state (intent, topic) prediction task as follows:
At time $t \in \{1..n\}$, given an input $X_t = [x_1,..,x_{t-1},x_t]$, which is a concatenation of the previous dialogue information $[x_1,..,x_{t-1}]$, and the current utterance x_t, the main objective of our prediction model is to maximize the probability of predicting the current dialogue state labels $Y_t \subset Y$, where $Y = E \cup T$ defines the target vocabulary set, which is a union of both intent tokens and topic tokens vocabularies, where $E = \{e_1,e_2,..,e_{|E|}\}$ is the intent vocabulary, and where $T = \{t_1,t_2,..,t_{|T|}\}$ is the topic tokens vocabulary. Therefore, $Y_t \subset \{y_1,y_2,..,y_{|Y|}\}$ where $\{y_1,y_2,..,y_{|Y|}\} = \{e_1,e_2,..,e_{|E|}\} \cup \{t_1,t_2,..,t_{|T|}\}$ and $|Y| = |E| + |T|$.

Our optimization objective for a specific utterance at time t is defined as:

$$max P(Y_t|X_t) = min - P(Y_t|X_t) \tag{6.2.1}$$

where

$$P(Y_t|X_t) = \prod_{y \in Y_t} P(y|X_t) \tag{6.2.2}$$

where y is the label word from the set of ground truth labels at time t, and X_t represents all of the conversation information at time t which is structured as follows:

$$X_t = r_1,u_1...,r_{t-1},u_{t-1}, < topic > ,E_{t-1},T_{t-1}, < /topic > ,r_t,u_t, < topic > \tag{6.2.3}$$

.

Where $r_t \in R = \{< seeker > , < recommender >\}$ defines the role of the conversation speaker at time t. $u_t \in U = \{u_1,u_2,...u_t\}$, which defines the utterance of the speaker at time t, where an utterance is a collection of words $u_t = \{w_1,w_2,...w_{|u_t|}\}$. "<topic>" and "</topic>" are special tokens that indicate the start and the end of the state tokens. $E_t \subset E$ is the set of intent tokens at time t. $T_t \subset T$ is the set of topic tokens at time t (an example of the X_t input shape is defined in table 6.1).

As we have already discussed, we designed the state prediction task as a multi-label classification task. Therefore, to optimize the learning process, we defined the objective function as minimizing the binary cross-entropy loss:

$$LOSS_{topic} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{S}\sum_{j=1}^{S} y_{ij} \cdot \log(\hat{y_{ij}}) + (1 - y_{ij}) \cdot (1 - \log(\hat{y_{ij}})) \tag{6.2.4}$$

where N defines the data size, S defines the Y set vocabulary size, y_{ij} defines the i^{eth} instance and j^{eth} label word ground-truth value {0,1}, and $\hat{y}_{ij}$ defines the i^{eth} instance and j^{eth} label word predicted value {0,1}

This model is trained under the objective of minimizing the $LOSS_w$, where X defines the training data and θ_{topic} defines the Elec-SP model learning parameters.

$$LOSS_w = min_{\theta_{topic}} \sum_{x \in X} LOSS_{topic}(x,\theta_{topic}) \tag{6.2.5}$$

Experiments and results show that including previous dialogue information (previous utterances and previous topic/intent) improves the effectiveness of our prediction model on different metrics such as @HR1, @HR2, @HR3, micro-precision score, micro-recall score, micro-f1 score, and the accuracy score. More details about the experiments are highlighted in section 6.4. The next section highlights our proposed solution for predicting the intent and topic information.

6.2.2. Model Architecture

Given the user's previous utterances, roles and intents/topics tokens, our model learns to predict both the current topic and intent as the current conversation state. Figure 6.1

highlights the main architecture of our Elec-SP component.

Our proposed solution combines a pre-trained transformer and an MLP network. We use

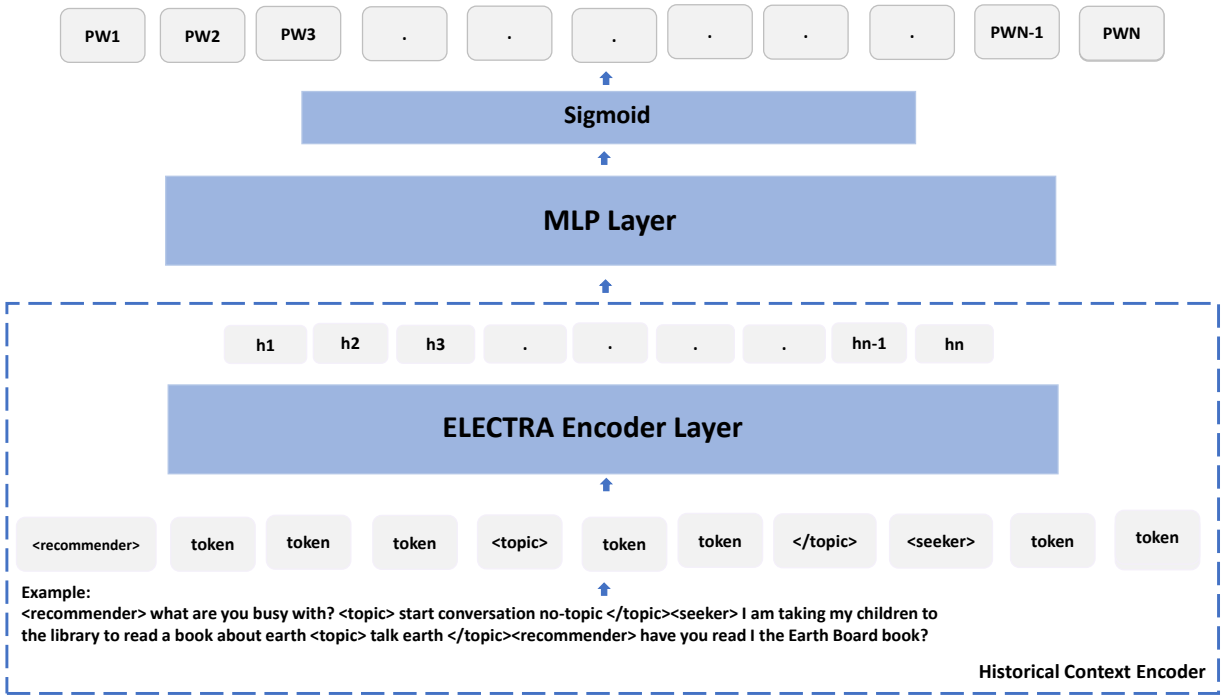

Fig. 6.1. Elec-SP model architecture.

the MLP network to fine-tune the pre-trained model on the topic prediction task. Our addition compared to state-of-the-art models is the use of the ELECTRA encoder layers instead of using BERT. As most transformer-based topic prediction solutions use BERT as their main pre-trained model [**8, 9**], we aim to investigate the impact of using ELECTRA rather than BERT to improve the topic prediction performance.

First, the textual historical information is tokenized using the pre-trained tokenizer of ELECTRA . Given these different tokens, the ELECTRA Encoder Layer encodes all of the conversation information in order to produce a new contextual vector representation for the input denoted as [h1,..,hn]. The [h1,..,hn] vector is a representation of the user's context information in the ELECTRA neural network space, which employs previous dialogue utterances and conversation state information. This vector is then used as an input to the MLP component to predict the next state (topic/intent tokens) using a sigmoid-based multi-label classification layer. The sigmoid-based layer is responsible for predicting a probability value for each word "class" in the vocabulary set denoted as Pw_1, Pw_2,.., Pw_n where Pw_n represents the probability likelihood of predicting the $word_n$ "$class_n$" as the state representation token. Only the predicted classes that have more than 0.5 probabilities are considered to be convenient predictions for representing the conversation state.

Now that we have defined our task setup and our architecture design for the CRS state tracking component, the next section highlights the data preparation and filtering process that we applied to the TG-Redial dataset in order to train and evaluate our proposed solution.

6.3. TG-Redial Dataset Pre-procesing for the DST Task

In order to train our proposed solution we used the TG-Redial topic-guided conversation dataset for the movie domain [**8**] (section 5.3.1 highlights the structure of this dataset). We aim to benefit from the topic labeled utterances within this dataset to build our state tracking model. For each conversation, we took the utterance and the topic/intent information related to that utterance. Then we prepared our data to encode the information in a way that for each current utterance we concatenate its information with all previous utterances and topics/intents using special tokens "<topic>, </topic>" as separators (as shown in equation 6.2.3) and then we place the current topic/intent as the label tokens. Table 6.1 highlights an example of the final filtered dataset structure for the state prediction task.

Table 6.1. The training data examples.

Labels: Intent/Topic	**Personality Traits**	**Historical utterances**
Intent: start conversation **Topic**: no-topic	Openness =33.12 Neuroticism =50.80 Agreeableness =63.12 Extraversion =12.14 Conscientiousness=20.72	<recommender> what are you busy with
Intent: talk **Topic**: earth	Openness =33.12 Neuroticism =50.80 Agreeableness =63.12 Extraversion =12.14 Conscientiousness=20.72	<recommender> what are you busy with ? <topic> start conversation no-topic </topic> <seeker> i am taking my children to read a textbook about the knowledge of the earth

Both personality traits and historical utterances information are used as features to train the learning models and the intent/topic column is the label column.
To add on, we use special tokens such as "<topic>, </topic>", indicating the start and the end of topic/intent tokens, where "<recommender>" and, "<seeker>" tokens are used to define the role of the utterance speaker. We add these specific tokens to make the ELECTRA encoder benefit from the structure information within the input text. The next section highlights the experiments that were conducted to evaluate our model as well as the obtained results.

6.4. Experiments and Results

Different experiments are done to investigate the performances of our proposed solution and compare it to state-of-the-art models on different metrics. In addition, to investigate the impact of different data properties on our system (for example, the user personality properties) we designed three other different baselines named respectively PElec-SP, Elec-SP w/o, Elec-SP w/o H. The different objectives of the proposed baselines are defined as follows:

- **PElec-SP**: The objective of this model is to investigate the impact of employing the individual Big Five personality traits for the state prediction task (figure A.7 in appendix section highlights more details about the PElec-SP model architecture).
- **Elec-SP w/o**: The intuition of this baseline is to investigate the impact of not employing intent tokens "only consider the topic tokens as the ground truth state information".
- **Elec-SP w/o H**: The purpose of designing this baseline is to investigate the impact of not employing the previous utterances and topics in the input text as well as the impact of considering the current utterance text information as the only feature for the state prediction task.

To keep the fairness of the evaluation process we used the same training, testing and validation subsets published by the original authors that have been used for training and evaluating the state-of-the-art baselines. Moreover, our model was trained with a $1.5e^{-4}$ learning rate (similar to the state-of-the-art model setup [**8, 9**]), and for 30 epochs. Also, for the ELECTRA pre-trained model, the weights were updated "fine-tuned" during the training epochs of each model. During the experiment process, the training set comprises 109892 unique data rows, the test set comprises 9748 unique rows, and the validation set comprises 9764 unique rows. Each row defines a combination of the current utterance, the set of previous utterances, the previous topics and the previous intents. The current topic/intent tokens are defined as labels (as shown in table 6.1).
The reasons for choosing the epochs number, the optimizer algorithm, the learning rate, the weight decay and the batch size hyperparameters are similar to those discussed in section 4.4.1.

6.4.1. Evaluation Results

We start evaluating our Elec-SP model using HR@k metric (as shown in equation 5.3.1) as it is the main metric used in the state-of-the-art TG-Redial dataset papers to evaluate the state-prediction performance. In order to guarantee a fair comparison, k was chosen to be k={1,3,5} as the original paper reported only the evaluation results for k=1, k=3 and

k=5 [**8**]. We defined the hit and the relevance as having the original label tokens in the model k predicted tokens. Table 6.2, highlights the performance of our model compared to different state-of-the-art baselines using the above evaluation setup.

Table 6.2. The different baselines performance on the test set.

	HR@1	**HR@3**	**HR@5**	**accuracy**	**M-prcision**	**M-recall**	**M-f1**
PMI [**8**]	0.0349	0.0927	0.129	**-**	-	-	-
MGCG [**8**]	0.6098	0.8128	0.8294	**-**	-	-	-
Conversation-BERT [**8**]	0.6114	0.8189	0.8341	**-**	-	-	-
Topic-BERT [**8**]	0.6155	0.8275	0.8405	**-**	-	-	-
Profile-BERT [**8**]	0.4986	0.8205	0.8344	**-**	-	-	-
TG-baseline [**8**]	0.6231	0.8370	0.8497	**-**	-	-	-
Topic-refine [**9**]	0.7651	0.8023	0.81889	**-**	-	-	-
Elec-SP w/o H	0.754	0.858	0.877	0.69562	0.9428	0.877	0.9087
Elec-SP w/o	0.8325	0.9017	0.9146	**0.8037**	0.9283	0.8248	0.8735
PElec-SP	0.7728	0.8575	0.8756	0.7323	**0.9703**	0.8899	0.9284
Elec-SP Ours	**0.8359**	**0.9070**	**0.9191**	0.7983	0.9692	**0.9213**	**0.9446**

Table 6.2 shows that not employing the intent tokens for the state prediction task "Elec-SP w/o" and not considering previous conversation information "Elec-SP w/o H" provides poor ranking results compared to the Elec-SP model. Therefore, it is obvious that knowing the previous user's utterance intents as well as the previous conversation context significantly improves the effectiveness of predicting the right conversation state. Besides the promising results on the ranking metrics (@HR1, @HR3, and @HR5), our model also provides promising results for the classification metrics, such as 0.79 accuracy value, 0.96 micro-precision value and 0.92 micro-recall value. Moreover, our solution outperforms all of the exiting local baselines on the micro F1 score with a 0.94 value. The reason why we did not compare our classification results compared to state-of-the-art baselines is that the original authors of these papers did not report the classification evaluation metrics (which explain the dash symbol in table 6.2). However, in this research, we report these classification evaluation values to demonstrate the effective improvement of the Elec-SP model compared to Elec-SP w/o, Elec-SP w/o H, and PElec-SP local baselines on the accuracy, M-precision, M-recall, and M-f1 metrics. In addition figure 6.2 shows that our proposed Elec-SP model overcome state-of-the-art-baselines (TG-baseline [**8**], Topic-refine baseline [**9**], etc.) on the HR@1 metric.

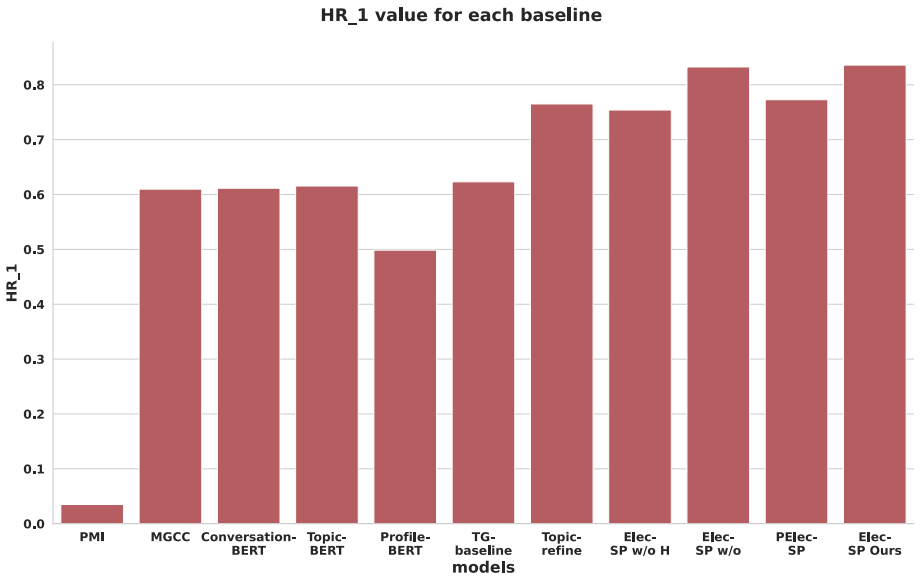

Fig. 6.2. The performance of our proposed baselines compared to state-of-the-art-models on the HR@1 metric.

Also, figure 6.3 shows that our proposed solution outperforms state-of-the-art-baselines on the HR@3 metric.

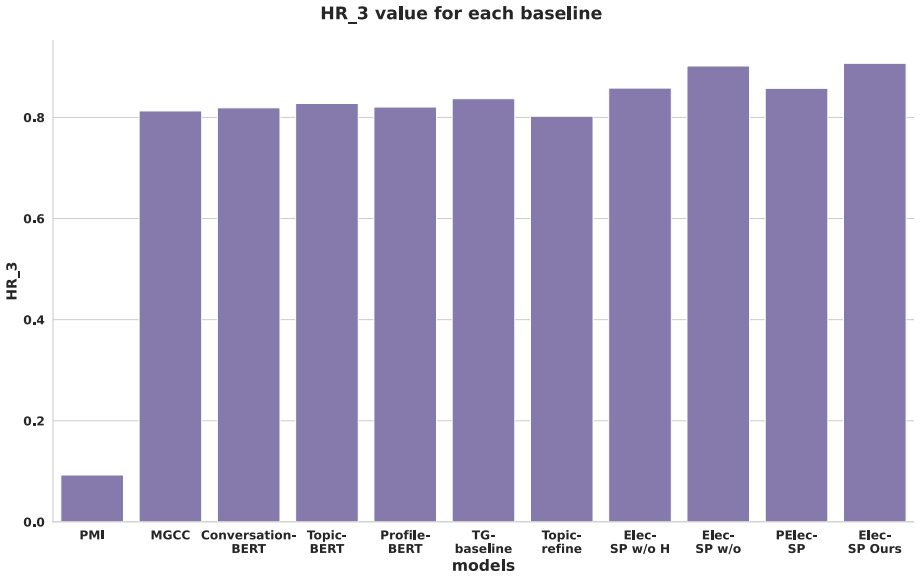

Fig. 6.3. The performance of our proposed baselines compared to state-of-the-art-models on the HR@3 metric.

In addition, figure 6.4 shows that our proposed solution provides a significant performance on the HR@5 metric compared to existing models.

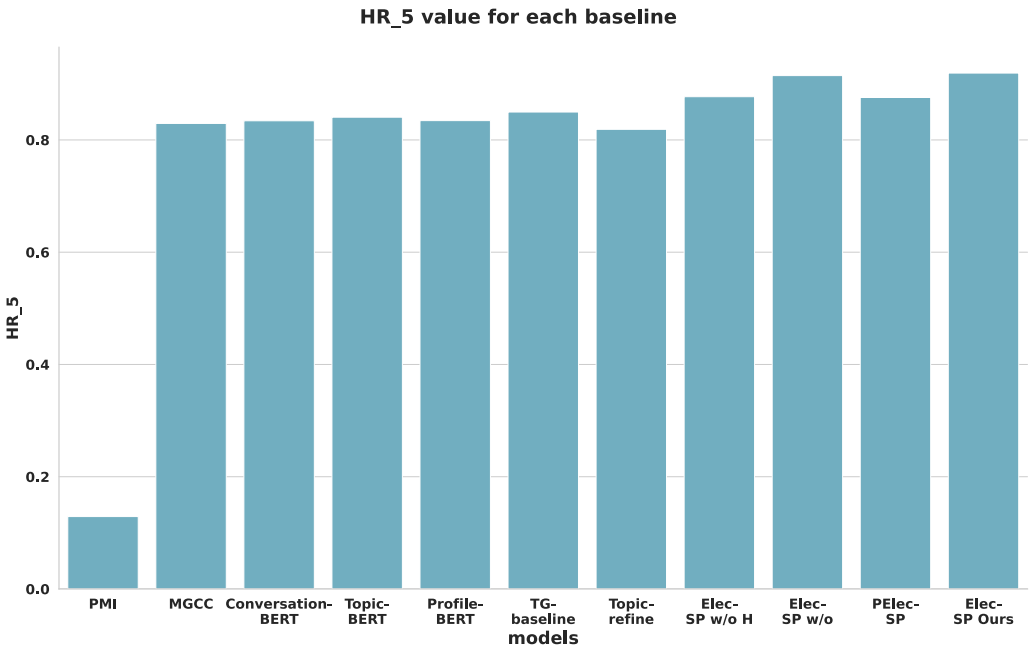

Fig. 6.4. The performance of our proposed baselines compared to state-of-the-art-models on the HR@3 metric.

This highlights the importance of using the ELECTRA pre-trained language model for the classification-based tasks.

Experiments (figures 6.2, 6.3 and 6.4, and table 6.2) show that including individual personality traits does not help increase the effectiveness of our predictive model on all metrics, except for the precision metric, which can be explained in two ways.
The first explanation could be that the dialogue context is sufficient to predict the state of the conversation, as predicting whether the user is talking about a specific topic (for example, sunrise or books) does not explicitly need the user's personality traits. It only needs the context within the user utterances to know what the user is talking about.
The second explanation could be related to the insufficient training steps, or to the gap between the Big Five traits representations provided by the MLP network and the ELECTRA LM representation space. It could be related also to the information combination technique that we used to combine both conversation context and personality traits, where we used a simple vector summation technique to combine both details which could be insufficient for the model to make the relation between the personality and the topic prediction task

especially when the two representation spaces are different from each other.

6.5. Conclusion

In this chapter, we have discussed the third contribution of our research (as shown in section 1.2) which consists of using a pre-trained ELECTRA model for the conversation state tracking component as a topic/intent prediction task. We showed that including previous intent information can help improve the effectiveness of the state prediction task. However, including the individual Big Five personality traits did not improve the dialogue state prediction performances, except for the precision metric. Also, experiments show that the use of the ELECTRA model outperforms state-of-the-art models on the TG-Redial dataset (i.e., BERT).

Despite the promising results of our proposed solution, we believe that this approach can be furthermore optimized. Different future improvements are explained in chapter 8.

Now that we have discussed our research's first, second, and third objectives (discussed in chapters 4, 5, and 6 respectively), in the following chapter, we discuss our fourth contribution (as shown in section 1.2) by highlighting our proposed Dialogue Generation approach.

Chapter 7

The Dialog Generation module: PPPG-DialoGPT

This chapter highlights the response generation component (NLG component), which aims to generate a *contextual-aware* response to fulfill the user's request. The generated response can also employ a *session-based movie recommendation* by using a manually designed *prompting template* and a fine-tuned *autoregressive* LM approach. This model efficiently and effectively provides a personalized response generation by prompting the pre-acquired knowledge within the *DialoGPT* language model. We also report the impacts of using individuals' *personality traits* to improve the language model performance on both movie recommendations and dialogue generation tasks.

This process is very important in the research pipeline since the success of this phase helps us to establish coherent and personalized conversation sessions, as well as provide session-based and context-aware movie recommendations. The rest of this chapter is structured as follows: Section 1 discusses the motivation behind our proposed solution. Section 2 highlights the employment of prompting templates and the architecture of our proposed method. Section 3 discusses the training/evaluation data and the pre-processing steps. Section 4 conducts the experiments and results by comparing them to state-of-the-art baselines. Finally, section 5 concludes this chapter.

7.1. Motivation and Contribution

Throughout this section, we highlight in detail our contribution and the motivation behind the fourth objective of our research (as shown in section 1.2) which consists of building a novel personality-aware dialogue generation module using the prompt-based learning mechanism.

Dialogue generation approaches are multi-turn dialogue exchange systems between two speakers. Goal-oriented conversational recommendation systems are types of dialogue generation approach that provides context-aware recommendations within dialogue sessions. Autoregressive-based goal-oriented CRSs are types of system applications that designs the session-based recommendation task as a *next word generation problem*. During the learning phase of the autoregressive-based CRS, the recommended items are employed within the dialogue turns as normal words. The language model then tries to understand how to generate a response that employs the convenient recommendation by maximizing the likelihood of predicting the right item to recommend as a next word generation problem. Despite the recommendation task, these systems also provide an *open-domain* chitchat behaviour to better understand the current user's preferences and personalize the generated response.

Previous research employ fixed template responses to provide a coherent recommendation to the user where the question and the response templates are pre-fixed in advance [**46**]. Despite the effectiveness of these approaches in terms of providing coherent item recommendations, they still lack the human natural interaction behaviour, as during the conversation session the user will be receiving the same text response in each turn, and only the recommended item will be changed. They are also limited in terms of understanding the user's current preferences as most fixed-template approaches allow only yes/no responses which do not reveal the full user's preferences [**46**].

To better understand the current preferences, recent CRS systems employ the *attention mechanism* to build a more contextual-aware understanding. Modern CRS approaches fine-tune pre-trained transformer models to generate specific and natural conversation utterances [**8, 9**]. GPT-2 and BERT models are the most two popular transformer architectures used for the *topic-guided* conversation recommendation task [**8**]. Despite the significant improvements provided by these models on the goal-oriented dialogue task, they still lack knowledge of the conversation structure. These models are originally pre-trained on open-domain datasets (books scripts, movies scripts, open-domain internet information etc.). They were only trained on large unstructured textual data under the autoregressive objective. Therefore, they do not encode the conversation structure within their parameters. Moreover, recent topic-guided CRSs formulate the conversation recommendation task as a multi-objective learning problem [**9**]. They fine-tune a group of pre-trained models under different objectives to produce a topic-aware generated response. For example, for the response generation model, we can find a model that predicts the current utterance topic, and a model that generates a response given the predicted topic (this is similar to Hongru Wang *et al.* study [**9**]). The learning objective in this case is minimizing both model errors. Although multi-objective networks are shown to be very effective in understanding the contextual information within the user's preferences in many tasks, it is also shown that

fine-tuning such large pre-trained models by adding other networks on top of them to fulfill the multi-objective end-goal, can be computationally expensive and needs a lot of resources, which are not always available especially for the academic research.
Moreover, despite the significant increase of performance provided by employing personality traits within different NLP tasks [**123**], previous published CRS research does not employ individuals personality traits within the conversational-based recommendation process [**5**], which can provide less accurate and poor personalized response generation.
We believe that not employing these traits, makes the existing CRS approaches incapable of effectively building personalized dialogue sessions. Also, with the recent advancements in the prompt-based learning research field [**124**], we strongly believe that employing both the prompt-based learning paradigm and the user's personality traits can significantly improve both CRS efficiency and effectiveness performance.

Therefore, to address the discussed CRS limitations, we propose the first prompt-based, personality and preference-aware DialoGPT [**125**] model named *PPPG-DialoGPT* "Personality and Preference-aware Prompt-based Goal-oriented DialoGPT model" (appendix section A.6.2 highlights the DialoGPT working mechanism). Our work benefits from the DialoGPT pre-trained model knowledge, which pre-encodes the dialogue structure information within its trained parameters to provide more natural conversation responses. Moreover, we use the user's preferences and it's Big Five personality traits as prompting controls to better understand the user's behaviour and generate more personalized responses and movie recommendation. We also employ the conversation topics information as prompting controls to guide the response generation task.
To the extent of our knowledge, this is the first research that uses prompt-based learning and personality traits as controls for the CRS problem. Also, it is the first work that uses DialoGPT on the TG-Redial dataset. Experiments and results show that the use of the users' personality traits yields an improvement for both recommendation, and response generation tasks on the TG-Redial movie-based dialogue dataset. Moreover, our prompt-based approach overcomes several competitive baselines on both effectiveness and efficiency aspects, thus confirming the validity of this work.
To summarize, in this chapter our main contributions compared to existent CRS approaches are as follows:

- The employment of the users' personality traits for the conversation recommendation task, and proposing the first Big Five personality-aware CRS.

- Prompting the knowledge of the DialoGPT model for the personalized CRS task and proposing the first prompt-based learning CRS.

- Designing different prompting templates to combine the control information with the conversation utterances.

- Evaluating our proposed CRS framework on different recommendation and response generation metrics

- Providing similar and better generalization performances for the dialogue generation and movie recommendation tasks compared to state-of-the-art models using a more efficient and less complex approach.

The next section highlights our proposed PPPG-DialoGPT approach and the different prompt templates designing methodology that we used to reach our personalized CRS task.

7.2. The PPPG-DialoGPT Model Description

Throughout this section, we discuss our proposed PPPG-DialoGPT architecture. We start by stating the problem, then we highlight the designing process of the prompting templates. Finally, we discuss the full architecture design of our proposed approach.

7.2.1. Problem Statement

We study the dialog-based recommendation task as an autoregressive language generation problem where the probability of generating the next word at time t "$w_t \in V$" depending on all of the previously generated words "$w_{1:t-1}$" is defined as $p(w_t|w_{1:t-1})$, where V defines the language model vocabulary.
In our dialogue generation framework, we consider the dialog-based recommendation process as a controllable language generation task where the previous conditional generation probability is modified to include not only the previous words, but also a set of controls named as $C = \{S,R,T,P,B\}$ where S is the set of utterances, R is the set of the speakers role, T is the set of topics mentioned in the conversation dialogue, B=$\{b_1,b_2,b_3,b_4,b_5\}$ is the user Big Five personality trait tokens and P=$\{p_1..p_n\}$ represents the different words that defines the user's preferences "profile".

Conditioning on the set of controls information, the probability framework for generating the *next word* a time t can be designed as follows:

$$p(w_t|w_{1:t-1},C_{1:t-1}) = p(w_t|w_{1:t-1},S_{1:t-1},R_{1:t-1},T_{1:t-1},P,B) \tag{7.2.1}$$

where w_t is the current word to be generated at time t and $w_{1:t-1}$ are the previously generated words within the same utterance at time t, $S_{1:t-1}$ is the set of previous utterances within the conversation session, $R_{1:t-1}$ highlights the sequence of speakers role for the previous utterances, and $T_{1:t-1}$ is the sequence of preceding topics mentioned within the dialogue.
In a general way, the conditional probability of generating the *next target sentence* at time t "$x_t = \{w_t^1, ..., w_t^{|x_t|}\}$" given the previous sentences $S_{1:t-1} = \{s_1..s_{t-1}\}$ where $s_1 = \{w_1^1..w_1^{|s_1|}\}$, can be described as follows:

$$p(x_t|S_{1:\{t-1\}},R_{1:\{t-1\}},T_{1:\{t-1\}},P,B) = p(x_t|s_1,s_2...s_{t-1},r_1,r_2...r_{t-1},t_1,t_2...t_{t-1},P,b_1..b_5) \quad (7.2.2)$$

In a more detailed way this probability can be written as the product of conditional probabilities:

$$p(x_t|S_{1:\{t-1\}},R_{1:\{t-1\}},T_{1:\{t-1\}},P,B) = \prod_{i=1}^{|x_t|} p(w_t^i|S_{1:t-1},R_{1:t-1},T_{1:t-1},P,B,w_t^{1:i-1}) \quad (7.2.3)$$

where w_t^i is the current word to be generated at utterance t and position i, and $w_t^{1:i-1}$ are all the previously generated words within the utterance t.
Approximating this type of probability is very hard and computationally expensive as we are conditioning on too much information by multiplying different probabilities which requires a large complex neural network to effectively learn a good approximation function. Especially when we have a multi-objective model where this probability can be divided into different parts in which each part will be approximated by a different neural network. However, using the prompt-based learning paradigm, we can simplify this problem definition to just modeling the probability p(x). We reformulate the task of maximizing $p(x_t|S_{1:\{t-1\}},R_{1:\{t-1\}},T_{1:\{t-1\}},P,B)$ to just maximizing the input probability $p(x)$, where x is a concatenation of all the $x_{1:\{t-1\}}$, $S_{1:\{t-1\}}$, $R_{1:\{t-1\}}$, $T_{1:\{t-1\}}$, P, and B information.
Instead of conditioning on a different part of information and using different networks to understand each part, the prompt-based learning paradigm allows us to concatenate all of the information within the same input using a specific template where all types of information are considered as words. Then it is the learning model's responsibility to learn them in an autoregressive way. Therefore, our task setup for the probability of generating the target sentence at time t becomes as follows:

$$\prod_{i=1}^{N} p(w_t^i|S_{1:t-1},R_{1:t-1},T_{1:t-1},P,B,w_t^{1:i-1})_{standard_{framework}} = \prod_{i=1}^{N} p(w_t^i|w_t^{1:\{i-1\}})_{prompt-based_{framework}} \quad (7.2.4)$$

To concatenate the control information within the same input, we used different prompting templates. The next section highlights the designing procedure of these templates.

7.2.2. Prompting Templates

As we have already mentioned, prompting is just about changing the input form by adding some additional information to it (known as controls), which improves the final-goal task. Throughout this research, we manually designed different prompting templates and evaluated their performances on the conversation recommendation task. Table 7.1 highlights our manually designed templates and their objectives.

Table 7.1. The prompting templates.

Prompt Name	Template	Objective
Promptless	[X][Z]	This prompt was designed as a baseline prompt.
T-prompt	$[X] < topic > t_1,t_2,..,t_n < /topic >< role > [Z]$	We designed this prompt to evaluate the importance of adding topic information for the dialogue generation process compared to the Promptless template.
Pro-prompt	$< profile > p_1, p_2,..,p_n < /profile > [X]$ $< topic > t_1,t_2,..,t_n < /topic >< role > [Z]$	We designed this prompt to evaluate the impact of adding the user's profile and it's preferences as controls for the dialogue generation process.
Pre-prompt	$< big5 > per_1, per_2,..,per_5 < /big5 > [X]$ $< topic > t_1,t_2,..,t_n < /topic >< role > [Z]$	We designed this prompt to evaluate the importance of adding the user's Big Five personality traits as controls for the dialogue generation process.
Pre-pro-prompt	$< big5 > per_1, per_2,..,per_5 < /big5 >$ $< profile > p_1, p_2,..,p_n < /profile > [X]$ $< topic > t_1,t_2,..,t_n < /topic >< role > [Z]$	We designed this prompt to investigate the impact of employing both the user's Big Five personality traits and it's preferences on our model performance

In our proposed templates, "[X]" is the original input that defines the previous conversation history. "[Z]" is the next utterance response given the original input "[X]", "<topic>" and "</topic>" are special tokens that define the start and the end of the topic information section. "t1, t2, .., tn" highlight the previous utterances topic information (the topic tokens), "<role>"=$\{< recommender >, < seeker >\}$ is defined to make the model aware of the speaker's role of the next generated utterance "[Z]", "<profile>" and "</profile>" are special tokens that define the start and the end of the users' profile information, where "p1, p2, .., pn" highlights the users' profile and preferences, "<big5>" and "</big5>" are special tokens that define the start and the end of the user Big Five personality traits, where "per1, per2, .., per5" highlights the users' Big Five personality traits values. The performance of prompting the DialoGPT model knowledge using these different templates is discussed in the experiments section 7.4.

7.2.3. The Model Architecture

In order to benefit from the users' personality traits, their preferences, and the previous conversation information during the dialogue generation process, we designed our input utterances using the *Pre-pro-prompt* template. Figure 7.1 highlights our PPPG-DialoGPT model architecture design. Where N defines the vocabulary size, w_j defines the j^{eth} word in

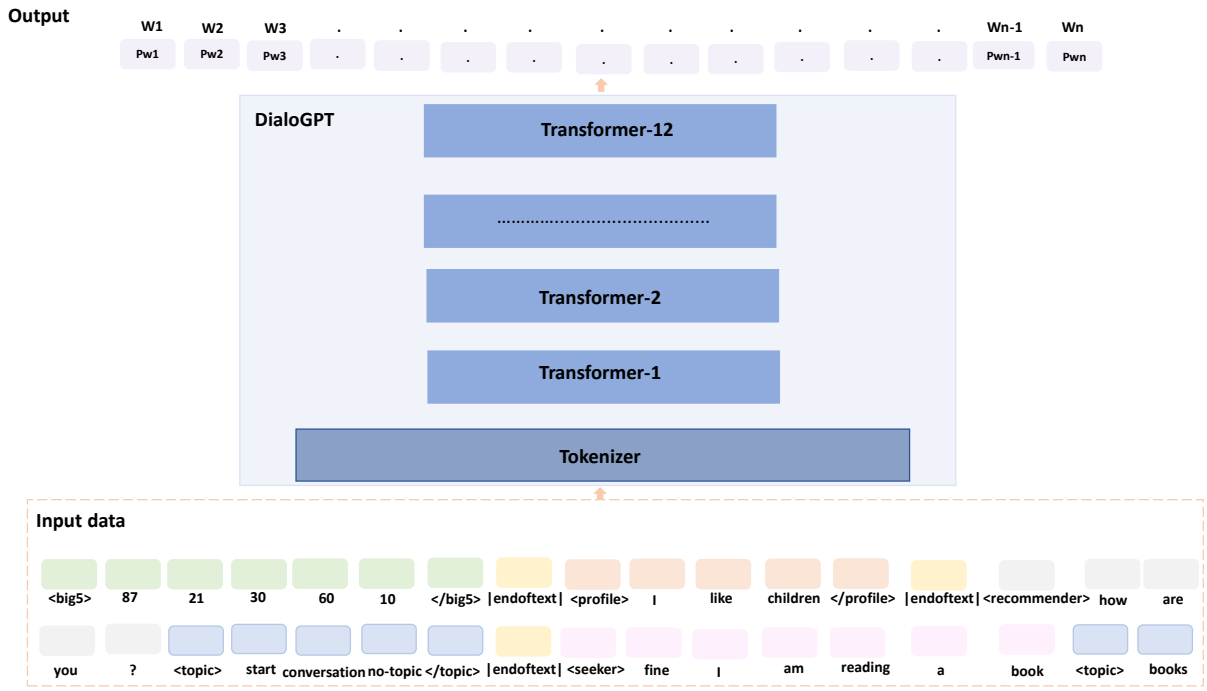

Fig. 7.1. PPPG-DialoGPT model architecture.

the vocabulary set where j $\in \{1..N\}$ and Pw_j defines the probability of generating the word j given the input data "all previous words".

Given the prompted input x, the words within the conversation session are tokenized using the DialoGPT pre-trained tokenizer. This tokenized vector is then used to train the DialoGPT model in an autoregressive way by maximizing the probability p(x) of generating the input x where at each step t, the model tries to maximize the probability $p(w_j|w_1..w_{t-1})$ of generating the word w_j as the next word by minimizing the CrossEntropy loss function. Then, at inference time, we use the probability logits Pw to get the most likely word as the next output token.

7.3. Dataset and Data Preparation

To train our model we used the recently published conversational recommendation TG-Redial dataset [**8**].

As we have already mentioned in the previous chapters (chapter 5), this dataset is a movie-based conversational recommendation dataset. It includes different information such as the user profile, his previous movie interactions, the conversation utterances and the dialogue topics to ensure a coherent dialogue and a personalized recommendation. Figure 7.2 highlights more insights into this dataset.

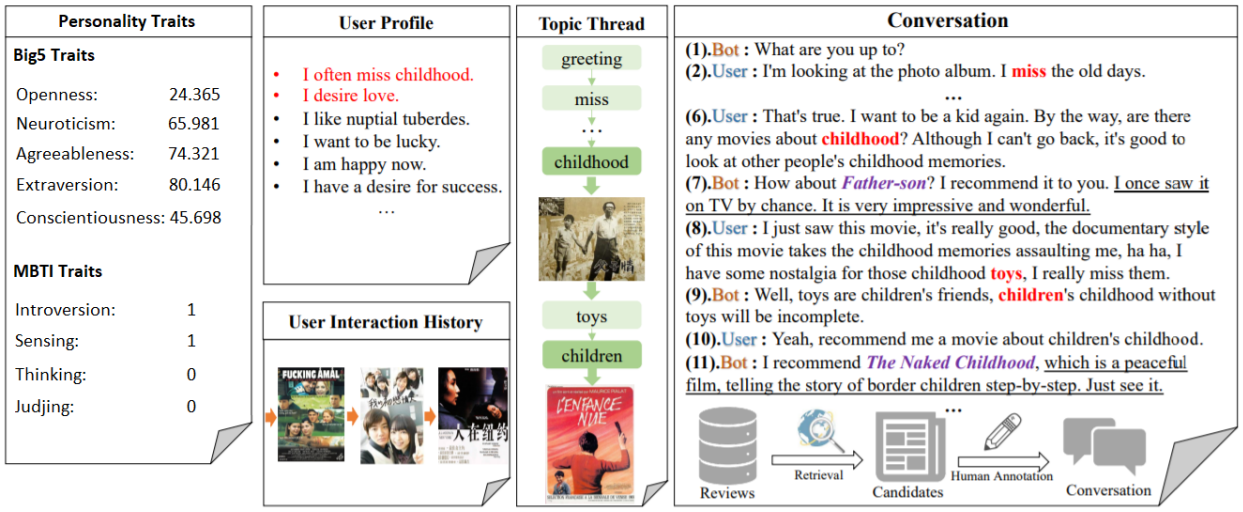

Fig. 7.2. An illustrative example of the TG-Redial dataset [**8**].

The fact that this dataset employs different types of conversation sessions and a rich meta-information, made it convenient for training our PPPG-DialoGPT model. In this section, we start by discussing the filtering steps that we applied to the TG-Redial dataset to keep only the convenient information for the CRS task. Then we highlight some statistical information about the obtained dataset.

- **Data preparation**:
  As we have already mentioned in section 5.3.1, in order to translate the TG-Redial dataset language we used the Google translation API. Using this API we managed to transform all conversations' utterances, users' profiles, topics information as well as movie names from Chinese to the English language. Using prompting templates we combined the different translated information into a structured input so that the model understands the relationships between this different information. To take advantage of the users' personality traits we applied the AWS-EP model (explained in section 4.2) on top of the dialogue information to produce the users' Big Five personality traits. Then we used these traits as control factors within the prompting template to control the generation process for more personalized performance. To make our data convenient for the PPPG-DialoGPT model, we applied the Per-pro-prompt template (as discussed in table 7.1) to each conversation session.

Having the prompted input, we used a pre-trained tokenizer to encode each token within the conversation session and send the tokenized vector representation to our PPPG-DialoGPT model.

- **Data statistics**:

During the experiment process, the dataset was partitioned into three parts, training, validation and testing subsets. The training set comprises 8495 unique conversation session that employ the dialogue's utterances, the dialogue's topics, the user's profile, and the user's Big Five personality traits. The testing set employs 748 dialogue samples, and the validation set comprises 757 unique samples.
Table 5.1 in chapter 5 highlights global statistics of the TG-Redial dataset. Figures 7.3, and 7.4 highlight different information statistics about the train, test, and validation sets independently.

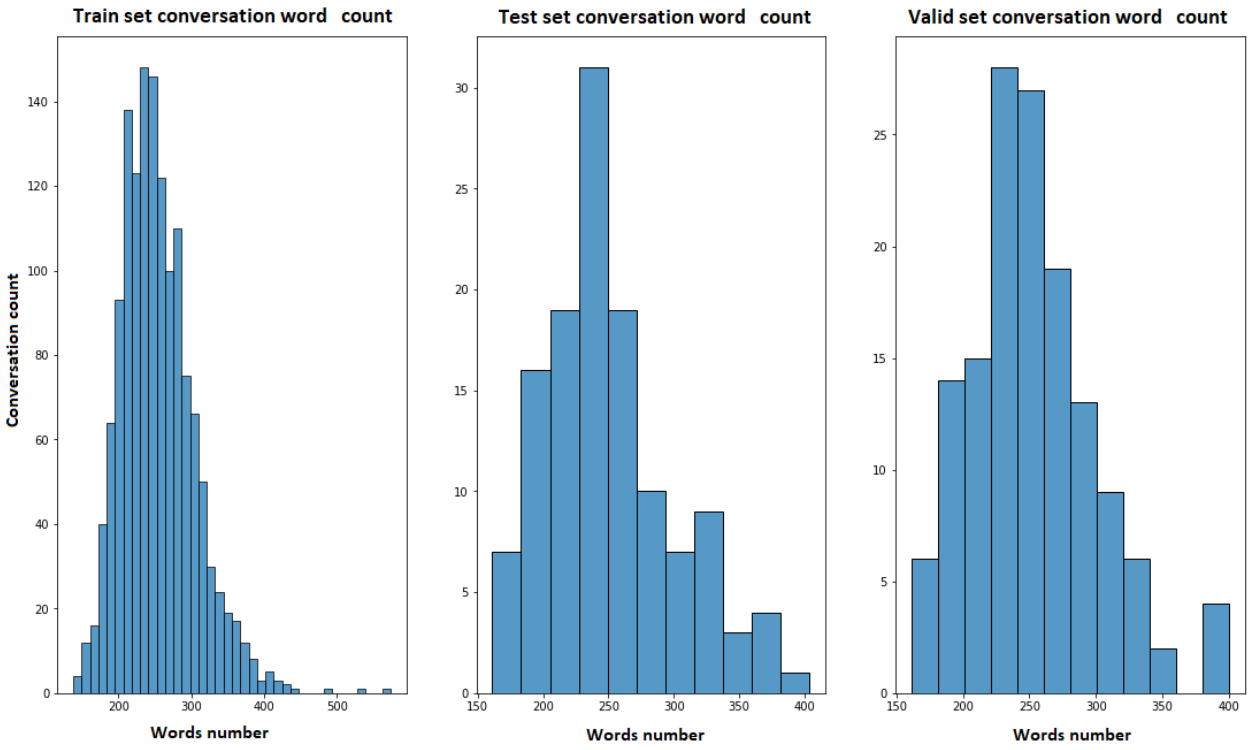

Fig. 7.3. Conversation word count for the train, test, and validation sets.

Figure 7.3 shows that the average word count within the training, validation and testing conversation subsets are around 200 and 300 words per conversation. Figure 7.4 shows that the most common words within the conversation sets are "good", "love", and "beautiful".

Fig. 7.4. Train, test, and validation sets words cloud.

This highlights the fact that within the TGD-Redial conversation set, most users tend to express positive sentences within their dialogues, whether liking and accepting the movie recommendations or expressing positive opinions about specific subjects.

7.4. Experiment and Results

Different experiments are done to investigate the generalization performances of our models using different prompting templates. To keep the fairness of the evaluation process, we used the same training, testing and validation subsets in all the experiments as well as the same hyperparameter values. Table 7.2 highlights the training hyperparameters.

The reasons for choosing the epochs number, the optimizer algorithm, the learning rate, the weight decay and the batch size hyperparameters are similar to those discussed in section 4.4.1.

Table 7.2. The model's hyperparameters.

Hyper-Parameter	Value
Epochs number	50
Optimizer	Adam optimizer
Learning rate	2e-5
Weight decay	0.01
Loss	Autoregressive loss
Batch size	5
Trainable parameters	117M

For each epoch during the training process, we compared the current validation results with the least validation loss and for each prompt-based model we kept the one that gave us the least generalization loss.
The next section highlights the task settings and evaluation results.

7.4.1. Task Settings and Evaluation Results

To evaluate the performance of our prompt-based models, we used a set of automatic evaluation metrics. For evaluating the dialogue generation performance we used the perplexity, Blue 1-2-3 [**126**] and Bert-score [**127**] metrics. These are common metrics that are widely used for evaluating autoregressive language models [**128**] (as shown in appendix section A.6.3). For the movie prediction performance, we used the Mean Reciprocal Rank MRR@k 5.3.2 and the Hit Ratio HR@k 5.3.1 metrics where $k = \{1,3,5,10\}$, which are common metrics for automatically evaluating the model recommendation performance.
Also, to effectively evaluate the generated responses, we removed all the template structure special tokens from both input and generated utterances. We argue that keeping the template special tokens within the generated response and the input text can easily bias our evaluation process. By the fact that we are employing special tokens within the template for each training input, we are encouraging the generation model to understand the structure of the input text. Understanding that each dialogue should start with the "<big5>" token, and each utterance should start with either "<recommender>" or "<seeker>" tokens and ends with the "</topic>" token can easily bias our evaluation results. As when we compare the generated response and the ground-truth ones, we will find that the structure tokens are generally predicted in the right position. Therefore, the results of our metrics, especially for the Blue-1 metrics can be high due to the correct order of the template tokens and not the utterance words. This cannot reveal the real response generation performance of our model. Therefore, in our evaluation setup, we report our models' results by removing the special template tokens from both the input and the generated output and keeping only the utterance information.

Tables 7.3, 7.4 and 7.5, as well as figures 7.5, 7.6 and 7.7 highlight the performance of each prompt-based model compared to state-of-the-art models on different metrics.

Table 7.3. The different prompt-based DialoGPT performance on the test set.

	Perplexity	**Blue-1**	**Blue-2**	**Blue-3**	**precision**	**recall**	**f1**
Redial [**8**]	81.614	0.177	0.028	0.006	-	-	-
KBRD [**8**]	28.022	0.223	0.028	0.009	-	-	-
TG-Redial [**8**]	7.223	0.28	0.065	0.031	-	-	-
GPT-2 [**8**]	13.383	0.279	0.066	0.032	-	-	-
GPT-2-Topic [**9**]	-	0.278	0.064	0.031	-	-	-
TG-RG [**9**]	-	0.282	0.067	0.033	-	-	-
Stage-Two (BERT) [**8**]	-	**0.294**	0.086	**0.043**	-	-	-
Transformer [**8**]	32.856	0.287	0.071	0.032	-	-	-
Promptless	3.921	0.162	0.030	0.009	0.850	0.868	0.859
T-prompt	**3.911**	0.246	0.075	0.028	0.863	0.882	0.874
Pro-prompt	4.192	0.256	0.087	0.035	0.873	0.889	0.881
Pre-prompt	3.946	0.274	**0.095**	0.038	0.876	0.889	0.883
Per-pro-prompt	4.174	0.2668	0.0924	0.0357	**0.9040**	**0.9161**	**0.9099**

Table 7.3 shows that by adding more controls to the generation process both profile-based controls "Pro-prompt" and personality-based template "Pre-prompt" are outperforming the T-prompt base model (which uses only topics as controls) on all evaluation metrics, exception for the perplexity metric. This highlights the importance of using different control information to employ more structured notions for the input text to generate more convenient responses. In addition, figure 7.5 shows that for the Blue@k metrics the Pre-prompt template provides higher results compared to all of the other non-personality template-based models.

This highlights the importance of using personality traits for the response generation task compared to using other types of information such as the profile information "Pro-prompt". However, we can see that combining all generation controls within one template "Pre-pro-prompt" gave us the most effective results on the BERT-score metrics (as demonstrated in figure A.12 in appendix section).

Experiments show that prompt-based leaning paradigm does affect the response generation performance on different metrics. Using our prompt-based templates we outperformed all state-of-the-art baselines on the perplexity metric while having similar results on the Blue-1, Blue-2 and Blue-3 metrics (as demonstrated by figures 7.5 and A.10 in appendix section). This highlights the importance of using the prompt-based learning framework for the dialogue generation task, as by only reshaping the input data we achieved similar and better results compared to existing complex state-of-the-art models. For example, the Stage-Two (BERT) baseline [**9**] which outperforms our proposed solution on the Blue-1 and Blue-3 metrics uses a learning architecture with 800 million trainable parameters in

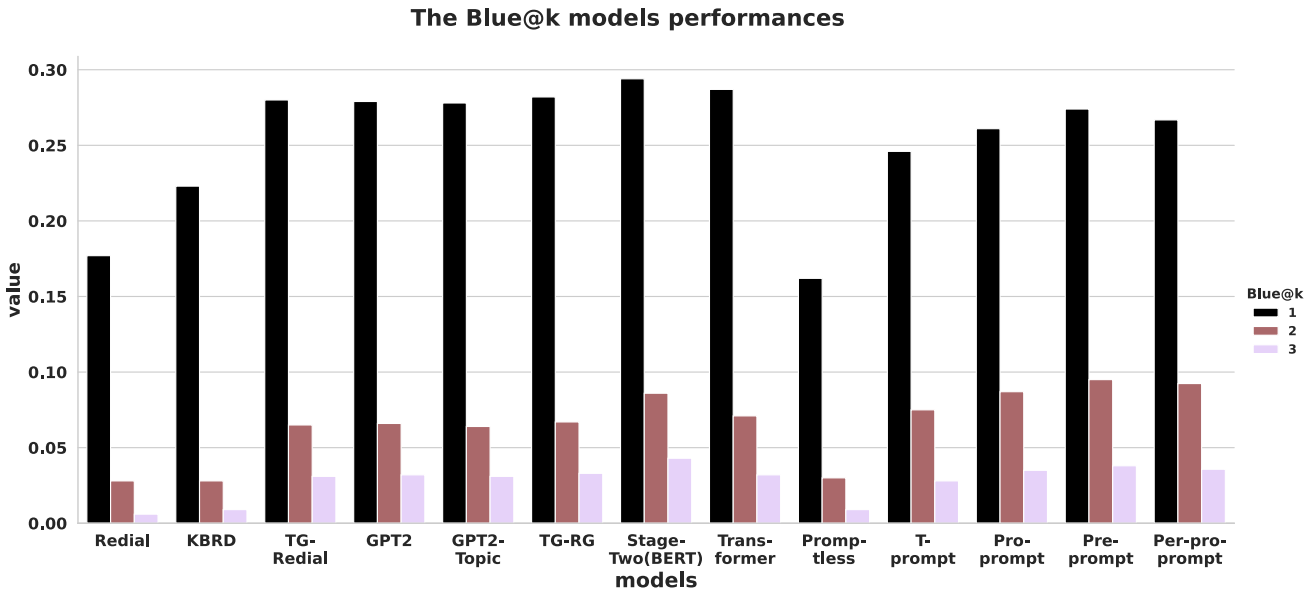

Fig. 7.5. The PPPG-DialoGPT model performance on the Blue@k metric compared to different baselines.

total, it combines two GPT-2-based models and one BERT-based model. However, in our proposed solution due to the prompt-based learning paradigm we only used a 117M trainable parameters model (DialoGPT-small) which is 700% less computationally expensive compared to the Stage-Two (BERT) model (as shown in figure A.8 in appendix section). Yet by using a 700% less complex model, we managed to outperform state-of-the-art models on both perplexity and Blue-2 metrics while having similar results for the Blue-1 and Blue-3 metrics. This highlights the importance of using the prompt learning-based paradigm for enhancing both efficiency and effectiveness of the response generation model.

In addition to the significant efficient and effective results on the response generation task, we were curious to know if our prompt-based models will also affect the movie recommendation performance. In other words, we were curious to know if our solution is capable of providing effective movie recommendations within the generated responses. To do that we formulated the movie prediction task as a movie generation task. When given the ground truth utterance x which contains a movie recommendation, we compare if the correspondent generated response y employs the same movie as x or not. To ensure a fair evaluation process we choose to evaluate our models using the MRR@k and HR@k metrics where $k = \{1,3,5,10,50\}$. For each generation step, we encourage the model to generate k responses using the *Beam search* algorithm [**129**]. We consider these k generated responses as ranked predictions. Then, we define a relevant generated response as a response that

contains the same movie as the one mentioned in the ground truth response. Table 7.4 and figure A.9 (in appendix section) highlight the movie prediction "generation" results of our prompt-based models on the TG-Redial dataset compared to the existing state-of-the-art models.

Table 7.4. The different prompt-based DialoGPT movie prediction performance on the test set.

	MRR@1	**MRR@5**	**MRR@10**	**MRR@50**	**HR@1**	**HR@5**	**HR@10**
Popularity	-	-	0.0011	0.0015	-	-	-
ReDial	-	-	0.0005	0.0009	-	-	-
KBRD	-	-	0.0040	0.0049	-	-	-
GRU4Rec	-	-	0.0014	0.0020	-	-	-
SASRec	-	-	0.0050	0.0068	-	-	-
TextCNN	-	-	0.0119	0.0133	-	-	-
BERT	-	-	0.0182	0.0221	-	-	-
TG-Redial	-	-	0.0240	0.0277	-	-	-
Promptless	0.0194	0.02681	0.0288	-	0.0194	0.0421	0.0575
T-prompt	0.0134	0.02684	0.0286	-	0.0134	0.0506	0.0644
Pro-prompt	0.0182	0.0313	0.0332	-	0.0182	0.0561	0.0712
Pre-prompt	0.0189	0.0318	0.03519	-	0.01898	0.0576	0.0850
PPPG-DialoGPT (Our's)	**0.0245**	**0.03732**	**0.04080**	-	**0.0245**	**0.0635**	**0.0901**

Table 7.4 shows that our personality-based PPPG-DialoGPT model which is trained on an autoregressive language modeling task, is outperforming state-of-the-art models on all of the ranking metrics. This highlights the importance of using prompt-based learning to transfer the knowledge of text generation models for other tasks such as the movie recommendation "classification" task. Also, figure A.9 in appendix section shows that the personality-based models "Pre-prompt, and Pro-pre-prompt models" are outperforming all of the other prompt-based baselines on the movie recommendation task. This highlights the importance of including the individuals' personality traits for the session-based recommendation problems. Moreover, another outstanding result that we obtained during our experiments is that by only producing 10 ranked results (MRR@10), our PPPG-DialoGPT model outperforms state-of-the-art models on the MRR@50 metric. This significantly improves the efficiency of the TG-Redial movie prediction task compared to existing baselines as we have managed to predict more correct movies by considering less ranked predicted samples.

Despite the promising results provided by employing all of the individual Big Five personality traits on the movie recommendation task, we were curious to investigate the impact of each trait on the ranking results independently. In order to do this, we managed to select a specific personality trait within the personality control tokens "<big5>...,</big5>" and we changed its original value with the DialoGPT tokenizer unknown token while keeping all the other traits unchanged. By doing this we succeeded in investigating

the impact of each personality trait absence on the movie recommendation task. Table 7.5 highlights the impact of each personality trait absence on the MRR@k and HR@k metrics.

Table 7.5. The impact of personality traits on the movie recommendation task.

	MRR@1	**MRR@5**	**MRR@10**	**HR@1**	**HR@5**	**HR@10**
PPPG-DialoGPT	0.0245	0.03732	0.04080	0.0245	0.0635	0.0901
Removed AGR	0.0247	0.0377	0.0417	0.0247	0.0628	0.0943
Difference: with AGR	+0.0002	+0.00038	+0.0009	+0.0002	-0.0007	+0.0042
Removed CSN	0.0251	0.0370	0.0416	0.0251	0.0585	0.0929
Difference: with CSN	+0.0006	-0.0003	+0.0008	+0.0006	-0.005	+0.0028
Removed EXT	0.0255	0.0381	0.0417	0.0255	0.0632	0.0913
Difference: with EXT	+0.001	+0.00078	+0.0009	+0.001	-0.0003	+0.0012
Removed NEU	0.0246	0.0380	0.0416	0.0246	0.0637	0.0917
Difference: with NEU	+0.0001	+0.00068	+0.0008	+0.0001	+0.0002	+0.0016
Removed OPN	0.0238	0.0373	0.0414	0.0238	0.0619	0.0929
Difference: with OPN	-0.0007	-0.00002	+0.0006	-0.0007	-0.0016	+0.0028

Table 7.5 shows that removing the Big Five *Openness* (OPN [**130**]) trait from the structure-based input, led to a small decrease in the performance of our PPPG-DialoGPT model on almost all of the ranking metrics. This is also shown in figure 7.6.

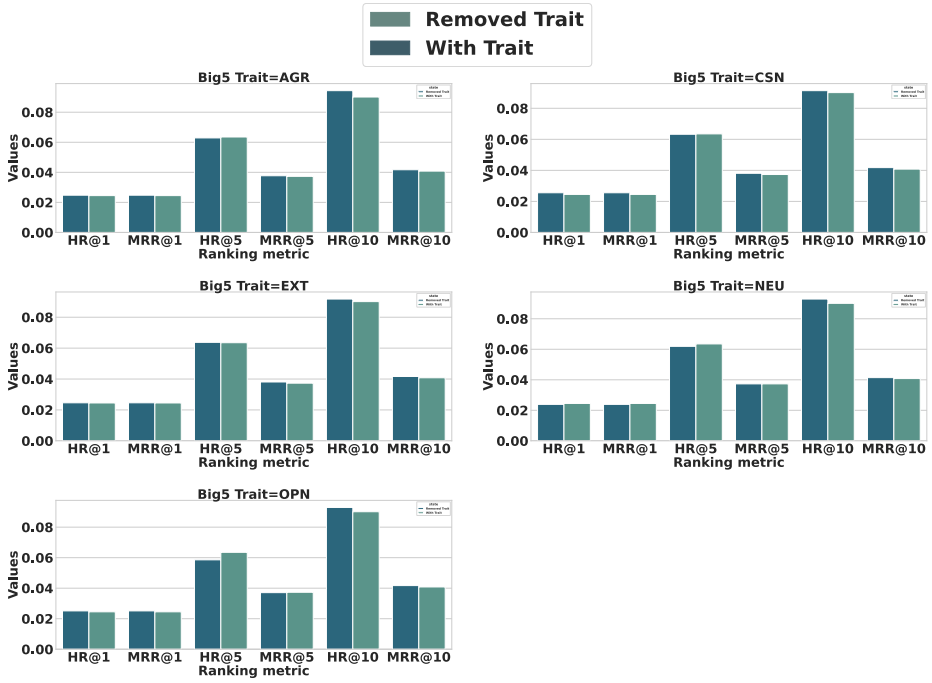

Fig. 7.6. The impact of personality traits on all metrics for the movie recommendation task.

In addition figure 7.7 highlights the impact of the different personality traits on each ranking metric.

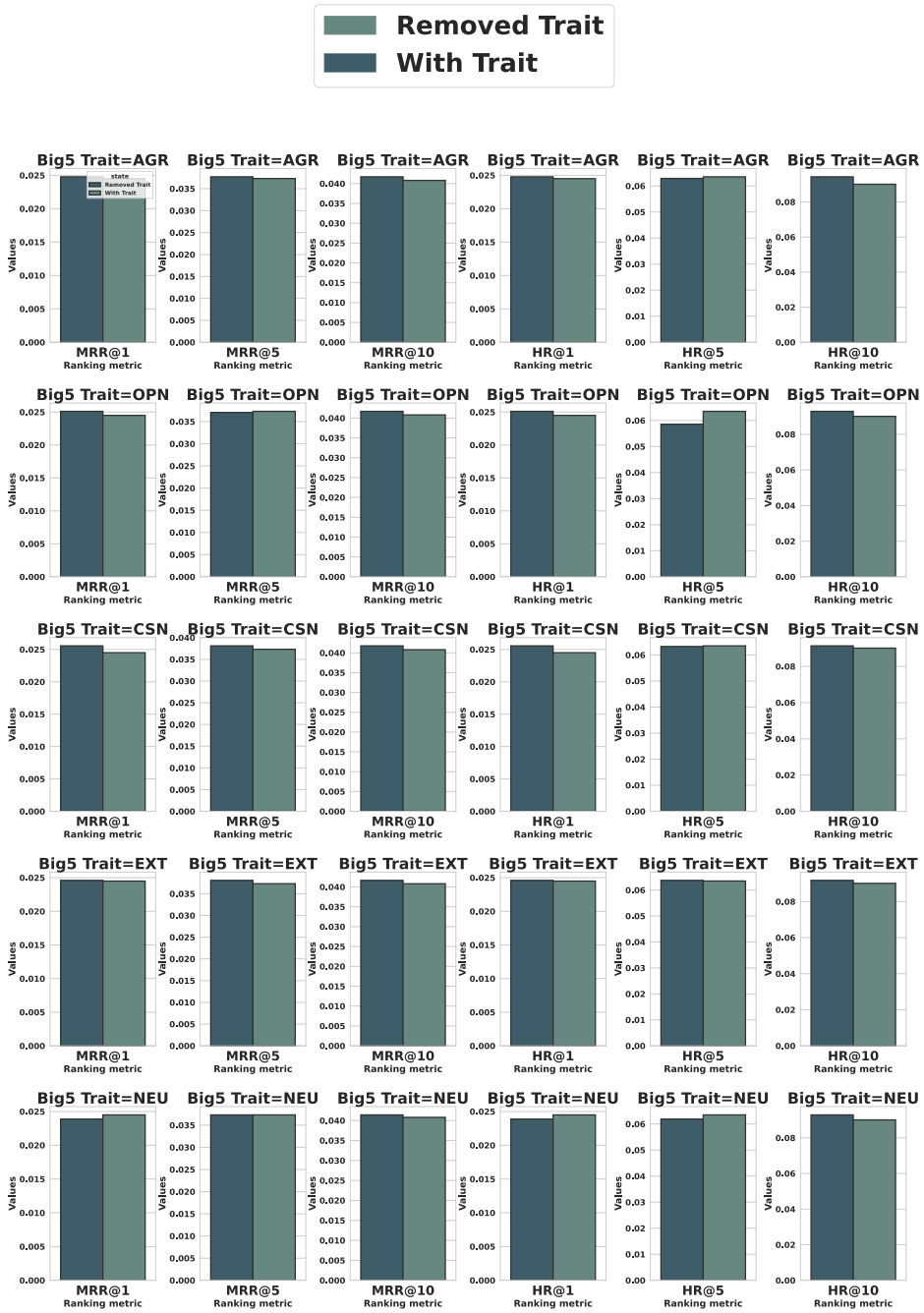

Fig. 7.7. The impact of personality traits on the HR@k ranking metric.

The results shown in table 7.5 as well as in figures 7.6, and 7.7 highlight the importance of having the individual's OPN trait to effectively provide convenient and personalized movie recommendations compared to the other personality traits, where experiments show that removing them didn't affect much the performance of our proposed approach. Also, figure 7.7 and table 7.5 show that the HR@5 metric is the one to get affected the most by removing the different personality traits, as table 7.5 shows that the value of this metric is decreased each time we remove a specific personality trait value, exception for the NEU trait.
To summarize, experiments and results show that using the prompt-based learning paradigm yields a learning model that efficiently and effectively outperforms state-of-the-art models on different metrics for different tasks (response generation and movie recommendation). Results show that the Big Five OPN trait factor is the most important Big Five personality trait for effectively producing a convenient recommendation.

7.5. Conclusion

In this chapter, we highlighted our forth contribution "as discussed in section 1.2" which consists of using the prompt-based learning paradigm to create a personality-aware response generation component. We presented a new Big Five personality-aware dialogue generation framework based on the prompt-learning paradigm. In particular, we evaluated the impact of promoting the knowledge of a pre-trained DialoGPT model on the recommendation task in different ways. We reformulated the CRS utterances by introducing more structures using five different manually designed templates (Promptless, T-prompt, Pro-prompt, Pre-prompt, and Pre-pro-prompt). Experiments and results showed that our PPPG-DialoGPT model yielded excellent results on both response generation and movie recommendation compared to many baselines for the TG-Redial benchmark dataset. Also, it showed a significant efficiency improvement compared to complex recommendation models. Moreover, our experiments showed that adding more structural information to the dialogue utterances has a strong impact on improving the CRS performance. Also, adding individual personality traits is shown to significantly improve CRS performance. This highlights the importance of using the user's personality traits to yield more personalized conversation sessions. However, as in this work, we used only manually designed prompts, in future work we aim to investigate the impact of using automatic prompt-learning techniques instead of just designing them manually (more future work is discussed in the following chapter 8).

Now that we have discussed all of our research's contributions (as shown in section 1.2 and discussed in chapters 4, 5, 6 and 7 respectively) to build our modularized conversation recommendation system (PPerMo). The following chapter concludes this thesis work and discusses possible future work.

Chapter 8

Conclusion and Future Work

This chapter summarizes the thesis by justifying all of the research goals outlined in chapter 1 and by providing insight for future studies.

8.1. Conclusion

Recommendation systems are very important platforms that help to provide users with personalized services. Different recommendation paradigms have been proposed in the literature in the past year. The CRS paradigm has gained significant popularity in the past few years due to its capability of maintaining both current and previous user preferences. In addition, recent advancements in the NLP field have had a significant impact on CRS research advancements. Despite the promising results provided by modern CRSs, these systems still have their drawbacks. They are still considered as large and complex systems to be used by regular resources. Moreover, they still lack some personalization aspects, especially when it comes to using individuals' personality traits to improve the recommendation performance. To the extent of our knowledge, no CRS study employs individual personality traits as features to generate more coherent responses and movie recommendations.

The main objective of this thesis was to bridge the gap between the *CRS paradigm*, recent *NLP techniques* and the *personality* research fields. We mainly focused on investigating the impact of using individuals' personality traits within a modularized CRS by using recent NLP techniques such as pre-trained LM, transfer learning and prompt-based learning paradigms. In our research, we mainly focused on fine-tuning different pre-trained models while also training new hyperparameters on top of them to fulfill different tasks such as movie, topic and personality prediction tasks. We organized our work as a two-goal research project. We *first* designed a novel personality prediction model that predicts the individuals' personality traits using their textual social media posts. In the *second* step, we

used this novel personality prediction approach to create a personality-aware modularized conversation recommendation system and enhance the performance of the different proposed CRS components. More specifically, the following objectives were attained:

- Proposing a novel multi-task and transfer-learning-based personality prediction approach that simultaneously predicts the two most popular personality tests systems (Big Five and MBTI tests [**27**]) from users' social media posts using the weight sharing mechanisms. This contribution is proven through the proof of concept in section 4.2, where our proposed solution outperformed state-of-the-art models by reporting an average Macro-F1 score of 86% on the Pandora dataset and an average F1-score of 83% on the MBTI Kaggle dataset.

- Proposing a sequential movie recommendation model that predicts the most likely movie to be watched by a user using the sequential structure of his previous movie-watching history. We also investigated the impact of using individuals' personality traits in improving the performance of this task. Chapter 5 discusses the main architecture in detail as well as our contribution to this task. Moreover, the experimental results in section 5.3.2 prove the effectiveness of our proposed model, where the MRR@10 ranking value was increased from 0.024 to 0.0912 units.

- Developing a simple but effective topic prediction model that predicts the conversation current topic. We designed this component as a multi-label classification task. To do this, we fine-tuned a recent pre-trained language model called ELECTRA to encode the conversation context. We then built a classification MLP network to predict the right topic labels. Our proposed solution is proven through the different results obtained in section 6.4, where our model reported a precision of 96% and 0.91 units on the HR@5 metric.

- Proposing a prompt-based conversational recommendation system framework. This component aims to generate personality-based and topic-aware conversation recommendation responses. To do this, we reshaped our input text using a manually designed prompting template that employs both the Big Five personality and the utterance topic information. Section 7.2.2 highlights our input-proposed prompting design. This contribution is proven through the proof of concept, where our model reported a 90% precision value for generating context-aware responses and 0.04 units on the MRR@10 metric for providing the convenient movie recommendation.

- Combining all of the proposed components into one modularized CRS called **PPerMo** "**P**rompt-based and **Per**sonality-aware **Mo**dularized CRS", and proposing the first personality-aware and prompt-based CRS framework. Sections 3.1.3 and 3.2 describe how we combined these components into one CRS framework and discussed the objective of each component as well as the working pipeline of our refined movie approach. In addition, figure A.7 highlights different example scenarios of the working mechanism of our proposed modularized CRS system.

8.2. Discussion and Future Work

Throughout this section, we discuss the different drawbacks of our proposed solution and present different improvement approaches for future work.

8.2.1. Discussion

Building a modularized CRS architecture is tedious work and requires an independent design for each component. Manually designing and combining these components generally leads to a loss of performance. Moreover, independently training each component requires high computation resources. This work addresses high computational requirement concerns by training different language models for different tasks on a single GPU. The main intention behind our thesis is to develop a personality-aware modularized conversation recommendation system and discover the impact of employing individuals' personality traits on each module where large GPU resources are not available. Although our proposed solution shows promising performances for the CRS components using different automatic evaluation metrics, our model still has some drawbacks.

Experiments show that our PPerMo model still has some consistency problems, as it tends to contradict itself occasionally, especially when generating movie descriptions. Occasionally, our PPerMo model describes the same movie under different categories (for example, describing a Chinese horror movie as a thriller and horror movie in one utterance, but as a non-horror movie in the second utterance (this is highlighted in appendix section figure A.16). Also, the experiments show that our model tends to repeat itself on different occasions, especially for movie recommendations, where our model tends to recommend the same movie multiple times in the same conversation (as shown in figure A.16 in appendix section). Therefore, in the next section, we discuss different improvement solutions for future work.

8.2.2. Future Work

The following are the future aspirations to accomplish for each module within our PPerMo CRS:

- **For the Personality Prediction Component (AWS-EP)**:
  This model should be enhanced by employing more weight sharing to achieve better generalization results on the two different personality tests. Also, another way to improve this model would be to employ a greater amount of user-related information such as demographic information (age, gender, country, etc.) instead only using social media posts, which is not guaranteed to be the ultimate way to effectively predict their personalities. Moreover, more data and a bigger version of the ELECTRA pre-trained model can significantly improve our personality prediction solution.

- **For the Sequential Recommendation Component (DBT-SR)**:
  To enhance the efficiency and the effectiveness of this model, we may consider using one pre-trained model instead of two DistilBERT models, as by using both representation networks, we will have two different encoding vectors for the same movie, which can prevent the recommendation model from providing accurate item recommendations. Moreover, using larger versions of the DistilBERT pre-trained model instead of smaller ones should effectively improve recommendation performance. In addition, using IDs (for example, movie ID: @125698) instead of titles to present the different movies could introduce a domain gap between the original Distil-BERT input space "natural language words" and our input space "movie IDs". Therefore, in order to solve this problem in future work, we aim to represent the movies by their natural language names, which will allow our model to better generalize for unseen movies. Also, instead of using Distil-BERT for the prefix objective, we can use other prefix-based, pre-trained LM such as GPT-2, which could suit our next movie prediction task better than DistilBERT's MLM. Another possible direction for improving this component is to train it on different sequential recommendation domains such as travel, product, medicine, etc., which should improve its generalization results, especially for one-shot and zero-shot recommendation scenarios.

- **For the Dialogue State Tracking Component (Elec-SP)**:
  One possible direction to improve the efficiency performance of this component is by prompting the knowledge of the pre-trained LM using automatically learned templates instead of using an additional classification network on top of it. Also,

using larger pre-trained ELECTRA model versions should significantly improve our component effectiveness. In addition, in our DST experiments, we did not investigate the impact of users' item interaction history on the prediction of the convenient state nor the effect of different personality traits rather than the Big Five ones (for example, the MBTI traits). Moreover, we did not investigate the impact of different information concatenation techniques rather than the simple summation approach. Therefore, for future work, we aim to focus on investigating the effect of employing these features' information for the topic prediction task.

- **For the Dialogue Generation Component (PPPG-DialoGPT)**:
  Different approaches can be applied to this component to enhance its performance. The first thing that one can do is to clean the training data and keep only the grammatically correct utterances or to try to use a better Chinese to English translation algorithm. Moreover, another way to improve the short-preferences recommendation performance of this component is by adding movies as controls to generate the responses instead of only considering them as words within a response. Also, since we manually designed our prompting template, one way to improve the performance of this module is by automatically learning the prompt templates instead of defining them manually. Moreover, we can use larger DialoGPT pre-trained versions and increase the model training steps to capture more contextual information. Another direction of improvement is by prompting other recent dialogue generation pre-trained models such as BlenderBot [**101**] and Meena [**131**] instead of the DialoGPT LM. Also, using external movie-related data such as movie descriptions or actors etc., can significantly improve the performance of our dialogue generation model by creating a more grounded End-to-End approach.

- **For the whole CRS as one package (PPerMo)**:
  Our proposed PPerMo framework is a modularized framework where different modules are combined to improve the CRS task. One possible direction to improve the performance of this framework is by introducing additional modules such as a response ranking module, where instead of providing the top-one Beam-search-based response, we rank a set of n-generated responses by maximizing the similarity between the predicted and correct answer using a different ranking algorithm (such as the DialogRPT algorithm). In this thesis, we also focused on evaluating each CRS module independently using automatic evaluation metrics. Therefore, in future work, we aim to evaluate the performance of our proposed personality-aware CRS as one package using an online and human-based evaluation framework. Through this evaluation, we aim to expose our proposed CRS framework to real users and monitor our

system's performance while reporting the satisfaction of each user.
Another way to improve the performance of our proposed CRS framework is by designing it as an End-to-End architecture instead of a modularized one and training it as a multi-objective network instead of training each component separately. One way to do that is by using the recent prompt-learning-based paradigm, where a single language model is fine-tuned on different objectives (recommendation, state tracking, response generation, personality prediction, etc.) using specific prompt templates. Future work in this direction could improve both the efficiency and the effectiveness of our CRS performance.

References

[1] R. Chen, Q. Hua, Y.-S. Chang, B. Wang, L. Zhang, and X. Kong, "A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks," *IEEE Access*, pp. 64 301–64 320, 2018.

[2] P. B.Thorat, R. Goudar, and S. Barve, "Survey on collaborative filtering, content-based filtering and hybrid recommendation system," *International Journal of Computer Applications*, pp. 31–36, 2015.

[3] J. Bobadilla, F. Ortega, A. Hernando, and A. GutiéRrez, "Recommender systems survey," *Know.-Based Syst.*, pp. 109—-132, 2013.

[4] D. Jannach, A. Manzoor, W. Cai, and L. Chen, "A survey on conversational recommender systems," *ACM Comput. Surv.*, 2021.

[5] S. Dhelim, N. Aung, M. A. Bouras, H. Ning, and E. Cambria., "A survey on personality-aware recommendation systems," *arXiv:2101.12153*, 2021.

[6] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskeve, "Language models are unsupervised multitask learners," *In arXiv*, 2019.

[7] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *NAACL*, 2018.

[8] K. Zhou, Y. Zhou, X. W. Wayne Xin Zhao, and J.-R. Wen, "Towards topic-guided conversational recommender system," *Coling 2020*, 2020.

[9] H. Wang, M. Cui, and K.-F. W. Zimo Zhou, Gabriel Pui Cheong Fung, "Topicrefine: Joint topic prediction and dialogue response generation for multi-turn end-to-end dialogue system," *arXiv:2109.05187*, 2021.

[10] K. S. Kalyan, A. Rajasekharan, and S. Sangeetha, "Ammus : A survey of transformer-based pretrained models in natural language processing," *arXiv:2108.05542*, 2021.

[11] W. Wu and L. Chen, "Implicit acquisition of user personality for augmenting movie recommendations," *In Lecture Notes in Computer Science. Springer International Publishing*, pp. 302—-314, 2015.

[12] J. C. Seo, L. J. Yong, and J. K. Dong, "Adaptive recommendation system for tourism by personality type using deep learning," *International Journal of Internet, Broadcasting and Communication*, pp. 55—-60, 2020.

[13] J. Sun, D. Ren, and D. Xu, "Leveraging user personality and tag information for one class collaborative filtering," *Cham: Springer International Publishing*, pp. 830—-840, 2018.

[14] N. Neehal and M. A. Mottalib, "Prediction of preferred personality for friend recommendation in social networks using artificial neural network," *in 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE). IEEE*, 2019.

[15] V. Moscato, A. Picariello, and G. Sperli, "An emotional recommender system for music," *IEEE Intelligent Systems*, 2020.

[16] J. Golbeck and E. Norris, "Personality, movie preferences, and recommendations," *in Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining - ASONAM*, pp. 1414—-1415, 2013.

[17] S. Sarawagi, "Information extraction," *USA: now Publishers*, 2008.

[18] Q.-M. Do, K. Zeng, and I. Paik, "Resolving lexical ambiguity in english-japanese neural machine translation." Association for Computing Machinery, 2020, p. 46–51.

[19] T. Hayashi, T. Yoshimura, M. Inuzuka, I. Kuroyanagi, and O. Segawa, "Spontaneous speech summarization: Transformers all the way through." 2021 29th European Signal Processing Conference (EUSIPCO), 2021, pp. 456–460.

[20] G. Malik, M. Cevik, D. Parikh, and A. Basar, "Identifying the requirement conflicts in srs documents using transformer-based sentence embeddings," 2022.

[21] M. B. D. Sánchez and A. Viejo, "Detecting sensitive information from textual documents: An information-theoretic approach," *in Conference: Modelling Decisions in Artificial Intelligence*, 2012.

[22] J. Tang, M. Hong, D. Zhang, B. Liang, and J. Li, "Information extraction: Methodologies and applications," *Emerging Technologies of Text Mining: Techniques and Applications*, 2007.

[23] Y. Susanto, A. Livingstone, B. C. Ng, and E. Cambria, "The hourglass model revisited," *IEEE Intelligent Systems*, pp. 96—-102, 2020.

[24] C. C. Liem, M. Langer, A. Demetriou, A. M. Hiemstra, A. S. Wicaksana, M. P. Born, and C. J. König, "Psychology meets machine learning: Interdisciplinary perspectives on algorithmic job candidate screenings," *In Explainable and interpretable models in computer vision and machine learning*, pp. 197—-253, 2018.

[25] Y. Li, S. Wang, Q. Pan, H. Peng, T. Yang, and E. Cambria, "Learning binary codes with neural collaborative filtering for efficient recommendation systems," *Knowledge-Based Systems*, pp. 64—-75, 2019.

[26] W. Youyou, M. Kosinski, and D. Stillwell, "Computer based personality judgments are more accurate than those made by humans," *Proceedings of the National Academy of Sciences*, pp. 1036—-1040, 2015.

[27] H. Harper, "The best personality tests in ranking order," https://www.uvic.ca/education/assets/docs/best-personality-tests-2019.pdf, 2019, publishing date: 09-05-2019, last visited: 18-02-2022.

[28] M. Ward and S. Gould, "Here are the best jobs for every personality type," https://www.businessinsider.com/the-best-jobs-for-every-personality-type-myers-briggs-2020-9, 2020, publishing date: 15-09-2022, last visited: 09-01-2022.

[29] I. B. Myers, M. H. M. Caulley, and A. L. Hammer, "Introduction to type: A description of the theory and applications of the myers-briggs type indicator," *Consulting Psychologists Press*, 1990.

[30] D. Murano, J. Way, C. Anguiano-Carrasco, K. E. Walton, and J. Burrus, "On the use of the big five model as a sel assessment framework," https://measuringsel.casel.org/use-big-five-model-sel-assessment-framework/#:~:text=On%20the%20Use%20of%20the%20Big%20Five%20Model%20as%20a%20SEL%20Assessment%20Framework,-Twitter%20LinkedIn%20Facebook&text=The%20Big%20Five%20factors%20include,extraversion%20(sociability%3B%20assertiveness)., 218, publishing date: 29-08 2018, last visited: 09-01-2022.

[31] K. Cherry, "The big five personality traits," https://www.verywellmind.com/the-big-five-personality-\dimensions-2795422, 2021, publishing date: 20-02-2021, last visited: 30-09-2021.

[32] S. C. Matz, M. Kosinski, G. Nave, and D. J. Stillwell, "Psychological targeting as an effective approach to digital mass persuasion," *Proceedings of the national academy of sciences*, pp. 12 714—-12 719, 2017.

[33] F. Celli, B. Lepri, J.-I. Biel, D. Gatica-Perez, G. Riccardi, and F. Pianesi, "The workshop on computational personality recognition 2014," *ACM*, pp. 1245—-1246, 2014.

[34] M. Smallcombe, "Structured vs unstructured data: 5 key differences," https://www.xplenty.com/blog/structured-vs-unstructured-data-key-differences/, 2020, publishing date: 03-01-2022, last visited: 30-02-2022.

[35] M. R. Davahli, W. Karwowski, E. G. Franco, and K. Fiok, "Identification and prediction of human behavior through mining of unstructured textual data," *DOI*, 2020.

[36] G. Carducci, G. Rizzo, D. Monti, E. Palumbo, and M. Morisio, "Twitpersonality: Computing personality traits from tweets using word embeddings and supervised learning," 2018.

[37] T. Tandera, D. Suhartono, R. Wongso, and Y. L. Prasetio, "Personality prediction system from facebook users," *Procedia computer science*, pp. 604—611, 2017.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NIPS*, 2017.

[39] M. Gjurkovic, M. Karan, I. Vukojevic, M. Bosnjak, and J. Snajder, "PANDORA talks: Personality and demographics on Reddit," in *Proceedings of the Ninth International Workshop on Natural Language Processing for Social Media*. Association for Computational Linguistics, 2021, pp. 138–152.

[40] T. Yang, F. Yang, H. Ouyang, and X. Quan, "Psycholinguistic tripartite graph network for personality detection," *The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

[41] D. Kelly, "Forget the myers-briggs, use the big five," https://headstuff.org/topical/science/myers-briggs-big-five/, 2019, publishing date: 31-05-2019, last visited: 24-09-2021.

[42] Y. Li, A. mohammad Kazameini, Y. Mehta, and E. Cambria, "Multitask learning for emotion and personality detection," *IEEE*, 2021.

[43] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators." *ICLR*, 2020.

[44] Y. David, D. D. Castro, and Z. Karnin, "One-shot session recommendation systems with combinatorial items," *arXiv:1607.01381*, 2016.

[45] A. Manzoor and D. Jannach, "Conversational recommendation based on end-to-end learning: How far are we?" *Computers in Human Behavior Reports*, p. 100139, 2021.

[46] Y. Zhang, X. Chen, Q. Ai, L. Yang, and W. B. Croftt, "Towards conversational search and recommendation: System ask, user respond," *CIKM*, pp. 177–186, 2018.

[47] W. Lei, G. Zhang, X. He, Y. Miao, X. Wang, L. Chen, and T.-S. Chua, *Interactive Path Reasoning on Graph for Conversational Recommendation*. Association for Computing Machinery, 2020, pp. 2073—-2083.

[48] J. Andreas, J. Bufe, D. Burkett, C. Chen, J. Clausman, J. Crawford, K. Crim, J. DeLoach, L. Dorner, J. Eisner, H. Fang, D. H. Alan Guo, K. Hayes, K. Hill, D. Ho, W. Iwaszuk, S. Jha, D. Klein, J. Krishnamurthy, T. Lanman, P. Liang, C. H. Lin, I. Lintsbakh, A. McGovern, A. Nisnevich, A. Pauls, D. Petters, B. Read, D. Roth, S. Roy, J. Rusak, B. Short, D. Slomin, B. Snyder, S. Striplin, Y. Su, Z. Tellman, S. Thomson, A. Vorobev, I. Witoszko, J. Wolfe, A. Wray, Y. Zhang, and A. Zotov, "Task-oriented dialogue as dataflow synthesis," *Transactions of the Association for Computational Linguistics*, pp. 556–571, 2020.

[49] C. A. Thompson, M. H. G¨oker, and P. Langley, "A personalized system for conversational recommendations," *AAAI*, pp. 393–428, 2000.

[50] huggingface community, "Summary of the tokenizers," https://huggingface.co/transformers/tokenizer_summary.html, 2021, publishing date: x-x-2020, last visited: 26-09-2021.

[51] S. Polamuri, "Most popular word embedding techniques in nlp," https://dataaspirant.com/word-embedding-techniques-nlp/, 2020, publishing date: 18-09-20, last visited: 18-04-2022.

[52] C. Anish, "forming a feature vector for natural language processing," https://medium.com/spidernitt/forming-a-feature-vector-for-natural-language-processing-b49486e1c637, 2021, publishing date: 27-07-2021, last visited: 03-10-2021.

[53] F. Chollet, "Deep learning for text," https://freecontent.manning.com/deep-learning-for-text/, publishing date: 02-10-2017, last visited: 08-02-2022.

[54] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv:1301.3781*, 2013.

[55] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," *Conference: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[56] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "" albert: A lite bert for self-supervised learning of language representations," *in International Conference on Learning Representations*, 2019.

[57] 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," *ICLR*, 2019.

[58] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *in IArXiv*, 2019.

[59] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *in Advances in Neural Information Processing Systems, Curran Associates,Inc*, 2019.

[60] J. Allan, "Introduction to topic detection and tracking," *In Topic detection and tracking. Springer*, 2002.

[61] A. Tumasjan, T. O. Sprenger, P. GSandner, and I. M. Welpe, "Predicting elections with twitter: What 140 characters reveal about political sentiment," *In Fourth international AAAI conference on weblogs and social media*, 2010.

[62] P. S. Earle, D. C. Bowden, and M. Guy, "Twitter earthquake detection: earthquake monitoring in a social world," *Annals of Geophysics54,6*, 2012.

[63] O. Oh, K. H. Kwon, and H. R. Rao, "An exploration of social media in extreme events: Rumor theory and twitter during the haiti earthquake 2010," *In Icis*, pp. 7332—-7336, 2010.

[64] J. Zhang and C. Danescu-Niculescu-Mizil, "Balancing objectives in counseling conversations: Advancing forwards or looking backwards," *in Proceedings of ACL*, 2020.

[65] T. M. Lai, Q. H. Tran, T. Bui, and D. Kihara, "A simple but effective bert model for dialog state tracking on resource-limited systems," *ICASSP*, 2020.

[66] G. Shani, D. Heckerman, and R. I. Brafman, "An mdp-based recommender system," *Appears in Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI2002)*, pp. 1265—-1295, 2005.

[67] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," *ACM*, pp. 811—-820, 2010.

[68] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, pp. 1735—-1780, 1997.

[69] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," *EMNLP*, pp. 1724—-1734, 2014.

[70] T. Donkers, B. Loepp, and J. Ziegler, "Sequential user-based recurrent neural network recommendations," *Association for Computing Machinery*, pp. 152—-160, 2017.

[71] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "A dynamic recurrent model for next basket recommendation," *ACM*, pp. 729—-732, 2016.

[72] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations," *ACM*, pp. 843—852, 2018.

[73] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *ICLR*, 2016.

[74] J. Li, P. Ren, Z. R. Zhumin Chen, T. Lian, and J. Ma, "Neural attentive session-based recommendation," *ACM*, pp. 1419—1428, 2017.

[75] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, "Personalizing session-based recommendations with hierarchical recurrent neural networks," *RecSys*, pp. 130—-137, 2017.

[76] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing., "Recurrent recommender networks," *ACM*, 2017.

[77] X. Chen, H. Xu, Y. Zhang, J. Tang, Y. Cao, Z. Qin, and H. Zha, "Sequential recommendation with user memory networks," *ACM*, pp. 108—-116, 2018.

[78] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," *ICDM*, pp. 197—206, 2018.

[79] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: Shortterm attention/memory priority model for session-based recommendation," *KDD*, pp. 1831—-1839, 2018.

[80] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," *WSDM*, pp. 565—-573, 2018.

[81] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," *CIKM*, 2019.

[82] A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, , and Y. Choi, "Learning to write with cooperative discriminators," *preprint arXiv:1805.06087*, 2018.

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

[84] X. Zhang and M. Lapata, "Chinese poetry generation with recurrent neural networks," *EMNLP*, pp. 670—-680, 2014.

[85] T. Wolf, V. Sanh, J. Chaumond, and C. Delangue, "Transfertransfo: A transfer learning approach for neural network based conversational agents," *preprint arXiv:1901.08149*, 2019.

[86] 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.

[87] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, "Wizard of wikipedia: Knowledge-powered conversational agents," *preprint arXiv:1811.01241*, 2018.

[88] J. Weizenbaum, "Elizaa computer program for the study of natural language communication between man and machine," *Communications of the ACM*, pp. 36–45, 1966.

[89] S. Santhanam and S. Shaikh, "A survey of natural language generation techniques with a focus on dialogue systems - past, present and future directions," *arXiv:1906.00500*, 2019.

[90] O. Borisov, "Text generation using n-gram model," https://towardsdatascience.com/text-generation-using-n-gram-model-8d12d9802aa0, 2020, publishing date: 27-10-2020, last visited: 21-01-2022.

[91] T. Mikolov, M. Karafiat, L. Burget, J. Cernock, and S. Khudanpur, "Recurrent neural network based language model," *In Interspeech*, 2010.

[92] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, pp. 436–44, 2015.

[93] Y. Goldberg, "A primer on neural network models for natural language processing," *J. Artif. Int. Res.*, pp. 345—420, 2016.

[94] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, pp. 1137—-1155, 2003.

[95] C. G.-D. B. F. B. H. S. Y. B. Kyunghyun Cho, Bart van Merrienboer, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *EMNLP*, 2014.

[96] pretrain.nlpedia, "Pretrain, prompt, predict a new paradigm for nlp)," http://pretrain.nlpedia.ai/, publishing date: 10-10-2021, last visited: 04-03-2022.

[97] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu, "Plug and play language models: A simple approach to controlled text generation," *In International Conference on Learning Representations*, 2020.

[98] E. M. Smith, M. Williamson, K. Shuster, J. Weston, and Y.-L. Boureau, "Can you put it all together: Evaluating conversational agents' ability to blend skills." ArXiv, 2020, pp. 2021–2030.

[99] B. Hedayatnia, K. Gopalakrishnan, S. Kim, Y. Liu, M. Eric, and D. Hakkani-Tur, "Policy-driven neural response generation for knowledge-grounded dialogue systems," 2020.

[100] 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*. Association for Computational Linguistics, 2020, pp. 7871–7880.

[101] S. Roller, E. Dinan, N. Goyal, D. Ju, M. Williamson, Y. Liu, J. Xu, M. Ott, K. Shuster, E. M. Smith, Y.-L. Boureau, and J. Weston, "Recipes for building an open-domain chatbot," *arXiv:2004.13637*, 2020.

[102] M. Tsimpoukelli, J. Menick, S. Cabi, S. Eslami, O. Vinyals, and F. Hill, "Multimodal few-shot learning with frozen language models," *In Advances in Neural Information Processing Systems.*, 2021.

[103] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," *In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pp. 4582—4597, 2021.

[104] W. Yin, J. Hay, and D. Roth, "Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach," *In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3912—-3921, 2019.

[105] N. Z. J. Y. S. D. C. T. F. H. L. S. Xiang Chen, Xin Xie and H. Chen, "Adaprompt: Adaptive prompt-based finetuning for relation extraction," *arXiv:2104.07650*, 2021.

[106] T. Schick and H. Schutze, "Generating datasets with pretrained language models," *arXiv:2104.07540*, 2021.

[107] L. Li, Y. Zhang, and L. Chen, "Personalized prompt learning for explainable recommendation," *arXiv:2202.07371*, 2022.

[108] D. Sileo, W. Vossen, and R. Raymaekers, "Zero-shot recommendation as language modeling," *arXiv:2112.04184*, 2021.

[109] R. Li, S. Kahou, H. Schulz, V. Michalski, L. Charlin, and C. Pal, "Towards deep conversational recommendations," *NeurIPS*, 2018.

[110] J. Zou, Y. Chen, and E. Kanoulas, "Towards question-based recommender systems," *SIGIR*, 2020.

[111] Z. Liu, H. Wang, Z.-Y. Niu, and H. Wu, "Towards conversational recommendation over multi-type dialogs," *ACL*, 2020.

[112] K. Zhou, W. X. Zhao, S. Bian, Y. Zhou, J.-R. Wen, and J. Yu, "Improving conversational recommender systems via knowledge graph based semantic fusion," *KDD*, 2020.

[113] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *JMLR*, 2019.

[114] C. Soto, "Personality can change over a lifetime, and usually for the better)," https://www.npr.org/sections/health-shots/2016/06/30/484053435/personality-can-change-over-a-lifetime\-and-usually-for-the-better, publishing date: 30-06-2016, last visited: 09-03-2022.

[115] J. Mitchell, "(mbti) myers-briggs personality type dataset," *Kaggle*, 2017.

[116] F. Celli, F. Pianesi, D. Stillwell, and M. Kosinski, "Workshop on computational personality recognition: Shared task," *In Proceedings of WCPR13, in conjunction with ICWSM-13*, 2013.

[117] U. of Phoenix, "I believe personality is so hard to define," https://www.coursehero.com/file/13318101/I-believe-personality-is-so-hard-to-define-because-there-are-so-many-different-characteristics-invol/, 2020, publishing date: x-x-x, last visited: 10-19-2021.

[118] I. Masnikosa, I. Vukojević, I. Crnomarković, J. Šnajder, J. Jukić, M. Gjurković, M. Bošnjak, M. Karan, and S. Bakić, "Pandora," https://psy.takelab.fer.hr/datasets/all/pandora/, 2020, publishing date: 20-05-2021, last visited: 09-01-2022.

[119] M. Mosbach, M. Andriushchenko, and D. Klakow, "On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines," 2020.

[120] H. Bao, L. Dong, F. Wei, W. Wang, N. Yang, X. Liu, Y. Wang, S. Piao, J. Gao, M. Zhou, and H.-W. Hon, "Unilmv2: Pseudo-masked language models for unified language model pre-training," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20. JMLR.org, 2020.

[121] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," *In Proceedings of WWW*, 2017.

[122] P. Xu and Q. Hu, "An end-to-end approach for handling unknown slot values in dialogue state tracking," *ACL*, 2018.

[123] S. Mirkin, S. Nowson, C. Brun, and J. Perez, "Motivating personality-aware machine translation," *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1102—-1108, 2015.

[124] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *arXiv:2107.13586*, 2021.

[125] Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan, "Dialogpt: Large-scale generative pre-training for conversational response generation," *ACL*, 2020.

[126] 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*, ser. ACL '02. Association for Computational Linguistics, 2002, pp. 311—-318.

[127] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," *arXiv:1904.09675*, 2021.

[128] C. Gao, W. Lei, X. He, M. de Rijke, and T.-S. Chua, "Advances and challenges in conversational recommender systems: A survey," *AI Open*, pp. 100–126, 2021.

[129] M. Freitag and Y. Al-Onaizan, "Beam search strategies for neural machine translation," *Proceedings of the First Workshop on Neural Machine Translation*, 2017.

[130] L. R. Goldberg, "The structure of phenotypic personality traits," *American psychologist*, 1993.

[131] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu, and Q. V. Le, "Towards a human-like open-domain chatbot," *arXiv:2001.09977*, 2020.

[132] V. Balakrishnan and E. Lloyd-Yemoh, "Stemming and lemmatization: A comparison of retrieval performances." *in In: Proceedings of SCEI Seoul Conferences, Seoul*, 2014.

[133] Bitext, "what is the difference between stemming and lemmatization," https://blog.bitext.com/what-is-the-difference-between-stemming-and-lemmatization/, 2021, publishing date: 07-07-2021, last visited: 03-10-2021.

[134] Sendex, "Stemming words with nltk," https://pythonprogramming.net/stemming-nltk-tutorial/, 2021, publishing date: 03-05-2015, last visited: 03-10-2021.

[135] R. Mitkov, "Part of speech tagging," *n The Oxford Handbook of Computational Linguistics*, pp. 220–227, 2003.

[136] S. Bird, E. Klein, and E. Loper, "Categorizing and tagging words," https://www.nltk.org/book/ch05.html, 2021, publishing date: 04-09-2019, last visited: 03-10-2021.

[137] E. Tiu, "Understanding zero-shot learning — making ml more human," https://towardsdatascience.com/understanding-zero-shot-learning-making-ml-more-human-4653ac35ccab, publishing date: 23-06-2021, last visited: 08-03-2022.

[138] M. Thaker, "Comparing text summarization techniques," https://medium.com/@thakermadhav/comparing-text-summarization-techniques-d1e2e465584e, publishing date: 24-03-2019, last visited: 11-03-2022.

[139] J.-C. Gu, Z.-H. Ling, X. Zhu, and Q. Liu, "Dually interactive matching network for personalized response selection in retrieval-based chatbots," 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)*. Association for Computational Linguistics, 2019, pp. 1845–1854.

[140] Y. Zhang and Q. Yang, "A survey on multi-task learning," *On IEEE Transactions On Knowledge And Data Engineering*, 2021.

[141] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv:1503.02531*, 2014.

[142] J. Alammar, "The illustrated gpt-2 (visualizing transformer language models)," https://jalammar.github.io/illustrated-gpt2/, publishing date: 12-10-2019, last visited: 23-02-2022.

Appendix A

The Research Appendix

A.1. Background and Literature Review Appendix

This section highlights the appendix information related to chapter 2.

A.1.1. Lemmatization

Lemmatization is one of the most common and popular approaches during the NLU steps. This approach aims to convert words into dictionary form, which eventually reduces the text's dimension [**132**]. The lemmatizer algorithm converts the tokens into their corresponding lemma by applying predefined rules such as removing the suffix or using a similar word as a replacement [**133**]. This technique helps in understanding the similarity between different natural language words.

A.1.2. Stemming

The stemming approach aims to reduce the dimensionality of the word by changing them to their root form and keeping only the stem part of the word. Unlike Lemmatization, stemming will only remove the suffix of the words, which might result in a nonsense word that does not exist in the dictionary form [**132**]. Different stemming techniques were proposed in the literature. The most commonly used stemming approach is the Porter Stemmer algorithm [**134**].

A.1.3. Part-of-Speech Tagging

Part-of-speech (POS) tags are used to determine the meaning of the word tokens individually. The word meaning heavily depends on the context within a sentence since words can have different conjugal forms (noun, verb, adverb, etc.) [**135**]. The POS taggers algorithm assigns to each word a specific tag based on the conjugal forms "tag set" and the sentence natural language (English, French, Chinese, etc..) [**136**].

A.1.4. Zero-Shot Learning

Zero-shot learning is an automatic learning paradigm, where at test time, the trained model needs to predict samples from classes that were not observed during the training phase [**137**].

A.1.5. Few-Shot Learning

Few-shot learning is an automatic learning paradigm, where at training time, only a limited number of examples are given to the learner (a very small set of training samples, generally less than 100 samples). Then at test time, the trained model needs to use the small learned knowledge to predict instances within a very larger set compared to the training set.

A.1.6. Prompt-based Working Mechanism

In general, using the prompting paradigm, predicting a given output y can be defined as a three-step process. First, we need to apply a prompting function f_{prompt} which is used to modify an input x into a prompt x'. Within this function, a template "a textual string" which has an input slot [X] and an answer slot [Z] is created. The slot [X] will be filled with the input x, and the slot [Z] will be filled by the model-generated answer.
After having the template the second step is to search for the corresponding answer that maximizes the score of the LM. In this step, we need to define a Z space for the possible answers. Then given the answer space, we define a filling function $f_{fill}(x',z)$ that fills in the slot [Z] within the x' prompt. Then we use a searching function that tries to maximize the pre-trained LM objective of generating the correct answer using the template x'.
Finally, the third step is to map the generated answer z into the task end-goal output space using a mapping function. Figure A.1 discusses the description and examples of the different prompting notions.

As discussed in [**124**], prompt-based learning approaches attempt to reduce the models' parameters space by reformulating the input into a new form using a prompting template. The performance of a specific pre-trained model highly depends on the given prompt. Therefore, it is important to design a good prompt so that the model can provide effective end-goal results. To more discuss the idea behind promoting and prompt engineering we highlight in the next subsections the main component of prompt engineering.

- **Prompt Engineering:**
  As discussed in [**124**], "prompt engineering is the process of creating a prompting function $f_{prompt}(x)$ that results in the most effective performance on the downstream task". From the shape perspective, prompts can be distinguished into two main types,

Name	Notation	Example	Description
Input	$\boldsymbol{x}$	I love this movie.	One or multiple texts
Output	$\boldsymbol{y}$	++ (very positive)	Output label or text
Prompting Function	$f_{\text{prompt}}(\boldsymbol{x})$	`[X]` Overall, it was a `[Z]` movie.	A function that converts the input into a specific form by inserting the input $\boldsymbol{x}$ and adding a slot `[Z]` where answer $\boldsymbol{z}$ may be filled later.
Prompt	$\boldsymbol{x}'$	I love this movie. Overall, it was a `[Z]` movie.	A text where `[X]` is instantiated by input $\boldsymbol{x}$ but answer slot `[Z]` is not.
Filled Prompt	$f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z})$	I love this movie. Overall, it was a bad movie.	A prompt where slot `[Z]` is filled with any answer.
Answered Prompt	$f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z}^*)$	I love this movie. Overall, it was a good movie.	A prompt where slot `[Z]` is filled with a true answer.
Answer	$\boldsymbol{z}$	"good", "fantastic", "boring"	A token, phrase, or sentence that fills `[Z]`

Fig. A.1. Terminology and notation of prompting methods (z represents the answers that correspond to true output y) [**124**].

close prompts, and *prefix prompt*. Close prompts are a type of prompts where the answer slot [Z] is defined within the input x. However, a prefix prompt is a type of prompting approach where the answer slot is placed at the end of the input text. From the designing perspective, we can distinguish prompts as *manually designed* template-based prompts (where the template is handcrafted by humans) or *automated learning-based* templates (where the template is automatically learned through an optimization approach). Learned template-based prompts can further be divided into *discrete* prompts (where the prompt is a textual string) or *continuous* prompts (where the prompt is described using the LM embedding space). More details about the prompt engineering approach are discussed in Pengfei Liu et al. survey [**124**]. The next subsection highlights the different strategies to fine-tune a prompt-based pre-trained language model.

- **Training Strategies for Prompting Methods:**
  To obtain an appropriate prompt, we need to train the LM to provide a convenient answer using the designed prompt. To do that, different training strategies were proposed in the literature. Figure A.2 highlights the different possible training strategies.

  – **Promptless Fine-tuning**
    During the promptless fine-tuning learning strategy all the pre-trained LM parameters are updated using gradients from the downstream task.

Strategy	LM Params Tuned	Additional Trainable Parameters for Prompt	Examples
Promptless Fine-tuning	✓	NA	ELMO,BERT,BART
Tuning-free Prompting	✗	✗	GPT-3
Fixed-LM Prompt Tuning	✗	✓	Prompt-Tuning
Fixed-prompt LM Tuning	✓	✗	PET
Prompt + LM Fine-tuning	✓	✓	PADA

Fig. A.2. Characteristics of different tuning strategies.

- **Tuning-free Prompting**
  This training approach generates answers without changing the pre-trained language model parameters. Tuning-free Prompting is also called in-context learning.
- **Fixed-LM Prompt Tuning**
  In this scenario, the pre-trained language model parameters are fixed. However, the prompt parameters are updated using the downstream task supervision signal.
- **Fixed-prompt LM Tuning**
  Unlike the previous learning approach, This learning technique tunes the language model parameters. However, the prompt parameters are fixed.
- **Prompt+LM Fine-tuning**
  In this setting, both prompt and LM parameters are tuned during the learning process.

Choosing the right learning strategy depends on the task that we want to achieve and also on the selected prompt designing approach. More details are well discussed in Pengfei Liu *et al.* survey [**124**].

A.2. The Proposed PPerMo Framework Appendix

This section highlights the appendix information related to chapter 3.

A.2.1. Preference Generation Component

Text Summarization is a process that converts a large body of text (i.e., paper articles) to small sentences (i.e., abstract of an article) while keeping the main original context within the summarized text [**138**]. Standard automatic text summarization tasks are defined as supervised problems. Where a model takes as input a large body of text as a training feature and tries to generate a summary as a label.
To track and generate the user's preferences expressed within a conversation in our proposed solution. We formulated this problem as a text summarization problem. Where given the large body of text "the conversation utterances in our case", we aim to generate a small text-based summary "the user's preferences in our case".
To train our preference generation model as a text summarization problem, we used the PERSONA-CHAT dataset [**139**] which employs a set of conversation sessions labeled by the preferences of both dialogue speakers (person1, person2).

Within the PERSONA-CHAT dataset, each conversation is annotated by the users' preferences. To generate these preferences we used the T5 [**113**] hugging face pre-trained model to prompt its pre-acquired summarization knowledge. The reason behind using the T5 model compared to other models that have been also pre-trained on the summarization task (example BART [**100**]), is that T5 was not only pre-trained on the summarization task. It is also trained on different other NLP tasks such as the translation and QA tasks, etc. Recent research shows that training large LMs on a multi-tasks paradigm can significantly improve the generalization results in the fine-tuning step [**140, 113**]. Therefore, we intend to fine-tune this model for the preference generation task and benefits from its pre-acquired multi-tasks knowledge.

Now that we fine-tuned the T5 model on the PERSONA-CHAT for the summarization task, we now have a model that can generate the individuals' preferences from their conversation. We used this model as our preference generation model by providing the user's previous preferences with the conversation dialogue utterances as inputs to the model to finally generate the user's new preferences. Table A.1 highlights an example of the Preference Generation model training data.

Table A.1. A one instance example of the Preference Generation model training data.

Preferences	Conversation
I love horror movies. I love reading books. I have two jobs. I love my family	-Hello what are you doing today? -I am good, I just got off work and tired, I have two jobs. -I'm watching a horror movie. -Wow! I do love a good horror movie. loving this cooler weather but a good movie is always good. -Yes! my son is in junior high and I just started letting him watch them too. I work in the movies as well. -I used to work in the human services field. -My wife works and I stay at home. -Nice,i only have one parent so now I help out my mom. -I bet she appreciates that very much. -She raised me right, I am just like her. My dad was always busy working at home depot.

Training and evaluating our preference generation model yield promising results on different metrics as shown in table A.2.

Table A.2. The impact of personality traits on the movie recommendation task.

Metric	Value
rouge2-precision	0.1155
rouge2-recall	0.0983
rouge2-f1	0.1037
Blue@1	0.230667
Blue@2	0.054785
Blue@3	0.019440
Bert-Score-precision	0.8844
Bert-Score-recall	0.8601
Bert-Score-f1	0.8720

As shown in table A.2 our preference generation model provides effective values for different language modeling evaluation metrics, such as Blue@k, Rouge@2, Bert-Score, etc.

A.3. The Personality prediction module (AWS) Appendix

This section highlights the appendix information related to chapter 4.

A.3.1. Electra Model

ELECTRA is a new pre-training approach that aims to match or exceed the MLM (Masked language modeling) pre-trained model on downstream performance while using significantly fewer computational resources.

Unlike BERT which heavily relies on the Masked Language Modeling (MLM) during the pre-training phase, this model uses a new training approach named the Replaced Token Detection (RTD) approach. The ELECTRA architecture combines both a generator and discriminator modules. Figure A.3 highlights the ELECTRA model components. The Gen-

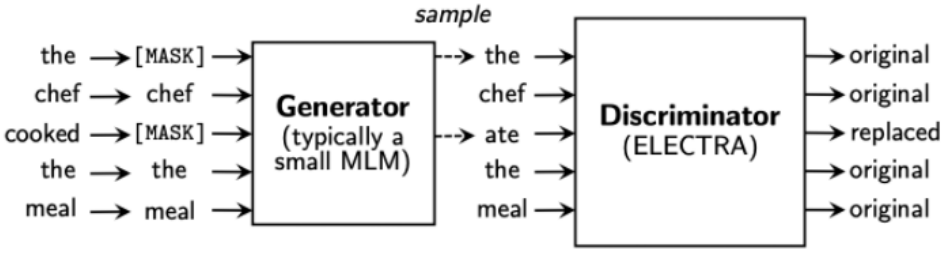

Fig. A.3. Electra Architecture [**43**].

erator component is trained using the MLM goal, meaning that it is trained to predict the masked token. The Discriminator aims to detect for each word provided by the generator whether it has been replaced or not. Therefore, instead of only knowing the 15% masked words in the sentence and predicting the right tokens as BERT does, this model will see all the tokens in the sentence and predict whether it is the original token or the replaced one. Knowing all the words instead of only 15% of them gave the Electra model more insights into the context within a sentence. Moreover, using the discriminator as a binary classifier to predict whether the word has been replaced or not has helped the ELECTRA model to gain time during the training phase. As binary classification is less computationally expensive compared to the word generation task.

To effectively train this model the authors propose two losses, one for the Discriminator L_{Disc} and one for the Generator L_{MLM}.

$$L_{MLM}(x,\theta_G) = E(\sum_{i \in m} -\log p_G(x_i/x^{masked})) \tag{A.3.1}$$

θ_G is the generator learning parameters, x_i is the current token input and x^{masked} is the replacement tokens vector.

$$L_{Disc}(x,\theta_D) = E(\sum_{t=1}^{n} -1(x_t^{corrupt} = x_t)\log D(x_t^{corrupt}) \\ -1(x_t^{corrupt} \neq x_t)\log(1 - D(x_t^{corrupt}))) \tag{A.3.2}$$

θ_D defines the discriminator learning parameters, 1 defines the indicator function and $x_t^{corrupt}$ defines the replaced token. To train both the generator and the discriminator in an End-to-End process they combined both losses into a single function with the addition of a new penalty term λ for the discriminator loss [**43**].

$$min_{\theta_G,\theta_D} \sum_{x \in X} L_{MLM}(x,\theta_G) + \lambda L_{Disc}(x,\theta_D) \tag{A.3.3}$$

This loss aims to minimize both the generator and the discriminator losses on an End-to-End training process.

Using the pre-trained Electra Masked Language Modeling head we aim to produce a more contextual representation for each user's textual sentence to produce better text classification performance.

A.3.2. AWS-EP Toy Example

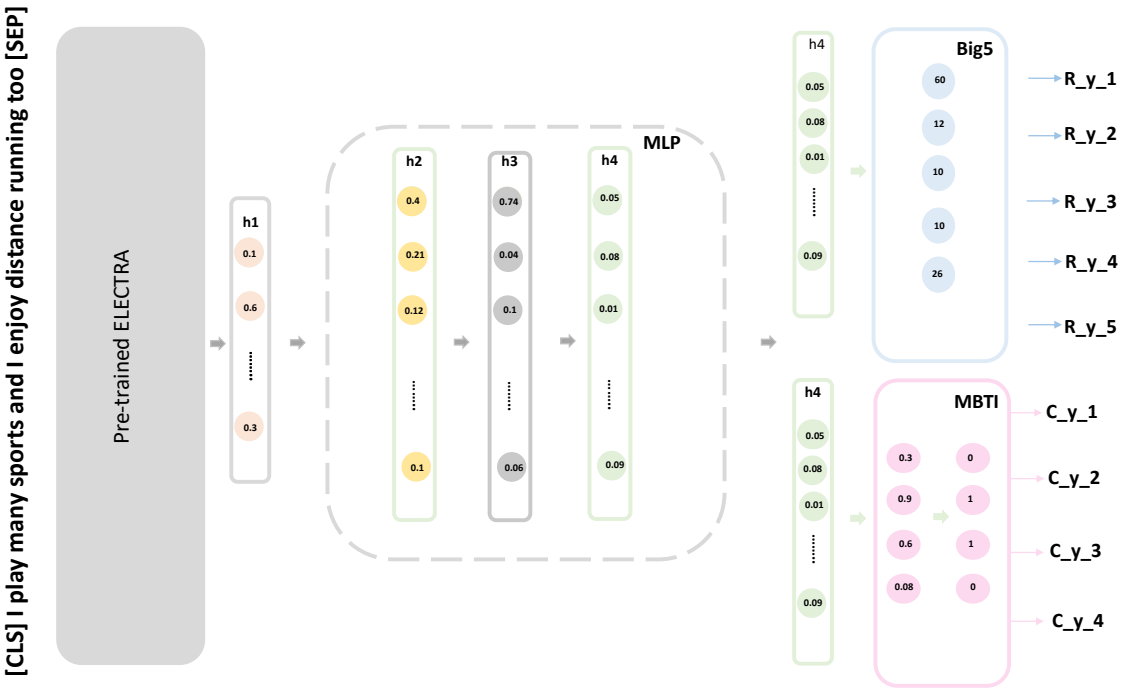

Fig. A.4. ALL Weights Shared ELECTRA for Personality prediction toy example.

First, a sequence of words "a user's social media comment or post" is defined by a sentence start "[CLS]" and a sentence end "[SEP]" tokens. This sequence of words is fed

as input to the Pre-trained ELECTRA model (ELECTRA tokenizer + ELECTRA base encoder). The encoder will create a contextual vector representation for the sequence of words denoted as h1. Then the new sentence representation h1 is passed to the MLP network, where we have different linear layers and normalization approaches. Each linear layer "h2, h3, h4" will refine the embedding representation given to it as input to improve the end-goal task. In a way that, h2 will refine the representation of h1, h3 will refine the representation of h2, and h4 will refine the representation of h3. The final embedding vector provided by the MLP network is the representation of its final layer h4.
This representation is then shared between both prediction heads to learn the optimal weights that effectively predict both personality factor tests. The last layers (Big Five, and MBTI heads) are used as prediction heads. Given the h4 vector representation, both MBTI and Big Five linear layers try to predict the convenient values for each personality trait factor (for example, [0,1,1,0] for the MBTI personality test and [60,12,80,10,26] for the Big Five personality test). In this toy example for the sequence "[CLS] I play many sports and I enjoy distance running too [SEP]" our AWS-EP predicts [0,1,1,0] for the MBTI personality test (which means that the user is not an Extrovert "he is an introvert", he is a thinker, he is a sensor, and he is not a judger "he is a perceiver"). For the Big Five personality test values our model predicts [60,12,10,10,26] (which means that our user is 60% open, 12% conscientious, 10% extroverted, 10% agreeable, and 26% neurotic).

In this toy example, we can see that the h4 vector representation is used as an input for both prediction heads which means that all the previous layers related to the h4 layer are shared by both heads. Feeding the same vector representation to two different tasks will encourage our AWS-EP model to learn a convenient representation that works well on both tasks. This can significantly improve the model generalization performance on unseen data.

A.3.3. Ethical Impact Of Our Work

Despite the vast benefits of knowing the user's personality on his/her daily life services, having the individual personality traits without his/her permission or explicitly indicating his/her personality to us can be unacceptable. We believe that attempting to detect the individual personality can be a personality intrusion. Knowing the individual's personality can help us know his/her preference, his/her behaviour and his/her social relationship with others, etc. If the user did not consent to us knowing all stated information, then knowing them is simply a privacy intrusion. Moreover, acquiring such information about the users can lead to mental and physical harm. Knowing what the user likes or dislikes can easily affect him/her and can be detrimental either mentally or physically (for example, manipulating

the user to do something dangerous).
These are the main reasons why the Pandora dataset [**39**] is not a public dataset, and to use it, you need to submit a request explaining why you are seeking the use of this dataset. Also, the authors of this dataset employs rigorous terms of use [**118**] to protect the users within the dataset. For example, one cannot transfer or reproduce any part of the dataset and attempt to identify or contact any user in the dataset. One cannot publicly display users' names and sensitive information and messages. Also, one can report findings publicly only on an aggregate level. We believe that the user has the right to keep his/her personality private. Whether personality is consciously or unconsciously revealed in any way, it is the other person's responsibility to act diligently and protect the shared information to prevent putting anybody in harm's way. Therefore, our work does not expose any users' private information, and we do not take users' unique identifiers or demographic information to predict their personalities. Our predictive model only focuses on the posted users' social media textual contents. In other words, we do not focus on "who" posted the content but rather on the content itself. Using only the textual content to predict the individual's personality helps us effectively reduce privacy intrusion risks. Our work is extremely valuable and can improve many service providers. Only using the content of the users' posted texts without employing specific users' information helped us reduce the privacy intrusion issues. However, we think that our model is limited in providing compelling encrypted personality predictions. For now, our model only predicts the personality traits in their original forms. However, it would be more secure in predicting them in an encrypted way. Therefore, we aim to enhance the capability of our model by introducing an encryption mechanism for the predicted results. We believe that it is essential for our personality predictive model to be used in the right, protected, and secured environment that does not harm the users or reveal their personalities in any way.

A.3.4. Personality Prediction Baselines' Training Behaviour

Figure A.5 highlights the training behaviour of the OC-EP, OR-EP, EWS-EP, and AWS-EP models.

More discussions are stated in section 4.4

A.3.5. EWS-EP Model Architecture

Figure A.6 highlights the main components within this architecture.

The objective of this architecture stated in section 5.2.2

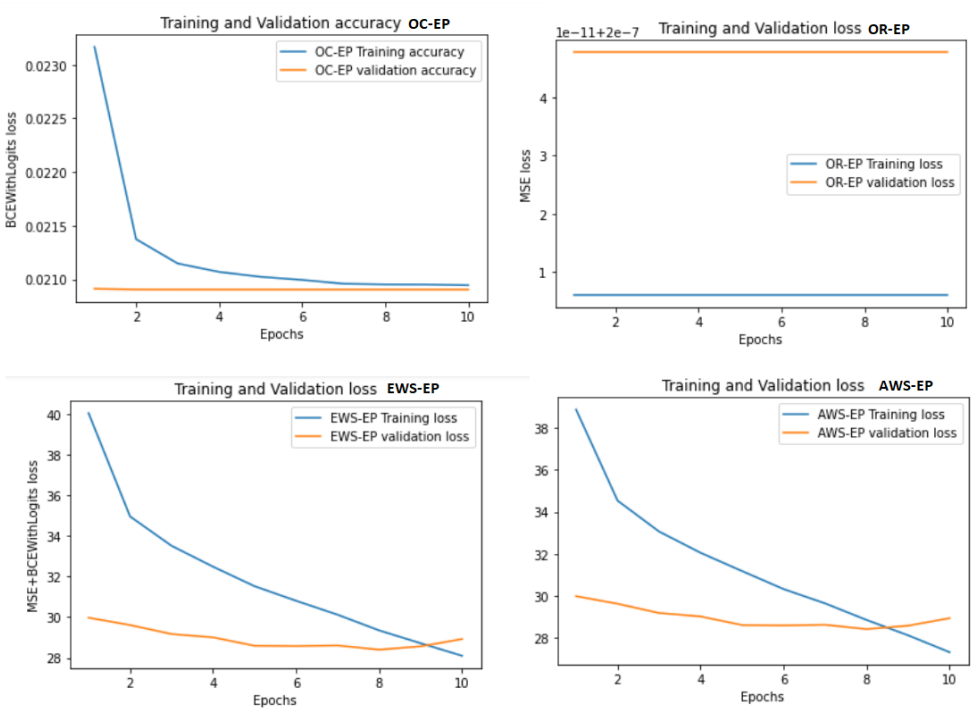

Fig. A.5. OC-EP, OR-EP, EWS-EP, and AWS-EP training performances.

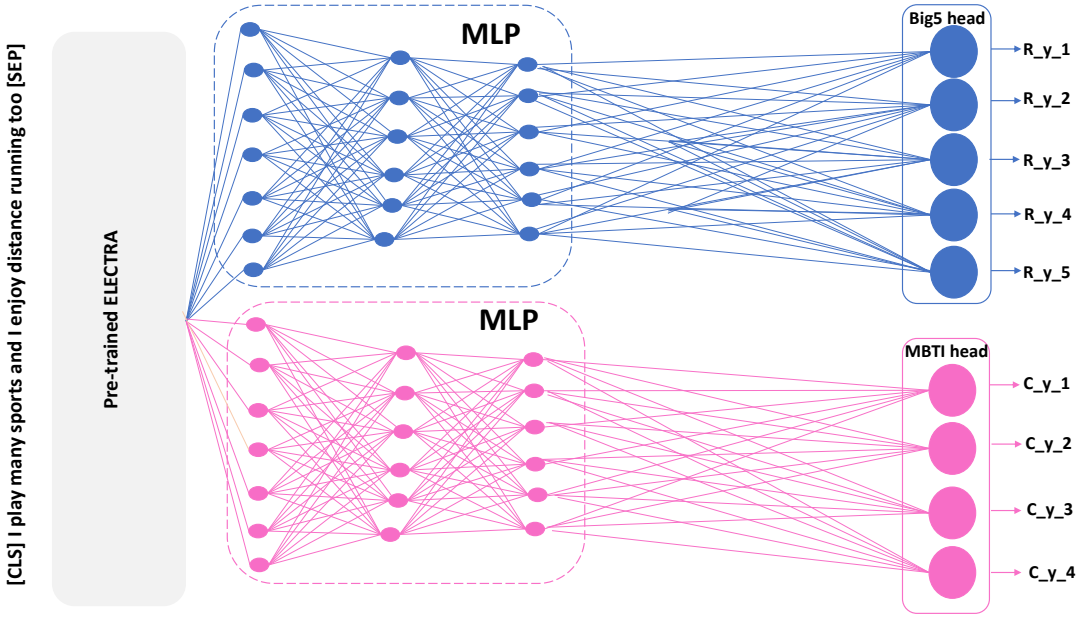

Fig. A.6. Electra Weights Shared Electra for Personality prediction Architecture.

A.4. The Sequential Movie Recommendation module (DBT-SR) Appendix

This section highlights the appendix information related to chapter 5.

A.4.1. DistilBERT

DistilBERT [**58**] is a light BERT version model that has 40% fewer parameters compared to the original BERT model while retaining 97% of its language understanding capabilities. It is also 60% faster than the original model. Using the *knowledge distillation* pre-training approach and a triple loss objective, the author shows that a 40% smaller Transformer LM (*student*) that is trained via the supervision of a bigger Transformer LM (*teacher*) is capable of providing similar results compared to the original model. As published in Hinton et al. [**141**] work, knowledge distillation is the process of compression of a large model (the teacher) into a compact model (the student) that is capable of reproducing the behaviour of the original model in a faster way and with fewer parameters. To do that the student network is trained over the soft target probability loss of the teacher.

$$L_{ce} = \sum_i t_i * \log(s_i)$$

where t_i is the teacher's estimated probability loss.
In DistilBERT the authors formulated t_i as a softmax-temperature function:

$$p_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

where T defines the output distribution smoothness control parameter, and zi defines the model score for class i. The DistilBERT authors defined the final training objective as a combination of the distillation loss L_{ce}, the original BERT objective loss L_{mlm}, and a cosine similarity loss L_{cos}.
To create the student-DistilBERT architecture 'the light model', the authors maintain the same general architecture as BERT while reducing the number of layers by a factor of 2 and removing the token-type embedding and pooler layers. They also optimized most of the Transformer architecture operations using linear algebra frameworks. Furthermore, to initialize the student parameters in an optimized way, the authors used the teacher network by taking one layer out of 2 as initialization factors for the student network to take advantage of the common dimensionality between the two networks. They also trained the distilled architecture without the NSP objective to reduce the training computation.
Using all these optimization tricks, DistilBERT was capable of providing 97% of the BERT performance in a faster way.

A.5. The Dialogue State Tracking module (Elec-SP) Appendix

This section highlights the appendix information related to chapter 6.

A.5.1. PElec-SP: Personality aware ELECTRA transformer for State Prediction

Figure A.7 highlights the PElec-SP model architecture.

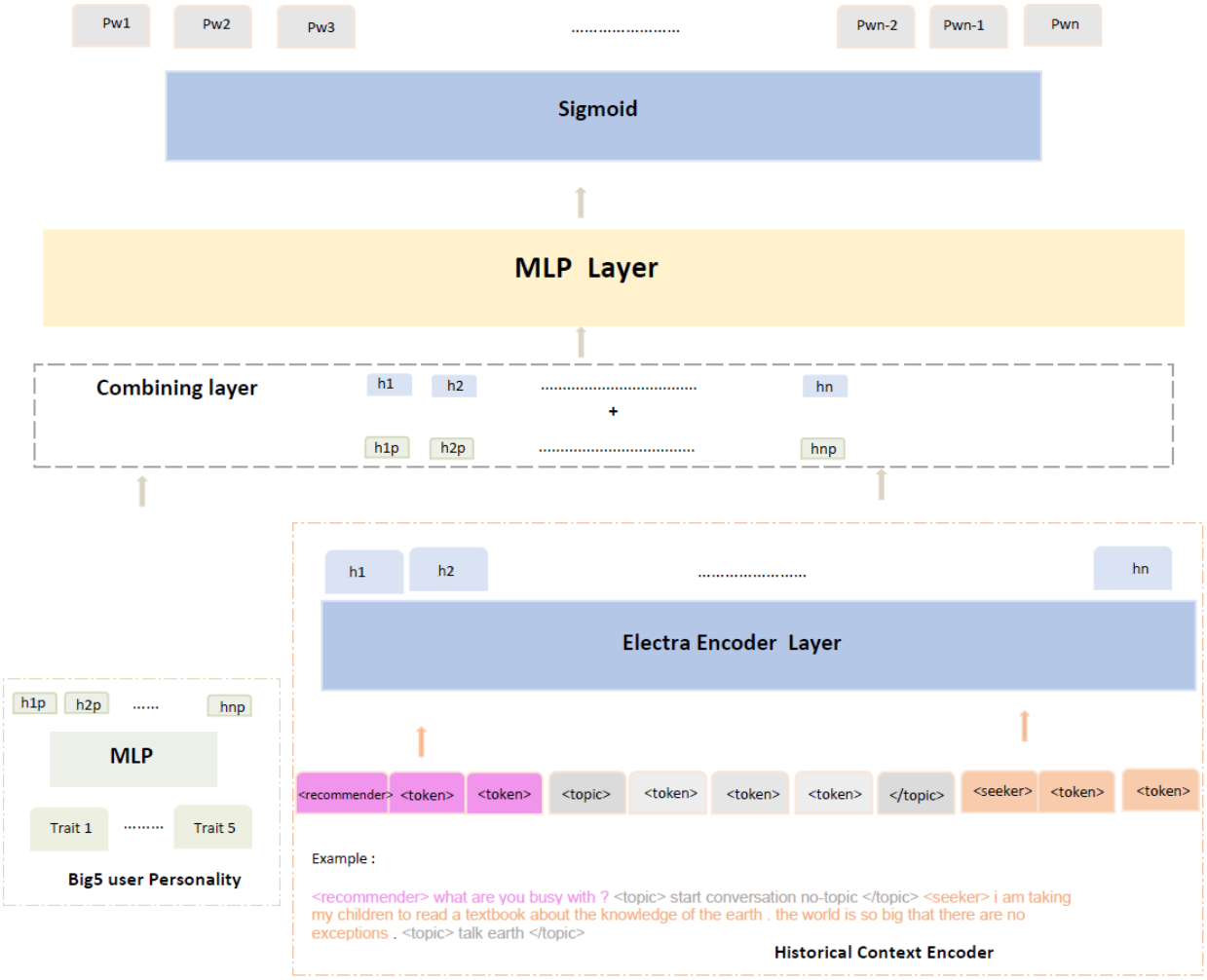

Fig. A.7. PElec-SP model architecture.

PElec-SP architecture is similar to the Elec-SP model 6.2.2. It uses the sigmoid function as the final layer, the Binary cross-entropy as the loss function, and a historical context encoder to encode previous utterances, intents, and topics. The only difference here is that in this approach we combine the user's personality p_u with the input feature X. Therefore, our objective becomes a maximizing function:

$$max \sum_{t=1}^{n} P(Y_t|X_t,p_u) = min \sum_{t=1}^{n} -P(Y_t|X_t,p_u) \tag{A.5.1}$$

where X_t and Y_t are the same as equation 6.2.1. The Electra Encoder component has the same role as in the Elec-SP model. The Big Five personality component is responsible for

creating vector representations for the user's personality traits. Given the personality and the representations of the user's previous conversation information, both information are combined into a single representation vector. This vector is then used as input to the MLP component to predict the next topic tokens using a sigmoid multi-label classification layer. The sigmoid layer has the same role as in the Elec-SP model.

A.6. The Dialog Generation module (PPPG-DialoGPT) Appendix

This section highlights the appendix information related to chapter 7.

A.6.1. GPT-2

GPT-2 [**6**] is a decoder-based transformer language model architecture 2.2.2 trained on a massive dataset. GPT-2 was trained on a 40GB internet crawled data called WebText using self-supervised learning by maximizing the probability of the input x "p(x)" as the main objective learning using the autoregressive learning paradigm. Given the previous tokens within a sentence, GPT-2 tries to predict the most probable token to appear using the maximum likelihood objective. To separate the different crawled text during the training process, the authors of this paper proposed to use special tokens "<|endoftext|>" for both start and end generation. more details are explained in [**6, 142**].

A.6.2. DialogGPT

DialoGPT is a GPT-2 instance based-architecture [**125**] where instead of training the model on web crawled text "as GPT-2 does", the model is pre-trained on a Reddit dialogue corpus. DialoGPT is a large, tunable response generation model. Unlike GPT-2 model [**6**] which aims to produce and generate unstructured open-domain text that is rich in content, DialoGPT aims to produce a dialogue structured response generation text. To do that the model was designed using the same architecture as the GPT-2 model but with simple upgrades. First, the training data was designed to contain only pair/session dialogue turns. Where each training row combines a set of utterances between two speakers separated with the "|<endoftext>|" token to finally produce a long text. The training objective is formulated as an auto-regressive task where we aim to maximize the probability of the most likely word to be predicted. Therefore, the objective function is to minimize the negative log-likelihood of predicting the most likely word.

$$L_{sentence} = -\sum_{i}^{|V|} \log p(w_i|c) \tag{A.6.1}$$

Where $w_i \in$ V , $V = \{w_1, w_2..w_n\}$ is the model vocabulary, c=$\{C = w_1, ..w_{i-1}\}$ is the set of all generated preceding words.

A.6.3. Evaluation Metrics Description

- **Perplexity:** Perplexity is the measurement of how accurate a probability model can predict a sample. For NLP, it measures how well a language model generates samples

that preserve the sentence context.

$$PP(W) = P(w_1,w_2,..,w_n)^{-\frac{1}{N}} = N\sqrt{\frac{1}{q(x_1)q(x_2),..,q(x_n)}} \tag{A.6.2}$$

where $q(x_i)$ presents the model prediction at the position i

- **Blue score:** Blue (Bilingual evaluation understudy) is a score used to compute the similarity between one text and one or more other references. In its original form, Blue was developed to evaluate translation models. However, it can be also applied for evaluating autoregressive and language generation models. This metric computes the n-gram matches between two sentences.

$$Blue = BP.\exp\sum_{n=1}^{N} w_n \log p_n \tag{A.6.3}$$

where BP is the brevity penalty ($BP = 1$ if c > r and $BP = \exp 1 - (r/c)$ if c<r), r is the count of words in a reference, c is the count of words in a candidate, and p_n is the n-gram modified precision (sum of all n-gram counts for all candidates in the corpus normalized by the number of candidate n-gram):

$$\mathrm{p}_n = \frac{\sum_{C\in\{Candidates\}}\sum_{n-gram\in\{C\}} Count_{clip}(n-gram)}{\sum_{C'\in\{Candidates\}}\sum_{n-gram'\in\{C'\}} Count(n-gram')} \tag{A.6.4}$$

where $Count_{clip} = min(Count, Max_{RefCount})$, $Max_{RefCount}$ is the maximum number of n-grams occurrences in any reference count, and Count is the maximum number of times a candidate n-gram occurs in any single reference

- **Recall**: It counts the number of overlapping n-grams in both generated output and the original reference divided it by the total number of n-grams in the original reference.

$$recall = \frac{count(n_{gram})_{match}}{count(n_{gram})_{reference}} \tag{A.6.5}$$

- **Precision**: Similar to the recall metric, it counts the number of overlapping n-grams in both generated output and the original reference but we divide by the total number of n-grams in the model $n_{grammodel}$ count instead of the $n_{gramreference}$ count.

$$precision = \frac{count(n_{gram})_{match}}{count(n_{gram})_{model}} \tag{A.6.6}$$

- **F1-Score**: The ROUGE F1 score is a weighted metric that combines both precision and recall.

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{A.6.7}$$

A.6.4. Training and Validation Behaviour:

Figure A.8 highlights the complexity of our proposed baselines compared to state-of-the-art Stage-Two (BERT) model [**9**].

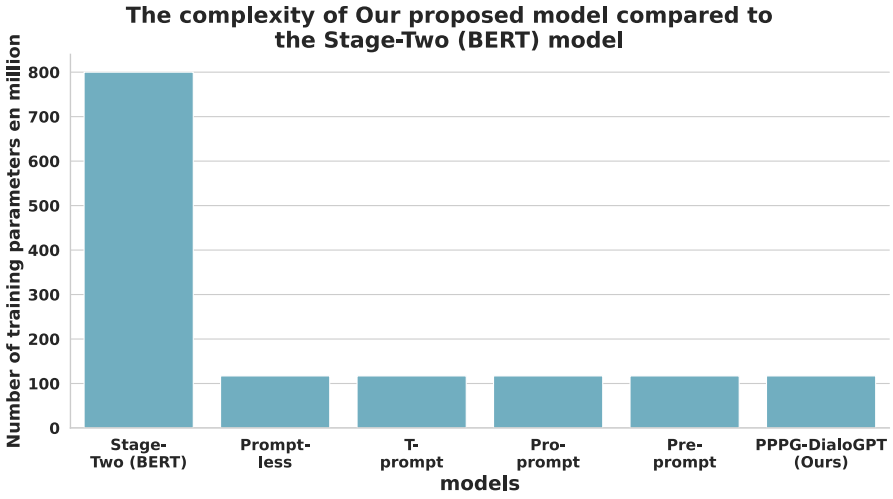

Fig. A.8. Our proposed model compared to the Stage-Two (BERT) model complexity.

Figure A.9 highlights the movie recommendation performance of each prompt-based baseline.

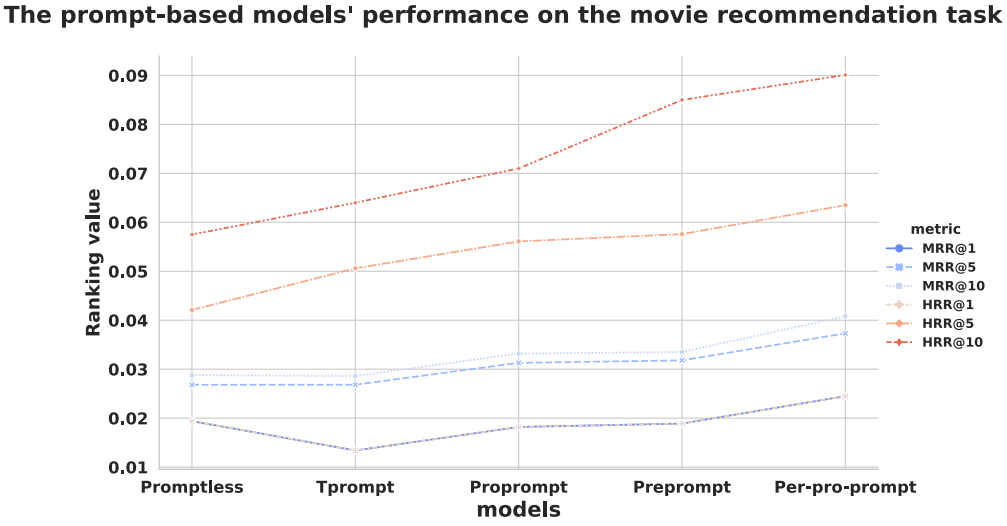

Fig. A.9. The prompt-based models performance on the movie recommendation task.

Figure A.10 highlights the perplexity value of each baseline compared to state-of-the-art models.

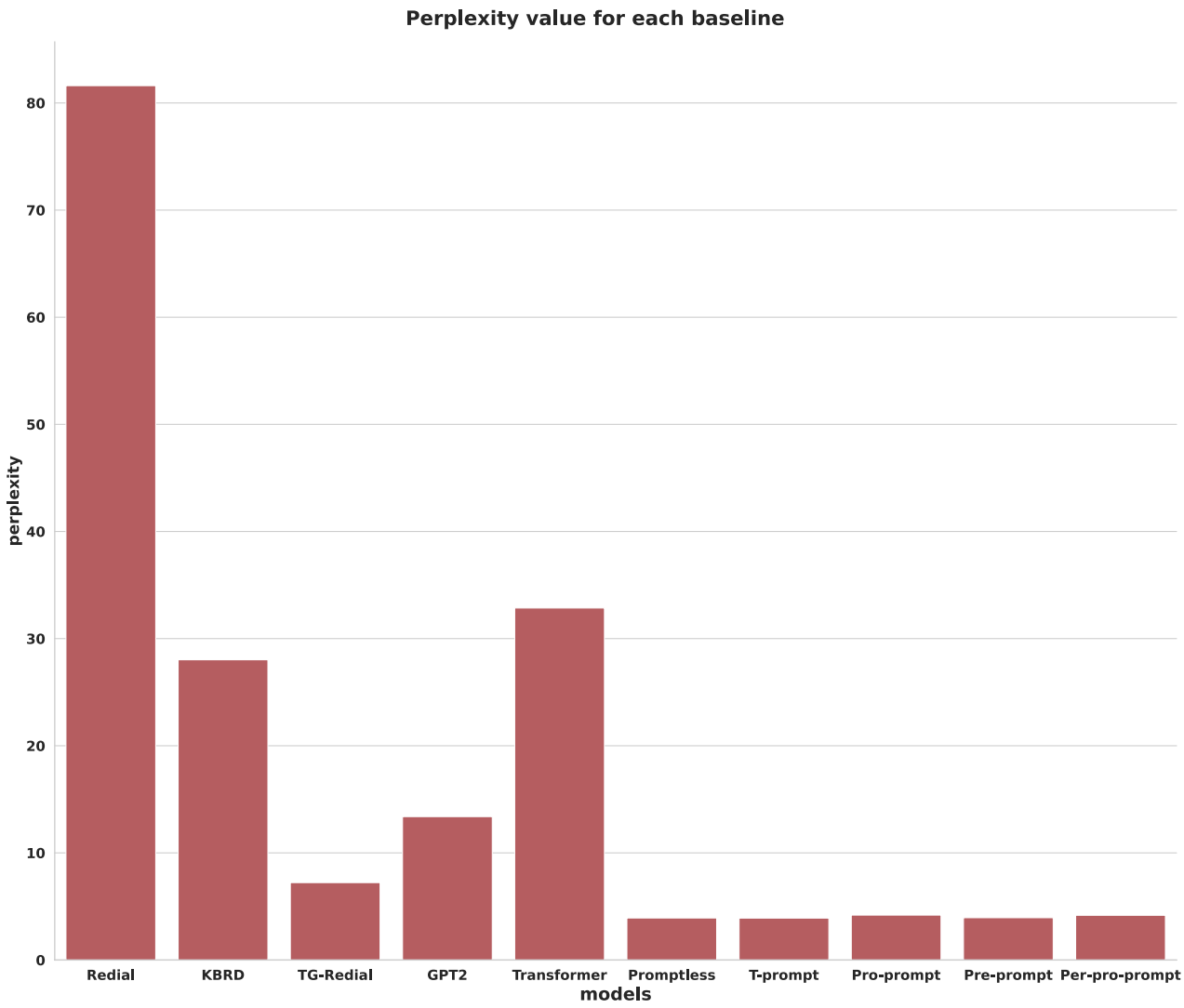

Fig. A.10. Perplexity value for each baseline.

Figures A.11 highlights the training and validation behaviour of our prompt-based baselines.

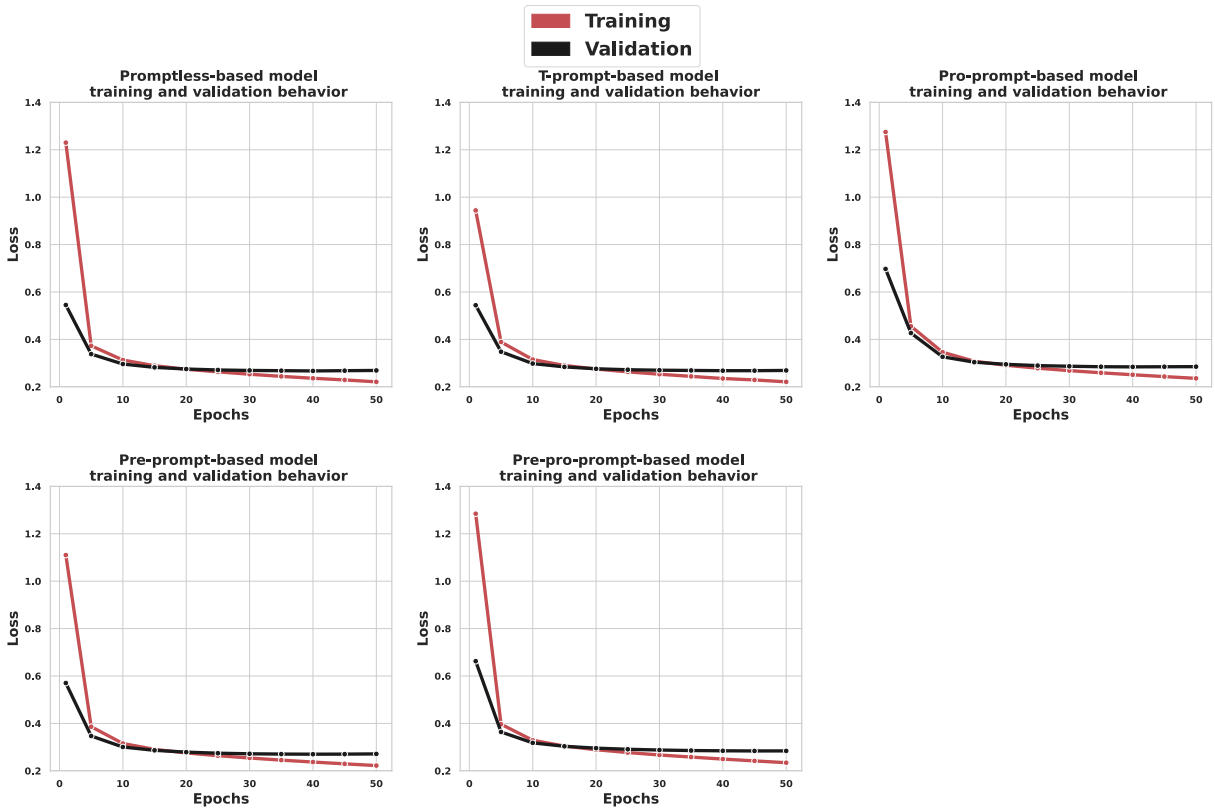

Fig. A.11. The prompt-based Baseline: Training and Validation performance.

Figure A.12 highlights the Bert-Score performance of each prompt-based baseline.

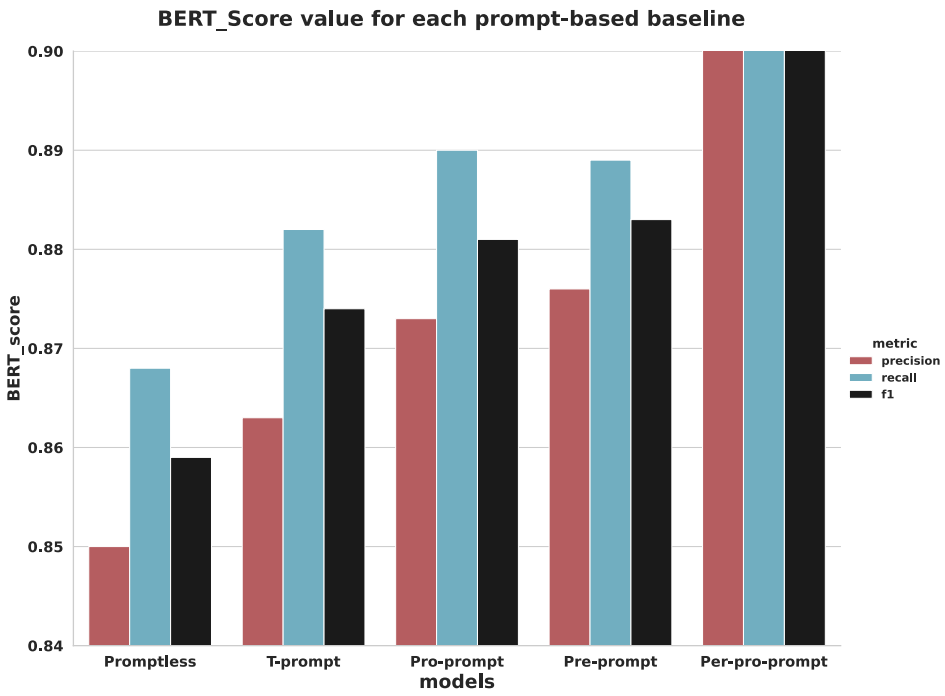

Fig. A.12. The BERT Score value for each prompt-based baseline.

The next section highlights the general working behaviour scenarios of our PPerMo approach.

A.7. The general working behaviour scenarios of our PPerMo framework

We build all the scenarios using the following User Model assumptions.

- **The user profile:** I like to live on the ground. The place I want to travel to most is India. I like kids. I like writing love letters very much. I like animation. I like sports. I like music very much. I really want to have sincere feelings. I think I am a loud voice. I like love very much.
- **The user movie-watching history:** "Movie1", ..., "the mind of the han and han princess", "lemon mouth", "undead debt collector", "live with integrity", "dumb boy".
- **The user personality:** AGR: 28%, OPN: 79%, CSN: 21%, EXT: 50%, NEU: 60%

Figures A.13, A.14, and A.15 highlights the performance of our proposed CRS on three different conversation recommendation scenarios (Chitchat and conversation scenario, Continuously Changing preferences scenario, Continuously Changing preferences with another user personality scenario).

A.7.1. First scenario: Chitchat and conversation scenario

Within this scenario, we assume that the user is a little bit sad and he misses his family. Figure A.13 highlights the response performance of our proposed model given all the above assumptions.

As we can see at the start of the conversation, our CRS is interacting with the user by chitchatting and answering the user's questions. Knowing that the user is feeling unwell and misses his parents, the CRS suggested recommending a movie about family and emotions "which is within the same topic of the conversation". Also, we can see that when the user expressed that he want to watch a movie about family, the CRS predicted an animation movie about the relationship between parents and children. The reason behind predicting an animation movie is that by monitoring the user profile information the CRS understands that the user likes the animation movies. Moreover, we can see that, when the user indicates that he wants to watch an American comedy movie, the CRS suggested "The Simpsons" which is both an American comedy and family animation movie. At the end of the conversation, we can see that the CRS provides a final-shot recommendation "The little prince movie" which is an animation family movie. The main reason for predicting this movie is that by monitoring the user's previous recently watched movies we found that he recently watched 2 family and children-type movies "Lemon mouth", "Dumb boy" and one movie that talks about a princess. Therefore, combining them with the "how to train your dragon" and "The Simpsons" movies which are animation movies, the CRS predicted a prince animation child

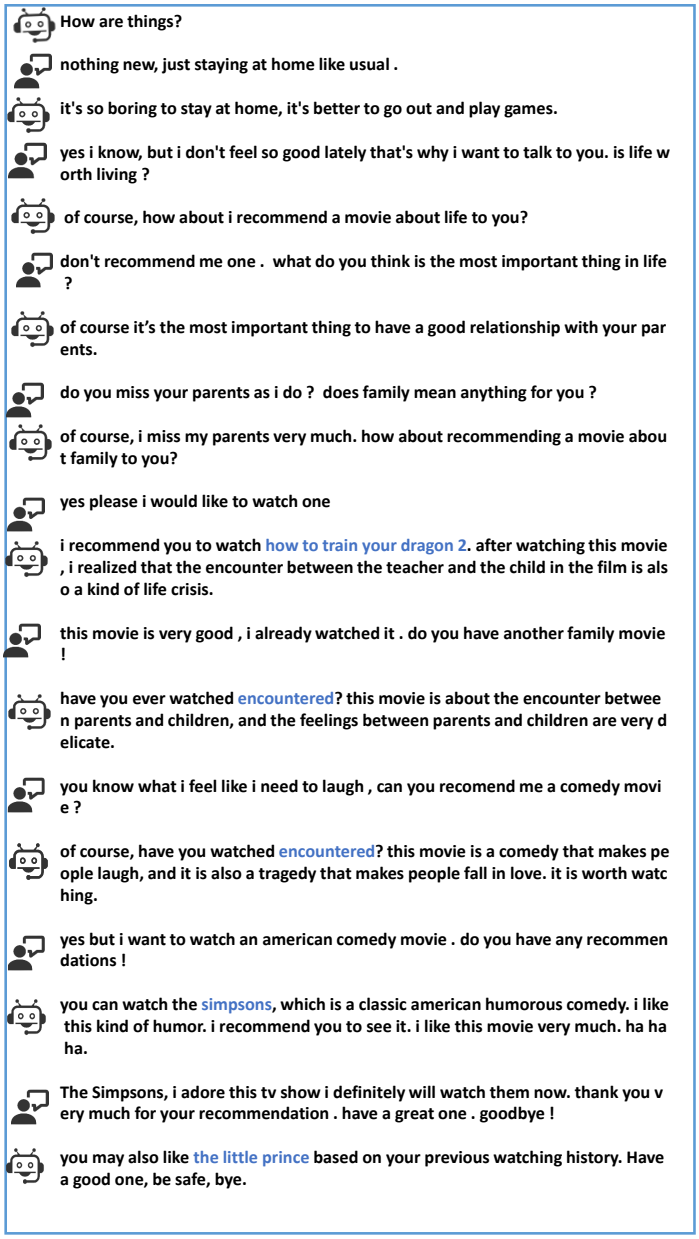

Fig. A.13. Our CRS Chitchat and conversation scenario.

movie "The little prince" as the final-shot recommendation which is also an emotional family movie.

A.7.2. Second scenario: Continuously Changing preferences scenario

Within this scenario, we kept the previous assumptions. However, in addition, we assume that the user is unaware of his preference and he keeps changing his preference continuously.

Figure A.14 highlights the response performance of our proposed CRS given the latest assumptions.

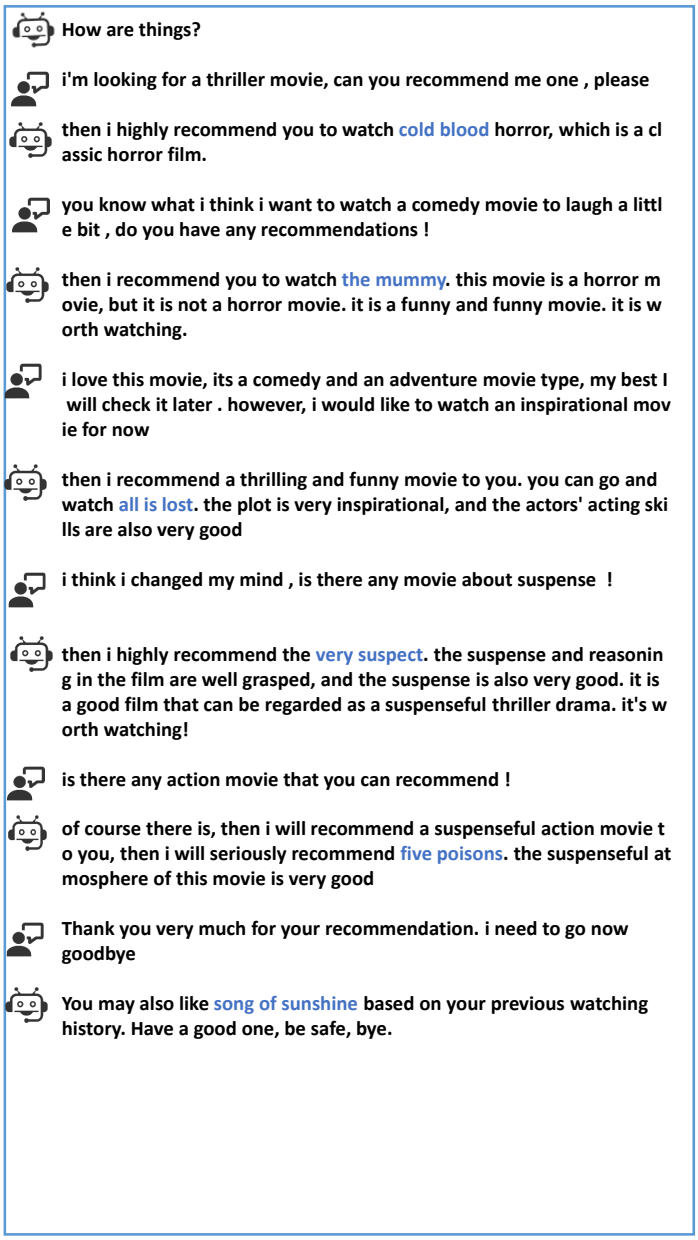

Fig. A.14. Continuously Changing preferences scenario.

Within this scenario the user is continuously changing his preferences and requesting different types of movies within each utterance. As we can see in the figure A.14, our CRS is capable of understanding the different user requests while keeping up with these requests by providing movies within the same or similar categories to the user's need. For example, when the user asked for a thriller movie, our CRS recommended the "Cold blood" movie which is a thriller/action/horror movie. Then when the user changed his

preference and asked for a comedy movie, our model predicted "The mummy" movie which is a horror/adventure/comedy movie. Then at the end of the conversation, our CRS predicted the "Song of sunshine" which is a thriller movie as a final-shot recommendation. The reason behind this recommendation is that within the conversation session the user kept expressing preferences for thriller-type movies "suspense, horror, thriller, and action movies". Also by monitoring the last movies watched by the user we found that he recently watched action and thriller-type movies such "Undead debt collector".

Therefore, combining the previous interactions (thriller, action) with the current interaction (thriller, action, horror) made the CRS refine the recommendations to finally predict the "Song of sunshine" thriller movie.

A.7.3. Third scenario: Continuously Changing preferences with another personality scenario

Within this scenario, we kept the same user's previous assumptions (as in the previous scenarios A.7). However, we changed only the user's personality from (AGR: 28%, OPN: 79%, CSN: 21%, EXT: 50%, NEU: 60%) to (AGR: 53%, OPN: 95%, CSN: 48%, EXT: 61%, NEU: 52%) to understand the impact of users' personality traits on the recommendation process. Figure A.15 highlights the response generation performance of our proposed PPerMo CRS given the new assumption.

By changing only the user personality traits we can see that the first two recommendations are the same as the previous scenario recommendations. This is because both users' personality traits share similar values for the OPN trait "where a high value of OPN means that the user tends to like having new adventures". This is the reason behind recommending "The mummy" adventure movie for both users. Also, we can see that when the user asked for an inspirational movie the CRS system provided a different movie compared to the previous scenario. This could be related to the higher AGR trait value compared to the previous scenario (a high AGR value means that the user has high pro-social and cooperating behaviours) which favors the recommendation of "The furious" car movie as it is a movie where a group of friends is cooperating to accomplish some suspenseful missions.
A demonstration of this scenario is presented in the PPerMo CRS link in the appendix Code Files Supplementary Material subsection A.7.5.

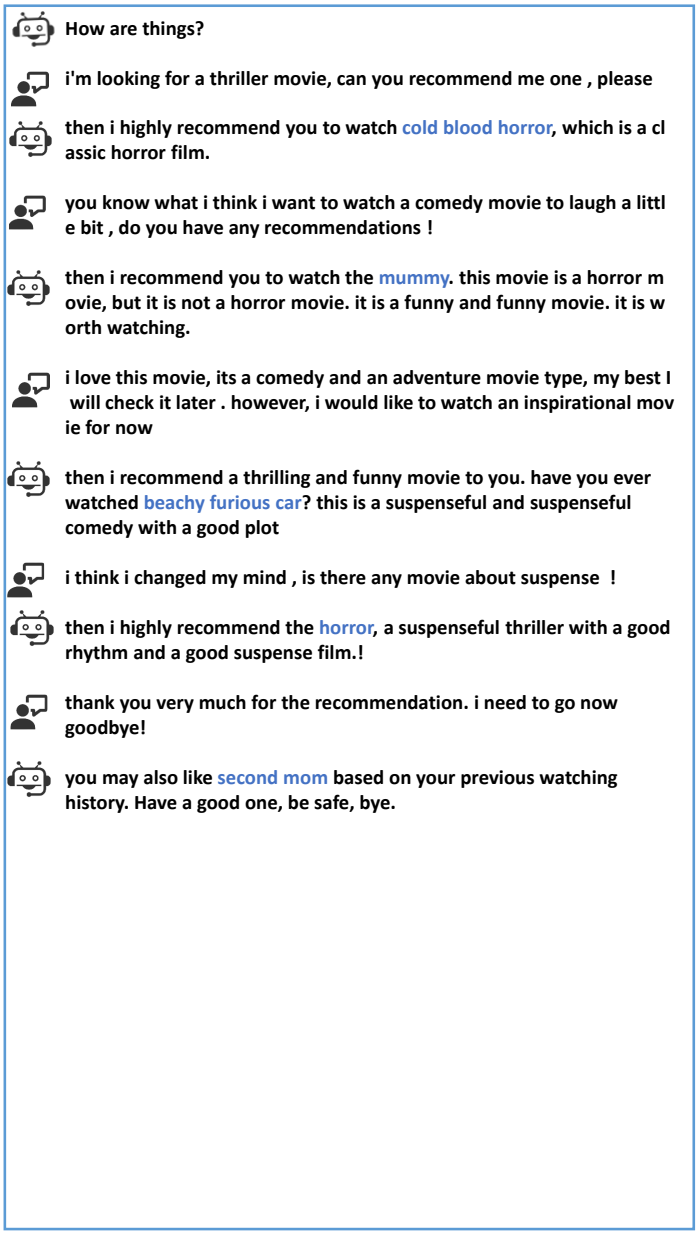

Fig. A.15. Our CRS Continuously Changing preferences with another personality scenario.

A.7.4. General Behaviour Of Our CRS

Despite the good performance of our CRS on different automatic evaluation metrics, we found that our CRS system still struggling with both recommendation and response generation tasks.

- **For the recommendation tasK:**
  We found that in some scenarios our proposed approach tends to recommend the same movie during the whole conversation when the movie is a multi-genre

movie. For example, if the user aims to watch a horror movie then the system might recommend the "Scream" movie. However, if the user changes his preference and requests a comedy movie, the system might also recommend the same movie "Scream" as it is both a comedy and a horror one. Also, as our conversation model keeps track of all previous utterances and topics during the dialogue session, it might sometimes stuck in recommending movies that fulfill previously discussed topics and not the current ones. For example if in all the previous utterances within a dialogue session the user tends to talk about family, but now he wants to get a movie about nature. Our CRS may not understand the user's current intent and it may recommend a movie about family as all the previous topics were talking about family. Also, sometimes when our CRS recommends a correct movie to the user, it might generate the wrong description for that movie. For example, if the user were talking about a horror movie in all the previous utterances and suddenly change his preference to watch a comedy movie. The CRS might recommend a comedy movie to the user but describe it as a horror movie. Figure A.16 shows a dialogue scenario that highlights different performance drawbacks of our proposed PPerMo framework.

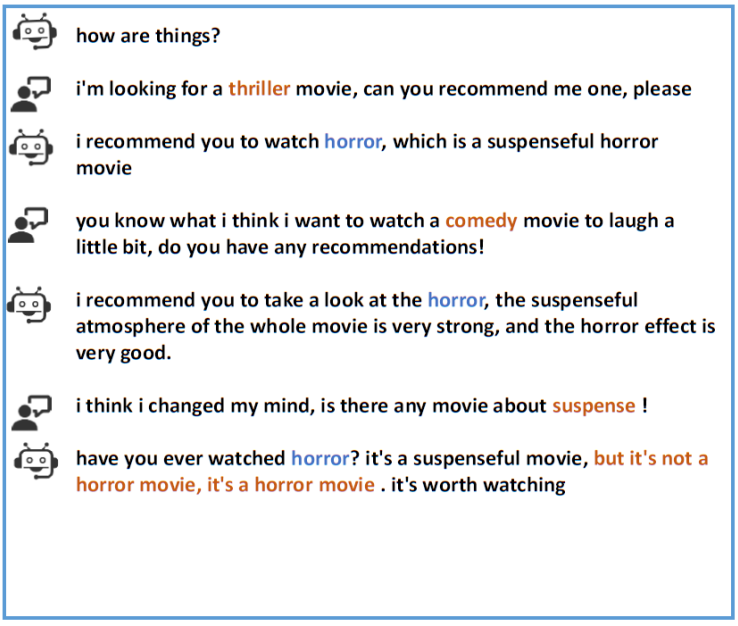

Fig. A.16. The performance drawbacks of our proposed PPerMo framework

The reason for such behaviour could be related to the complexity of our pre-trained model as we are using the small version of the DialoGPT model which does not encode sufficient contextual information compared to the large versions. Also, this could be related to the training steps where the number of steps that we used to

train our model could be insufficient to make the model effectively understands the related information within the different dialogue sessions.

- **For the response generation task:**
  By monitoring the generated responses within a conversation session we found that our CRS is generating utterances that are not grammatically correct, and sometimes it might even generate some non-sense words.

  The reason for such behaviour goes to the quality of our training dataset. As we have mentioned before in subsection 5.3.1, The TG-Redial dataset is a Chinese-based dataset and we used the Google translation API to translate it into the English language. After the translation process, we found that different translated utterances were grammatically incorrect which explains the utterance generation behaviour of our proposed CRS. More pre-processing and filtering techniques will be applied in future work.

A.7.5. Code Files Supplementary Material:

Personality prediction module: AWS-EP model
The Sequential Movie Recommendation module: DBT-SR model
The Dialogue State Tracking module: Elec-SP model
The Preference Generation module
The Dialog Generation module: PPPG-DialoGPT model
The Proposed modularized CRS Framework: PPerMo CRS