**Research Design and Methodology**

**Research Design**

The primary objective of this research is to enhance the explainability of recommendations through counterfactual analysis within an existing knowledge graph-based recommendation system. This study integrates counterfactual analysis into a pre-existing recommender system as described in the literature. This framework's unique aspect is that it is adaptable to any knowledge graph-based recommender system, provided that the system supports path-based recommendations. This method contributes to the broader field of recommendation systems research by providing deeper insights into the causality and reasoning behind recommendations.

The theoretical basis for employing counterfactual analysis in recommendation systems lies in its ability to generate "what-if" scenarios, thereby enhancing the transparency and explainability of recommendations. The human cognitive process inherently seeks counterfactual scenarios for enhanced understanding. Relevant literature and theories support this approach by demonstrating how counterfactuals can reveal hidden patterns and dependencies in data, thereby making recommendations more understandable and actionable for users.

**Data Collection and Preparation**

For this study, we utilize the implementation of the knowledge graph-based recommender system known as CAFÉ (Coarse-to-Fine Neural Symbolic Reasoning for Explainable Recommendation). CAFÉ integrates knowledge graphs with neural symbolic reasoning to enhance e-commerce recommendations. It operates on a coarse-to-fine principle, initially creating user profiles from historical data that outline general user behaviors. These profiles then guide the system in navigating the KG to generate reasoning paths that lead to specific item recommendations. This method not only improves recommendation accuracy but also provides clear explanations for why items are recommended by tracing the reasoning paths in the KG.

**Knowledge Graph Construction**

The knowledge graph is constructed using data from the existing recommender system, which includes entities such as users, products, brands, and words used in reviews, as well as related products that have been bought together, viewed, or also bought. Relationships like 'purchase', 'mentions', 'produced by', and 'described by' are included. This process involves:

* Representing the knowledge graph as a dictionary of entities and relationships, and including the reverse of each relationship.
* Connecting each user to the words they have mentioned and products they have purchased. Products are linked to the brands they are produced by, the categories they belong to, the words they are described by, and related products.

**Metapaths**

Metapaths refer to predefined paths in the knowledge graph that represent sequences of relations connecting different entities (like users and items). These paths are structured to reflect potential behavioral or relational patterns between entities relevant to making recommendations. The reasoning process using these metapaths operates as follows: The system first generates a coarse sketch of user behavior by identifying prominent user-centric patterns from historical data. These patterns, essentially metapaths, highlight typical ways users interact with items or other entities in the knowledge graph. Using the metapaths outlined in the user profiles as a guide, the system then performs fine-grained path reasoning. This involves a path-finding algorithm that navigates through the knowledge graph, starting from a user entity and moving through the specified entities and relations. The goal is to reach item entities that best match the user’s profiled behavior. This path reasoning is augmented by neural symbolic reasoning modules, which assist in making decisions at each step of the path (like choosing which relation to follow next based on the current context). This combination of structured metapaths and neural decision-making enables the system to efficiently and effectively find relevant items, offering explanations for recommendations based on the sequences of relations traversed.

**Recommender System Integration**

To integrate the knowledge graph into the framework, I have extended the implementation of CAFÉ’s knowledge graph to enable the handling of a subgraph of the original, very large system. Sampling is based on the number of users and is performed using a snowball strategy, sampling connected products and their attributes. A pruning step follows this, which eliminates all entities that are not in a complete relationship with the sampled knowledge graph. For example, entities where only one side of a relationship is sampled, and the reverse does not exist, are removed. Since the CAFÉ recommender system uses predefined embeddings, after pruning, the conversion of IDs from old to new is performed.

This revised version improves clarity, flow, and academic tone, and structurally it emphasizes a logical progression from general objectives to specific methodologies and integration steps.

**Framework Design**

The counterfactual analysis begins by examining the top 10 recommendations provided by the recommender system. For a given recommended path—which comprises entities and relationships defined by the recommender system's predefined metapaths—this path serves as the input for the counterfactual analysis. The analysis focuses on first-level attributes and entities that are connected to the attributes and related products of the recommended path, ensuring that the selections are highly relevant to the product under consideration.

The recommender system is designed to learn the tastes and behaviors of the user based on their purchased products, which is reflected in the design of the metapaths. After identifying related entities and attributes, a specific metapath associated with each entity is selected for analysis. For example, to explore whether a product would still be recommended if it were associated with a different brand, the metapath might be: user → purchase → product → produced\_by → brand → recommended product. The viability of this scenario is assessed by calculating the average score of the steps along the path; if this score exceeds the score of the least recommended product (the 10th product), the scenario is considered plausible.

**Knowledge Graph Entity Stats**

Some nodes in the knowledge graph, such as those connected to a "beauty" category, exhibit high connectivity degrees but provide limited specific insight for the analysis due to their general nature. These nodes can create computational overhead. To address this, we calculate the connectivity degree for each entity across all related entity types. Nodes with z-scores exceeding a specified threshold (typically set at 1) are identified for potential exclusion from the analysis to streamline computations.

**Community Identification**

To address the computational overhead of some selected entities, for example words that are relatively higher in number connected to products, the study furthers the filtering process. If the number of selected entities exceeds a specified threshold, filtering occurs based on their community membership relative to the recommended product.

To identify these communities within the knowledge graph, we employ the Louvain method. This method optimizes modularity, effectively grouping nodes into communities where connections are denser internally than with external nodes. The iterative process starts with each node as its own community and progressively merges them to maximize modularity. This is particularly beneficial for your counterfactual analysis as it helps identify clusters of products with similar attributes or relationships. Understanding these community structures allows you to analyze how alterations in product attributes might impact recommendation patterns, providing deeper insights into the factors that influence product categorization and recommendations.