Guided Text-based Item Exploration

Behrooz Omidvar-Tehrani, Aurelien Personnaz, Sihem Amer-Yahia

▶ **To cite this version:**

Behrooz Omidvar-Tehrani, Aurelien Personnaz, Sihem Amer-Yahia. Guided Text-based Item Exploration. Proceedings of the 31st ACM International Conference on Information & Knowledge Management, 2022, pp.3410-3420. 10.1145/3511808.3557141 . hal-04600295

HAL Id: hal-04600295
https://hal.science/hal-04600295

Submitted on 4 Jun 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Guided Text-based Item Exploration

Behrooz Omidvar-Tehrani
Amazon (USA)

Aurélien Personnaz
Univ. Grenoble Alpes (France)

Sihem Amer-Yahia
CNRS, Univ. Grenoble Alpes (France)

ABSTRACT

Exploratory Data Analysis (EDA) provides guidance to users to help them refine their needs and find items of interest in large volumes of structured data. In this paper, we develop GUIDES, a framework for guided Text-based Item Exploration (TIE). TIE raises new challenges: (*i*) the need to abstract and query textual data and (*ii*) the need to combine queries on both structured and unstructured content. GUIDES represents text dimensions such as sentiment and topics, and introduces new text-based operators that are seamlessly integrated with traditional EDA operators. To train TIE policies, it relies on a multi-reward function that captures different textual dimensions, and extends the Deep Q-Networks (DQN) architecture with multi-objective optimization. Our experiments on Amazon and IMDb, two real-world datasets, demonstrate the necessity of capturing fine-grained text dimensions, the superiority of using both text-based and attribute-based operators over attribute-based operators only, and the need for multi-objective optimization.

CCS CONCEPTS

• **Information systems** → **Data management systems**; • **Human-centered computing**; • **Computing methodologies** → **Machine learning**; **Artificial intelligence**; **Information extraction**;

KEYWORDS

exploratory data analysis, reinforcement learning

ACM Reference Format:
Behrooz Omidvar-Tehrani, Aurélien Personnaz, and Sihem Amer-Yahia. 2022. Guided Text-based Item Exploration. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557141

1 INTRODUCTION

Many decisions rely on a mix of structured and unstructured content such as item attributes and reviews [2, 37]. In online shopping platforms like Amazon, users read reviews to make an informed decision about products to purchase. Avid moviegoers get their inspiration for the next movie by dissecting reviews in IMDb. These domain actors need to navigate in a very large space of textual data and switch back and forth between items and text to find their target. While Exploratory Data Analysis (EDA) has received increased attention, its application to textual data has not been studied yet. In this paper, we develop GUIDES, a framework for guided Text-based Item Exploration (TIE).

The main motivation behind EDA is that there exist many scenarios where the target cannot be described with a clear single-shot query. Employing the traditional search-by-query paradigm is not optimal in such scenarios. The goal of EDA is to propose a multi-shot path leading to target instead of the target itself, so that users can make an educated decision following what they observed along the way. TIE is a specific case of EDA where users do not know precisely what they are looking for and generally only have partial knowledge of the underlying attributes of items and the textual content. The literature contains several solutions for EDA [15, 48–50, 55, 56]. These methods require purely structured data and they shape interactivity through navigating in the space of item attributes. In TIE, people rely on reviews as a source of collective intelligence to guide them in their decision making process. The natural form of interactive information seeking in TIE is through *text*, i.e., the user decides what to see next based upon reading a textual content, e.g., a review, and by exploring structured and unstructured content in tandem.

The EDA literature recognizes three settings: (*i*) *manual* where the system provides an exploration toolbox and the decision making is up to the user, (*ii*) *fully-guided* where the system makes all the decisions and the user is a mere observer, and (*iii*) *partially-guided* where the user can override the system's recommendations. Due to the inherent ambiguity of textual data, the design of a TIE system must follow a partially-guided setting where the user and the exploration system work hand-in-hand to reach target.

Example. Steve wants to buy a high-quality camera for his upcoming nature photo-shooting. *Steve cannot form a precise query to express his need, as he's unsure what attributes matter in his purchase. He needs to explore a cherry-picked set of items and reviews so that he can make an informed decision.* Based on his friends' recommendations, Steve starts browsing cameras in Amazon that have a high optical view and processing power. Figure 1 depicts the exploration steps suggested to Steve by a TIE system. Using his initial query, the system picks some relevant cameras. Steve reviews the suggestions and focuses on a Fujifilm X100V camera. The system then suggests a representative set of reviews for the selected camera. This helps Steve get a better understanding of the pros and cons of that product, such as improved lens resolution and fragility of the camera. A review on "autofocus issues" attracts his attention, as this is an important matter for capturing photos in nature. He intervenes and asks the system for other reviews on the same topic. In the results, Steve notices a review that suggests Canon cameras as better options for autofocus. He selects that review and the system recommends a few cameras related to that review among which Steve discovers a Canon 6D as his final target.

Our example raises two main challenges. Without a TIE system in action, the user must visit a large space of items and their associated

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '22, October 17–21, 2022, Atlanta, GA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557141

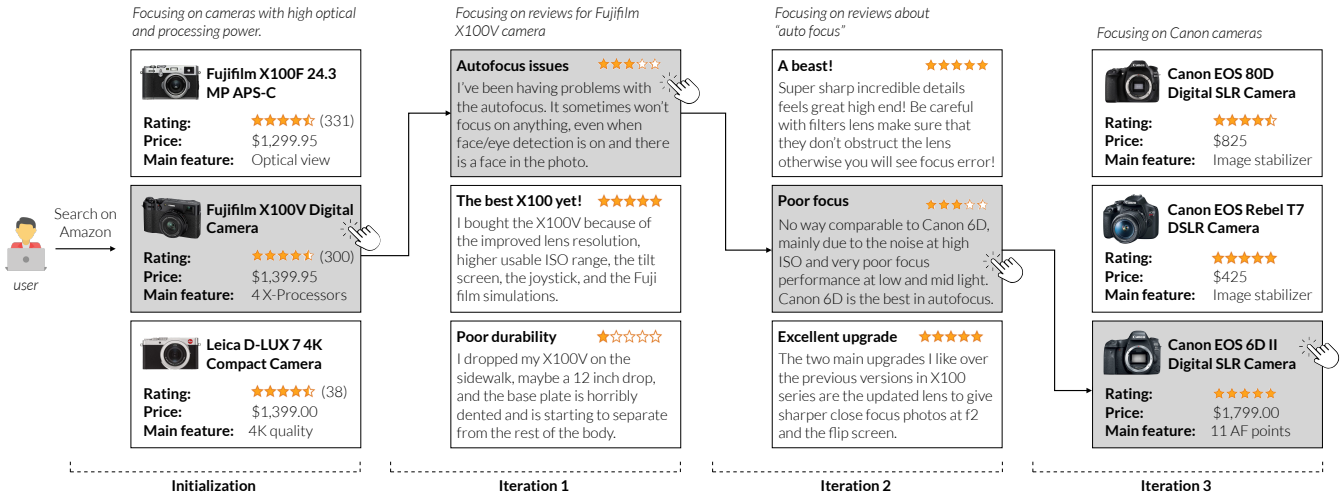

Figure 1: Example of Text-based Item Exploration (TIE).

reviews to make an informed decision. Additionally, the explorer has to draw insights from unstructured text by extracting topics, sentiments and summaries. If done manually, this process is tedious and prone to noise.

Our *first contribution* is to formalize the TIE problem. The exploration task is *compound*, which means that the exploration session does not necessarily terminate with binary outcomes of success or failure, but with collecting a subset of the target. While it is ideal to find the target set in its entirety, it is often infeasible in practice due to the inherent complexity of information seeking tasks, and to the sheer amount of available data. We model TIE as a Markov Decision Process (MDP) where states represent a mix of structured and unstructured content that is being explored at a given step, and transitions are triggered by operators [15, 48, 55]. To enable that, we represent text with several dimensions including topics, tags, sentiments, and summaries. In TIE, the reward of transitioning between states is *multi-valued* as it must capture multiple textual dimensions. For instance, a review might have high utility in terms of its topic and low utility when it comes to sentiment. The TIE problem seeks to find the highest utility exploration policy for a target set of items.

Our *second contribution* is computational. Given that exploration logs are unavailable in TIE and that transition probabilities between exploration states are unknown, model-free Reinforcement Learning (RL) [61] fits our context. We leverage a Deep RL (DRL) algorithm called Deep Q-Networks (DQN) [42] as the learning component of GUIDES to obtain TIE policies. We propose to modify the DQN architecture to account for the multi-objective optimization of rewards, where each reward dimension acts as one optimization objective. The modified DQN algorithm discards the exploration iterations with dominated rewards in its sampling strategy and trains policies only based on non-dominated iterations. As a result, it outputs the most rewarding actions at each exploration step.

Our *third contribution* is a thorough empirical investigation on TIE variants and EDA baselines with two real datasets, Amazon [9, 44] and IMDb [3]. We address three questions: *"are our TIE actions expressive enough to navigate in textual data and reach target?"*, *"does multi-reward optimization improve learning?"*, and *"does our approach scale with increasing data size and task complexity?"*. Our first finding is that the best results are attained when both text-based and attribute-based operations are integrated. Our second finding is that multi-reward optimization reduces the number of iterations by up to 30% due to their dominated rewards. Finally, we find that our approach is scalable as we increase data size and exploration complexity.

Paper organization. We discuss related work (Section 2), present our TDE model and problem (Section 3), develop our solution (Section 4) and experiments (Section 5), and conclude in Section 6.

2 RELATED WORK

To the best of our knowledge, GUIDES is the first to seamlessly integrate structured and unstructured exploration, and provide guidance in the dual space of items and text. We review interactive guidance approaches and text-based information seeking.

2.1 Interactive guidance

Enabled by data exploration approaches [17], interactive guidance is a novel paradigm of information seeking, where users are inside the analysis loop and their feedback is incorporated on-the-go to receive relevant guidance in future iterations of the exploration. Depending on the source of guidance, these systems are categorized as action-driven [7, 8, 13, 15, 31, 49, 55, 66, 69, 71] or example-driven [5, 7, 10, 12, 25, 43, 54, 58, 76].

Action-driven guidance. Most approaches rely on previously collected logs (e.g., SQL [14, 28, 46] and OLAP [1, 39] query logs) to recommend optimal query-based actions at run-time. Faceted search [31, 66, 71] is often leveraged to improve run-time experience, where users explore data using the categorical knowledge-based actions, called facets. In case logs are scarce,, automated EDA systems [10, 15, 49, 55, 69] simulate user interactions in an offline process to train optimal exploration policies using (deep) Reinforcement Learning (DRL). Automated EDA systems play an important role in decision-making as they guide users to pick an appropriate next action without the need for logs. However, these systems

have been designed for structured data only. *GUIDES adopts an action-driven approach. Additionally, the unstructured content of text is incorporated and the exploration is performed over the dual space of items and reviews. While most EDA systems leverage DRL as a black-box, GUIDES opens that box and refines the reward computation mechanism to increase the quality of recommended actions.*

Example-driven guidance. These approaches enable guidance by providing an input example and recommending "interesting" examples. For instance, the approach in [25, 33] recommends different OLAP drill-downs on a table. In [12, 20, 57, 67, 68], exploration examples are provided in the form of visualization. The approach in [10, 22] takes labeled examples and leverages active learning to learn a user query. *In GUIDES, we learn the exploratory behavior without any prior assumption about context or interaction history. Moreover, the multi-valued reward system of GUIDES generalizes interesting dimensions of an exploration state.*

2.2 Text-based information seeking

To seek information in textual form, prominent natural language understanding (NLU) approaches are subjective databases [2, 16, 37, 40, 41, 70, 72, 74, 75], conversational agents [11, 35, 59, 60, 64] and question answering systems [26, 73].

Subjective databases. OpineDB [37], Voyageur [16], and SubDEx [2] operate as mediators between unstructured data (i.e., the raw text) and structured queryable data (e.g., a relational database) by enabling a querying mechanism for subjective data. To interpret subjective data, these systems require various domain-specific annotations and parameter tuning (e.g., setting up subjective attribute-value markers and aspect-opinion pairs) which limit their generality. Moreover, as DBMSs, these systems do not natively enable multi-shot and exploratory scenarios. *GUIDES relies on a generic data model that can accommodate new datasets. Also, the multi-valued textual reward captures different dimensions of text without any prior domain knowledge.*

Conversational agents. Conversational agents aim to provide a meaningful goal-oriented dialog with the user. They are built using statistical methods and neural dialog models [11] equipped with an attention mechanism [35]. Examples include chit-chat systems (interact with human-like reasonable responses), informational chat systems (help users find information), and goal-oriented chat systems that help users accomplish a specific task, e.g., making a restaurant reservation [60]. *In GUIDES, we leverage textual data for a different purpose, that is, to elicit preferences of users with the goal of learning their optimal exploratory behavior.*

Question answering systems. These systems are implemented as an instance of a sequential transformation (Seq2Seq) using recurrent neural networks (RNNs) [26, 73]. These systems take raw text and output a sequence of potential questions and answers represented by four text annotations: answer span, clue span, question style and the question itself [27, 32, 38]. Question-answer conversations terminate when a target element is found [73]. The focus is hence on the end of the session and no optimization is performed to improve exploration quality during a session. Because they leverage supervised learning, these systems require a rich ground-truth for question-answer generation [4, 36, 51, 62], e.g., ReDial with 10K question-answer pairs on movie recommendation [36] and CoQA with 127K questions from 8000 conversations [51]. *GUIDES benefits from a value-based learning of exploratory behavior where the optimization encompasses the whole exploration session, and without the need for a ground-truth. Also, it operates on the dual space of items and reviews instead of a single question-answer space.*

3 TEXT-BASED DATA EXPLORATION MODEL

We model items and reviews (Section 3.1) and present our exploration model (Section 3.2). Then we formally define the TIE problem (Section 3.3).

3.1 Review model

We consider a set of items $\mathcal{I}$ and textual reviews $\mathcal{R}$, where each review r is written for a specific item i, s.t., $\forall r \in \mathcal{R}$, $item(r) = i \in \mathcal{I}$. When referring to an item or a review, we use the term "object".

Object attributes. Objects are identified with a set of attributes, $\phi(i) = \{\langle a, v\rangle\}$ and $\phi(r) = \{\langle a, v\rangle\}$, where $a \in \mathcal{A}$ and $v \in domain(a)$. Examples of item attributes are product category and movie genre, and examples of review attributes are helpfulness score and review time.

Object signature. We define a review signature as a structured set of attributes that represent the text of a review with different semantics and at different abstraction levels. The signature of a review $r \in \mathcal{R}$ is a vector $\eta(r) = \langle \kappa_1, \kappa_2, \ldots \kappa_{|\mathcal{K}|}\rangle$, where $\kappa_i \in \mathcal{K}$ is itself a vector that represents one dimension extracted from the text of the review, e.g., summary, sentiments, tags, and topics. Figure 2 shows the signature of a review for the movie "A Midnight in Paris". We also define the signature of an item $\eta(i)$ over its description, e.g., the script of a movie in IMDb and the item description in Amazon.

Object profile. Given a review $r \in \mathcal{R}$, $profile(r) = \langle \phi(r), \eta(r)\rangle$ consists of the attributes and signature of r, representing the structural and textual dimensions of the review, respectively. An item profile is defined similarly and denoted as $profile(i) = \langle \phi(i), \eta(i)\rangle$.

Object similarity. Given two objects $x, x' \in \mathcal{I} \cup \mathcal{R}$, we measure their similarity as follows:

$$\begin{aligned} sim(x, x') &= sim(profile(x), profile(x')) \\ &= (attribsim(\phi(x), \phi(x')), textsim(\eta(x), \eta(x'))) \end{aligned} \quad (1)$$

where $attribsim(\phi(x), \phi(x'))$ is the pairwise similarity between the attribute values of two objects and is typically performed using a Cosine over their attribute vectors, resulting in a scalar value in the range $[0, 1]$. The pairwise similarity between textual dimensions $textsim(\eta(x), \eta(x'))$ is computed over the vectors $\eta(x).\kappa_i$ and $\eta(x').\kappa_i$ for all dimensions $\kappa_i \in \mathcal{K}$:

$$textsim(\eta(x), \eta(x')) = Cosine(\eta(x).\kappa_i, \eta(x').\kappa_i);\ \forall \kappa_i \in \mathcal{K} \quad (2)$$

The result of this similarity computation is a vector of size $|\mathcal{K}|+1$ since x and x' will be compared according to their attributes, as well as their $|\mathcal{K}|$ textual dimensions. Given the multi-valued nature of the similarity function, we denote the j-th component of the output vector as $sim_j(x, x')$, where $j \in [1, |\mathcal{K}| + 1]$.

3.2 Exploration model

We define an exploration session as a sequence of iterations obtained by applying a generic exploration function defined as follows:

$$explore(x, \mathcal{X}, k) \rightarrow X \tag{3}$$

The exploration function admits an object $x \in \{i, r\}$, a space $\mathcal{X} \in \{items, reviews\}$ and an integer k, and returns a subset $X \subseteq \mathcal{X}$ of size k, that acts as input to the next iteration. For instance, $explore(i_2, reviews, 5)$ returns 5 reviews for further exploring i_2.

Components of the exploration function. We define the semantics of the exploration by specifying two conditions on the output X of an exploration function as follows:

- $\forall x' \in X, relevance(x, x', \delta) \geq \sigma$;
- $quality(X)$ is maximized.

Both functions return a value in the range $[0, 1]$ to capture the interestingness of the exploration [19, 30]. The *relevance* function captures that the results of iteration $t+1$ should be related to what the user experienced in the previous iteration t. Given X, the result of $explore(x, \mathcal{X}, k)$, where $|X| \leq k$, we define relevance between x and X as $relevance(x, X, \delta) = sim_\delta(x, x'), \forall x' \in X$ over one specified similarity dimension δ. The parameter δ can be either text-based or attribute-based. In the former, $\delta \in \mathcal{K}$ and relevance is computed on the signatures of x and all objects in X. In the latter, it is computed on their attributes. The *quality* function is independent of the previous iterations and only focuses locally on the quality of the exploration results. Examples of quality functions are diversity and coverage (more details are provided in Section 4.1).

Modeling TIE. We model TIE as a Markov Decision Process (MDP) comprising a quadruple (S, E, P, R) where:

- S is a discrete set of exploration states;
- E is a set of exploration actions, where each action instantiates $explore(x, X, k)$ and enables a transition between consecutive exploration states s_t and s_{t+1};
- $P(s_{t+1}|s_t, e_t)$ are the probabilities that the exploration action e_t will change state $s_t \in S$ to state $s_{t+1} \in S$;
- $R(s_{t+1}|s_t, e_t)$ are rewards for transitioning from state $s_t \in S$ to $s_{t+1} \in S$ by applying the exploration action e_t.

The exploration state $s_{t+1} = \langle x_{t+1}, X_{t+1} \rangle$ is obtained by applying action e_t to a previous state $s_t = \langle x_t, X_t \rangle$. The exploration process goes on by selecting another action e_{t+1}. The probabilities $P(s_{t+1}|s_t, e_t)$ reflect the behavior of the environment ($\mathcal{I}$ and $\mathcal{R}$) when exploration actions are applied, i.e., they represent what is displayed to the user in iteration $t+1$, if e_t is applied. To capture the complexity of text-based exploration, we consider a *model-free* setting [61], where the probabilities are not known a priori, and depend on the relevance and quality of the exploration function. An *exploration session* $\mathcal{S}$ is a sequence of exploration states and actions $\mathcal{S} = [(x_1, X_1, e_1), \ldots, (x_n, X_n, e_n)]$, where $n = |\mathcal{S}|$. An *exploration policy* π is a mapping function from an exploration state $s_t = \langle x_t, X_t \rangle$ to an action e_t, where $\pi(s_t) = e_t$. An exploration session generated by π is $\mathcal{S}^\pi = [(s_1, \pi(s_1)), \ldots, (s_n, \pi(s_n))]$.

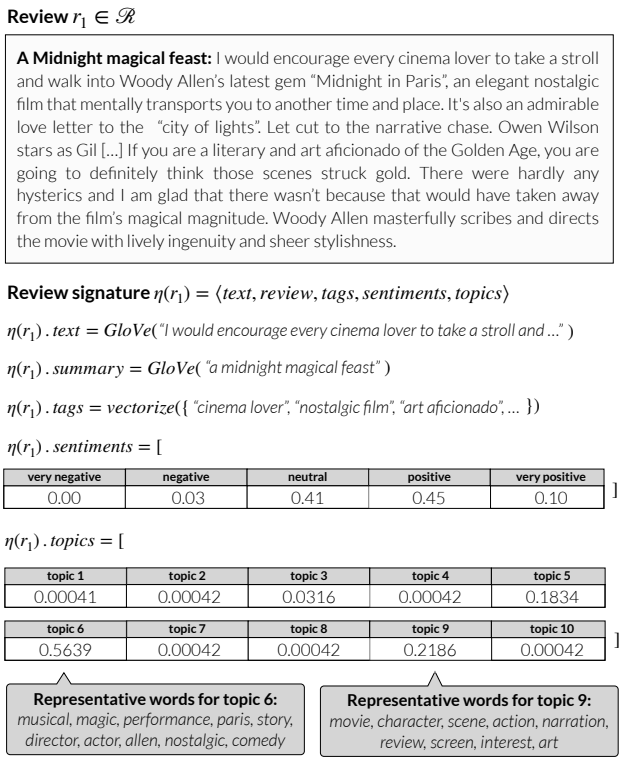

Figure 2: Signature of a review for "A Midnight in Paris".

3.3 TIE problem definition

In TIE, the objective of the explorer is to collect a target set of items and some of their reviews throughout the exploration session to enable decision making [46, 73]. We denote the set of target objects as $\mathcal{T}$. The exploration task is *compound*, which means that the exploration session does not necessarily terminate with binary outcomes of success or failure, but with collecting a subset of $\mathcal{T}$ throughout a session. While it is ideal to find the target set in its entirety, it is often infeasible in practice due to the inherent complexity of information seeking tasks.

Utility of objects. Given an object x and a target set $\mathcal{T}$, we compute the utility of x as follows:

$$utility(x, \mathcal{T}) = \sum_{x^* \in \mathcal{T}} sim(x, x^*) \tag{4}$$

Utility of an exploration policy. The utility of an exploration policy π is defined as follows:

$$policy\_utility(\pi, \mathcal{T}) = \sum_{(x_t, X_t, e_t) \in \mathcal{S}^\pi} \left(\gamma^t \sum_{x \in X_t} utility(x, \mathcal{T}) \right) \tag{5}$$

Policy utility is computed as the accumulation of the utility values for the objects of X_t in each iteration of $\mathcal{S}^\pi$, i.e., the session generated by π. The values are discounted by the parameter $\gamma \in [0, 1]$. Policy utility depends on the explorer's goal encapsulated in $\mathcal{T}$. In TIE, users' goals are subjective since users do not have a clear

idea of attributes that characterize their target, hence they need to navigate in the space of items and reviews. Training different policies ensures the applicability of our solution to different goals. We discuss different instances of user goals in Section 5.

Problem definition. The TIE problem is formulated as finding an optimal policy π^* with the highest utility for a target set $\mathcal{T}$.

$$\pi^* = argmax_\pi policy\_utility(\pi, \mathcal{T}) \tag{6}$$

To optimize Equation 6, we simulate a human TIE experience as an offline process to learn an optimal exploration policy. During that process, the policy gets updated after each interaction with items and reviews via exploration actions.

Reward system. The reward of a state $R(s_{t+1}|s_t, e_t)$ must reflect the utility of X_{t+1}. Given the compound targets in TIE, a positive reward is received every time some objects belonging to the target are discovered. To capture that, we use a Reward Machine [23] with three types of rewards: neutral reward, target reward, and similarity reward. If a set of domain-dependent rules Ω is violated in s_{t+1}, a *neutral reward* (e.g., 0) is given. If Ω holds and some of the target objects exist in X_{t+1}, a *target reward* (e.g., 1) is given. If Ω holds but no target is reached, a *similarity reward* is given proportionally to the degree of progression towards $\mathcal{T}$. This Reward Machine is formalized as follows:

$$R(s_{t+1}|s_t, e_t) = \begin{cases} [0]^{\mathcal{K}+1} & \text{if } \Omega(s_{t+1}) = False \\ [|X_{t+1} \cap \mathcal{T}|]^{\mathcal{K}+1} & \text{else if } X_{t+1} \cap \mathcal{T} \neq \emptyset \\ utility(X_{t+1}, \mathcal{T}) & \text{otherwise} \end{cases} \tag{7}$$

Equation 7 represents a multi-valued reward system, because the utility function (Equation 4) outputs a vector of size $\mathcal{K}+1$. This is a natural representation for TIE, as the textual content is identified with different exploration dimensions. For instance, a review might have high utility in terms of its topic dimension and a low utility when it comes to the sentiment dimension. In Section 4, we describe how policies can be processed in a multi-valued reward system.

TIE as an RL problem. Following the MDP model, our TIE problem is reformulated as finding a policy $\pi : S \rightarrow E$ such that it maximizes the discounted cumulative reward $\hat{R}$:

$$\hat{R} = \sum_t \gamma^t R(s_{t+1}|s_t, e_t) \rightarrow max \tag{8}$$

In Equation 8, $e_t = \pi(s_t)$ and $\gamma \in [0, 1]$ is a discount factor. As in a typical RL problem, we define our value function (Equation 9) and action-value function (Equation 10) for a given policy π.

$$V_\pi(s) = \mathbb{E}\Big[\sum_{l=0}^{\infty} \gamma^{t+l} R(s_{t+l+1}|s_{t+l}, e_{t+l})|s_t = s\Big] \tag{9}$$

$$Q_\pi(s, e) = \mathbb{E}\Big[\sum_{l=0}^{\infty} \gamma^{t+l} R(s_{t+l+1}|s_{t+l}, e_{t+l})|s_t = s, e_t = e\Big] \tag{10}$$

In Equation 9, $V_\pi(s)$ computes the expected cumulative reward of policy π obtained after observing state s at iteration t. The function $Q_\pi(s, e)$ in Equation 10 captures the expected cumulative reward that π gets from applying action e at state s. An optimal policy π^* always selects the exploration action with the highest value in the current state, thus maximizing expected reward. This yields an optimal function V^* and Q^* which satisfy the Bellman optimality equations [61]. Our goal is then to find an optimal policy π^* which yields the best exploration action at every exploration iteration.

Quality function	Formulation
Numerical diversity *(numdiv)*	$numdiv(X) = stdev(\{\eta(x').score, \forall x' \in X\})$
Textual diversity *(textdiv)*	$textdiv(X) = average(1 - Cos_{x',x'' \in X}(\eta(x'), \eta(x'')))$
Coverage *(coverage)*	$cov(X) = \sum_{x' \in X} word\_count(x')$
None *(none)*	$none(X) = \mathbb{1}\boxed{U = relevance(x, X)[:k]}$

Table 1: Instances of the quality function in GUIDES

4 THE GUIDES FRAMEWORK

We describe our proposed framework called GUIDES, a value-based RL solution to learn policies for TIE.

4.1 Instantiating TIE actions

In GUIDES, different quality functions combined with different relevance dimensions (textual dimensions and attributes) engender different semantics for the exploration function. For instance, in case where quality is defined as *diversity* and δ = *topics*, the exploration aims to find a diverse set of exploration options X whose topics are highly similar to the current object x. Below we present how we instantiate relevance and quality in GUIDES.

Instantiating relevance. The relevance component has $|\mathcal{K}|$ text-based and one attribute-based variants. Given an object x, its signature $\eta(x)$ is built over the schema of a set of textual dimensions $\kappa \in \mathcal{K}$. In this work, we pick a set of textual dimensions from the NLU literature [21], $\mathcal{K}$ = (*text*, *summary*, *tags*, *topics*, *sentiments*) ordered from very specific to more general representations. However, the model is generic and other dimensions such as aspects and opinions can be incorporated.

Building textual dimensions. The textual content of an object x is vectorized using the GloVe pre-trained language model [47] resulting in $\eta(x).text$. Among different language models, we choose GloVe as it leverages both global and local statistics of a corpus and hence fits the modeling of short texts such as item descriptions and reviews. A more general and informative representation is the text summary [58] whose vectorized version is captured in $\eta(x).summary$. One direction of our future work is to incorporate attention layers [73] and sentence-level transformers [53] to capture the inter-connections between the parts of text.

Building tags. Tags are part-of-speech (POS) patterns that include a noun and can be entirely replaced by a noun. This is a more general representation compared to text and summary. For a given object x, the set of tags $tags(x)$ acts as a distilled version of x focused on discussion points in the textual content of the object. The dimension $\eta(x).tags$ is the vectorized version of $tags(x)$ using the GloVe model.

Building topics. The dimension $\eta(x).topics$ is the topic distribution of x. It is a vector of size nt obtained using Latent Dirichlet Allocation (LDA) for topic modeling, using nt topics. In Figure 2, we set $nt = 10$ and observe that the review on the movie "A Midnight in Paris" is highly related to topics 6 and 9, where the former is

around words like "magic" and "nostalgic", and the latter is about "movie", "character" and "scene". One direction of our future work is to leverage attention-based autoencoder topic modeling [34, 63, 77] for a more effective modeling of topics in short texts.

Building sentiments. For a given object x, $\eta(x).sentiments$ is a sentiment distribution over the sentences of x. It is a vector of size nb that is defined as follows:

$$\eta(x).sentiments = temp\_softmax(bucketize_{nb}[pol(\rho), \forall \rho \in sentences(x)])$$

In the above equation, $pol(\rho) = [-1, 1]$ is a sentence polarization function, and nb is the number of buckets for the sentiment values. We use a tempered softmax function to obtain a distribution of sentiments. In Figure 2, we set $nb = 5$ and observe that the review has a neutral to positive side. The neutral sentiment is due to factual sentences that the reviewer uses.

Instantiating quality. In GUIDES, we consider four different instances of the quality function: *numerical diversity*, *textual diversity*, *coverage*, and *none*. These functions are employed in typical EDA systems [2, 15, 48, 55] for measuring quality. Table 1 provides details for these instances. Note that the *none* variant of the quality function is helpful when the relevance function suffices on its own, and no further maximization is necessary.

Example. Assume the example in Section 1 was performed in the setting of manual EDA. After initializing the exploration by focusing on cameras with high optical and processing power, Steve focuses on a review about "autofocus issues" for the Fujifilm X100V camera. In the first iteration, Steve picks δ = *text* and *quality* = *textdiv* to receive a diverse set of reviews whose text is similar to the current review. In the second iteration, Steve focuses on a review about "Canon 6D" and picks δ = *topics* and *quality* = *coverage* to receive a set of items covering the topics discussed in his selected review. The goal of our work is to automate this tedious manual process by learning TIE policies and enabling *partially-guided EDA*. We discuss the learning process in the next section.

4.2 The GUIDES architecture

Figure 3 provides the overall architecture of GUIDES. which consists of an offline TIE training phase and an online interactive TIE phase. The offline training phase simulates a DRL agent to learn a TIE policy. The agent interacts with a prediction neural network (step 1) to update its weights by performing actions on the environment (step 2). GUIDES is equipped with a multi-valued reward mechanism that filters actions with dominated rewards (step 3). These rewards are then used (step 4) to optimize the TIE policy (step 5). The optimal policy will then be leveraged in an online TIE context to recommend actions to the user. The user consumes these recommendations for an effective decision-making in her future exploration iterations.

4.3 Learning TIE policies in GUIDES

The input to GUIDES is $(S, E, P, R) \setminus P$. Given that exploration logs are unavailable in TIE and that transition probabilities between exploration states are unknown, model-free RL fits our context.

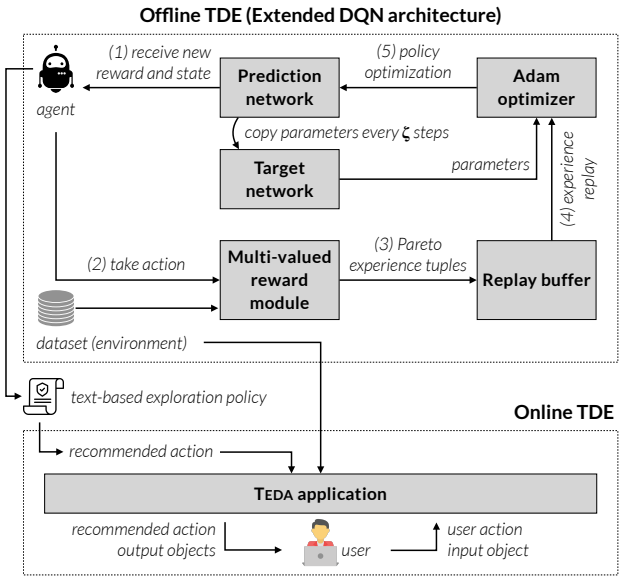

Figure 3: The GUIDES architecture.

We leverage a Deep RL (DRL) algorithm called Deep Q-Networks (DQN) [42] as the learning component of GUIDES. Among a variety of discrete-space DRL algorithms in the literature , we choose DQN because its architecture complies with compound targets. Recall that being "compound" means that the target is spread throughout the exploration session. Algorithm 1 illustrates the main steps of learning a TIE policy using DQN. The algorithm starts by initializing a buffer $\mathcal{B}$ and two neural networks, the prediction network θ and the target network θ^T, and then proceeds by updating them. The algorithm outputs an optimized exploration policy which is represented in terms of θ's learned weights. In this section, we first explain these variables (Section 4.3.1), and then present the learning procedure (Section 4.3.2).

4.3.1 Learning variables. GUIDES simulates interactions with the environment using TIE actions, and updates the DQN learning variables with the rewards received from the environment. In DQN, the updates do not impact the neural network weights right after each interaction with the environment. For more stability in the learning, each interaction is first pushed to a buffer $\mathcal{B}$, and then the learning is performed by sampling from $\mathcal{B}$. Each interaction stored in $\mathcal{B}$ is a tuple $\omega_t = \langle s_t, e_t, r_t, s_{t+1} \rangle$, called an *experience tuple* at iteration t (Line 10 in Algorithm 1), and the sampling process is called an *experience replay* (Line 12).

Prediction and target networks. We train the prediction network θ (using experience replays from $\mathcal{B}$) as a function approximator for our action-value function, denoted as $Q(s, e; \theta)$ (Lines 2 and 3 in Algorithm 1). To achieve learning stability, the parameters of the prediction network are copied to the target network θ^T, every ζ steps, where ζ is a hyper-parameter (Line 15). The internal architecture of both prediction and target networks are often identical. In GUIDES, the architecture that we consider for both networks consists of four layers of linear transformation with width 1024 and rectifier non-linearity functions (ReLU) in-between, and $\zeta = 100$.

Algorithm 1: Learning TDE polices in GUIDES

1 $\mathcal{B} \leftarrow \emptyset$
2 $\theta \leftarrow random(); \theta^T \leftarrow \theta$
3 $Q(s,e;\theta), Q(s,e;\theta^T) \leftarrow initialize(), \forall s \in S, e \in E$
4 **while** *not end_of_learning* **do**
5 $\quad s_t \leftarrow \langle x_0, X_0 \rangle$
6 $\quad e_t \leftarrow get\_action(\varepsilon)$
7 $\quad$ **while** *not end_of_session* **do**
8 $\quad\quad X_{t+1}, r_t \leftarrow e_t(s_t, k)$
9 $\quad\quad s_{t+1} \leftarrow \langle x_{t+1}, X_{t+1} \rangle$
10 $\quad\quad \omega_t \leftarrow \langle s_t, e_t, r_t, s_{t+1} \rangle$
11 $\quad\quad \mathcal{B} \leftarrow multi\_objective\_push(\omega_t, \mathcal{B}, \mu)$ // *Algorithm 2*
12 $\quad\quad \bar{\omega} \leftarrow sample(\mathcal{B})$
13 $\quad\quad \theta \leftarrow gradient\_descent(\bar{\omega})$
14 $\quad\quad$ **if** ζ *steps passed* **then** $\theta^T \leftarrow \theta$
15 $\quad\quad e_{t+1} \leftarrow get\_action(\varepsilon)$
16 $\quad\quad s_t \leftarrow s_{t+1}, e_t = e_{t+1}$
17 $\quad$ **end**
18 **end**

We also employ Adam [29] as the optimizer with a learning rate $\alpha = 0.0003$.

4.3.2 Learning procedure. Each step in the learning process begins by initializing a starting state $s_t = \langle x_0, X_0 \rangle$ (Line 5), picking an exploration action $e_t = get\_action(\varepsilon)$ using the ε-greedy method (Line 6), and then following an ensuing exploration session until its termination. In each step of the session, first the chosen action is applied to obtain the reward r_t and the next state s_{t+1} (Line 8), and then the experience tuple is formed to be pushed to the buffer (Line 11). Following that, an experience tuple $\bar{\omega}$ is replayed from $\mathcal{B}$ (Line 12) to update the parameters of the Q function approximator θ (Line 13). We now explain how states are represented and how rewards are processed in GUIDES.

Exploration state representation. To increase the effectiveness of the learning process, we represent the states and actions jointly in a fine-granularity space by leveraging a set of data-dependent state-action linear features $ft(s,e)$. These features enable the discovery of commonalities in the state-action space and hence widen the scope of the learned policy. Table 2 provides details of the features. We leverage approximate control methods [61] to learn an approximation of the action-value function $\hat{Q}(\mathbf{w}, s, e) \approx Q(s,e)$ where $\mathbf{w} \in \mathbb{R}^m$ and m is the number of features, $m \ll |S \times E|$. We compute the approximated action-value function as $\hat{Q}(\mathbf{w}, s, e) = \mathbf{w}^\top ft(s,e)$. The weights $\mathbf{w}$ are learned by the prediction network θ.

Reward representation. As instructed in Equation 7, the reward system in GUIDES is multi-valued due to the complex nature of textual data. This means reward maximization becomes a multi-objective optimization problem, where each reward dimension acts as one optimization objective. We propose to modify the DQN architecture to account for that. This process is depicted in Line 11 of Algorithm 1 and detailed in Algorithm 2. Experience tuples are pushed into the buffer if and only if they are *not dominated* by any other tuple, which means they belong to the Pareto front. Let $\succ$ denote the dominance function, we define the dominance relation between two experience tuples ω_i and ω_j as follows [65]:

$$\omega_i \succ \omega_j \Leftrightarrow \exists \delta; r_i^\delta > r_j^\delta \wedge \nexists \delta' \in \mathcal{K}; r_i^{\delta'} < r_j^{\delta'}$$

where r_i^δ is the δ-th component of the reward r_i for the experience tuple ω_i. Given a set of experience tuples B, we verify whether ω_i fits in the set of Pareto front B as follows:

$$Pareto(\omega_i, B) \Leftrightarrow \nexists \omega_j \in B; \omega_j \succ \omega_i \tag{11}$$

Following Equation 11, an experience tuple ω_i is pushed to $\mathcal{B}$ iff $Pareto(\omega_i, \mathcal{B})$ holds. This ensures that the buffer is always the container of non-dominated experiences.

Algorithm 2: Multi-objective optimization of reward

Input: experience tuple ω, buffer $\mathcal{B}$, approximation ratio μ
Output: updated buffer $\mathcal{B}^*$
1 $\mathcal{B}^* \leftarrow \mathcal{B}$
2 **if** $\mathcal{B}^*$ *is not full* **then**
3 $\quad$ **if** $Pareto(\omega, \mathcal{B}^*)$ **then**
4 $\quad\quad \mathcal{B}^*.append(\omega)$
5 $\quad\quad$ **if** $\omega \succ_\mu \omega_j$ **then** $\mathcal{B}^*.remove(\omega_i)\ \forall \omega_i \in \mathcal{B}^*$
6 $\quad$ **end**
7 **end**
8 **else**
9 $\quad \mathcal{B}^* \leftarrow \emptyset$
10 $\quad \mathcal{B}^*.append(\omega)$
11 **end**
12 **return** $\mathcal{B}^*$

5 EXPERIMENTS

Our experiments aim to validate the effectiveness of GUIDES by answering the following questions: *"are the semantics proposed for TIE actions expressive enough to navigate in textual data and reach targets?"*, *"are learned policies useful for TIE?"*, *"does multi-reward optimization improve learning?"*, and *"does our approach scale with increasing task complexity and data size?"*.

5.1 Summary of results

Our first finding is that exploration performs best when both text-based and attribute-based semantics are integrated (Section 5.3). We compare GUIDES with traditional EDA approaches and observe that text-agnosticism results in poor performance. We also observe that the DRL agent picks the textual dimensions *summary* and *sentiments* more frequently than others. This is likely due to the fact that the former provides adequate details about reviews, and the latter reflects opinion. We show that multi-objective reward optimization detects and invalidates up to 30% of all iterations (Section 5.4). We observe that fewer iterations are discarded in later stages of the training, which shows that the DRL agent learns to land on iterations with non-dominated rewards. We also observe that agents that leverage Pareto rewards discover 50% more targets on average than the ones with non-Pareto rewards. Last, we shed light on scalability aspects and show that GUIDES maintains its

Feature group	Components
Input object x_t	object type (item or review, one feature), rating (one feature for each 5-star rating scale), text word count (one-hot), tag count (one-hot), and sentiments and topics (10 features for each)
Output objects X_t	Same features as the ones for x_t, for each output object $x \in X_t$
Exploration action e_t	$textdiv(X_t)$, $numdiv(X_t)$, $coverage(X_t)$, all one-hot encoded
Target $\mathcal{T}$	number of targets discovered in $\mathcal{T}$ in one-hot encoding

Table 2: Text-based state-action features in GUIDES

efficiency on larger datasets and more complex tasks with a slower pace for finding targets (Section 5.5).

5.2 Experimental setup

Datasets. We test GUIDES on two real and publicly available datasets, Amazon [9, 44] and IMDb [3]. The former includes information about the products in the Amazon online shopping platform and their customer reviews. We focus on the *Electronics* category with 786, 868 products and $21M$ reviews. The latter contains 453, 528 movies and $5.5M$ reviews provided in the IMDb portal. Following our data model (Section 3.1), each object x in both datasets is associated with its $profile(x)$. Items in Amazon are described using the attributes *sub-category*, *title*, *rating*, and *price*. Item attributes in IMDb are *director*, *title*, *rating*, *release date*, *genres*, and *actors*. Reviews in both datasets are described using the attributes *review time* and *helpfulness score*. In all our experiments, we use a 50-core sample for each dataset and employ 5000 objects to cap the training time to 12 hours. We also build a larger variant of the IMDb dataset (denoted as Large) with 20000 objects for our scalability experiment (Section 5.5).

Building signatures. For each object x, we build its signature $\eta(x)$ over the schema $\mathcal{K} = (text, summary, tags, topics, sentiments)$. In both datasets, $\eta(x).text$ and $\eta(x).summary$ are the vectorized versions of the provided text and summary using GloVe [47]. The set of $\eta(x).tags$ is generated by extracting the POS patterns from the text. On average, we obtained 9.29 and 21.55 tags per object in Amazon and IMDb, respectively. We generate $\eta(x).topics$ using the gensim implementation of LDA for topic modeling [52], with the number of topic $nt = 10$. Last, we generate sentiment distributions $\eta(x).sentiments$ as the aggregation of polarity scores over the sentences of the item's description and the review's text, with the number of buckets $nb = 10$.

Relevance and quality computations. To simulate exploration sessions for policy learning, it is crucial that the execution of exploration actions is efficient. In an offline process, we build inverted indices for each object that list their pairwise similarity scores in decreasing order. Whenever an exploration action is triggered, we use a greedy hill-climbing optimization algorithm to maximize the quality function (diversity or coverage, depending on the semantics of the selected quality function) while respecting a threshold σ on relevance using the inverted indices [45]. In the greedy quality improvement loops, the inverted indices enable a sequential access, where the next candidate to improve quality is the next object in the inverted index under investigation. Also in case the quality function is *none*, the top-k objects in the inverted index are retrieved instantaneously.

Exploration tasks. To conform with the nature of EDA, we consider our exploration tasks to be subjective. This means that *the DRL agent does not know the target*. For each dataset, we consider two diverse exploration tasks, ET1 and ET2. In Amazon, the task is to explore 5 products and their reviews whose rating is the highest in a specific sub-category. We consider *Computers* as the sub-category for ET1 and *Camera & Photo* for ET2. In IMDb, the task is to explore 5 movies (50 movies for the Large dataset) and their reviews whose rating is the highest in a specific genre. We consider *Comedy* as the genre for ET1 and *Crime* for ET2. Moreover, we consider two exploration tasks on IMDb Large dataset with more complex semantics: ET3 that seeks 1000 reviews with the most positive sentiments, and ET4 that seeks 1000 reviews that are highly focused on a single topic. We leverage ET1 and ET2 in our study of the RL training and application and move to ET3 and ET4 to examine how GUIDES handles more complex scenarios.

Learning parameters. For learning TIE policies, we simulate 850 sessions with 200 iterations each, with $k = 5$ for each exploration action in the space $\mathcal{X} = items \cup reviews$. In each iteration, the DRL agent has 6 options for the relevance function δ (i.e, $|\mathcal{K} + 1|$ possible dimensions) and 4 options for the quality function (see Table 1), which means 24 different semantics can be selected as the exploration action. We set the learning rate α to 0.0003, the reward discount factor γ to 0.9, and ϵ to 0.5. The reward computation follows Equation 7 with the two following rules incorporated in Ω:

(1) Given an item i as the current selection $x_t = i$, for each review $r \in X_{t+1}$, $item(r) = i$;
(2) An object $x \in \mathcal{T}$ is only rewarding at the first visit.

GUIDES is implemented in Python 3.8.5 using a PyTorch-based DRL library called PFRL [18], and Gym library [6] for the instanciation of the TIE environment.

5.3 TIE actions

An important aspect of TIE actions is their ability to handle both structured and unstructured data. We employ four different variants: TRAD, TEXT, ABST, and ALL. TRAD is an adaptation of traditional EDA approaches that rely only on item attributes [15, 55]. In TEXT, the exploration actions use text-based representations, i.e., *summary* and *text*, as the only δ variants. In ABST, the abstract representations are used, i.e., all except the choices in TEXT. ALL leverages all text-based and attribute-based representations. We train our DRL agent and measure the utility of each policy as the number of objects discovered in $\mathcal{T}$ averaged over 5 runs. Figure 4 shows the results[1]. We observe that ALL is the winner in both tasks. This policy reaches around 40 targets in IMDb and 50 to 60 targets in Amazon, after 850 simulated sessions. In Amazon, ET1 reaches fewer targets than ET2, which shows that exploring *Computers* sub-category is a harder task compared to *Camera & Photo*, due to the increased heterogeneity in the computer-related products and their reviews. We also notice the low performance of TRAD, which acknowledges the importance of text-based representations in the exploration.

[1] *As we observed similar results for both datasets, we only report on Amazon. Upon acceptance, we will report the exhaustive set of results in a companion technical report.*

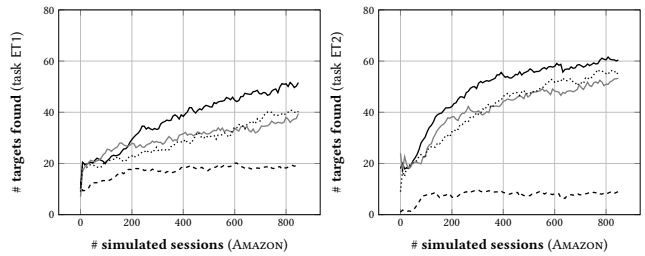

Figure 4: Exploration action variants for task ET1 (left) and ET2 (right) on Amazon.

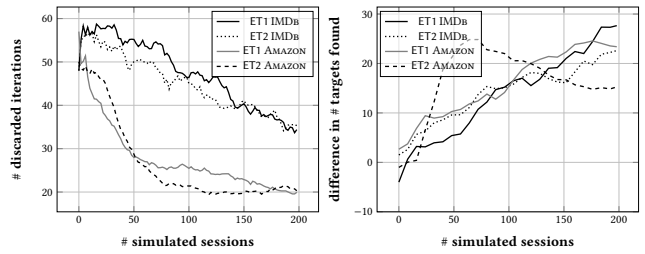

Figure 5: Multi-objective optimization of rewards.

5.4 Multi-valued rewards

Given the complex nature of textual data, GUIDES uses multi-valued rewards to represent different dimensions of text. To verify the effectiveness of the multi-objective optimization of rewards, we consider two different implementations: SCL and MOO. SCL is the implementation of the scalarization method [24] where all the reward signals are combined together in a linear function with equal weights to form one single scalar reward. MOO is the method integrated in GUIDES to find Pareto rewards and discard dominated ones (see Algorithm 2). Given a state s_t, MOO does not let the agent learn from s_t if r_t is dominated in the buffer $\mathcal{B}$. We compare SCL and MOO using two different measures: number of discarded iterations by MOO (Figure 5 left) and the difference in the number of discovered targets (Figure 5 right).

Number of discarded iterations. Figure 5 left shows the benefit of MOO over SCL in terms of number of discarded iterations. For instance, MOO discards 50 iterations in the task ET2 IMDb at 50 simulated sessions. We observe that overall MOO invalidates up to 30% of the iterations and it drops to less than 25% for IMDb and 15% for Amazon after 100 sessions. All discarded iterations are used by SCL which decreases the optimality of the policies. We only show the first 200 sessions as the agent reached stability around this point. The decrease in the number of discarded iterations shows that the Pareto rewards benefit the agent in learning TIE policies, because the agent lands more on iterations with non-dominated rewards in later sessions. Moreover, the drastic decrease in Amazon shows that the reward signals are more distinct compared to IMDb's, hence the Pareto rewards can be obtained earlier in the learning process.

Difference in the number of discovered targets. Figure 5 right shows the difference in the number of discovered targets between MOO and SCL. For instance, MOO reaches 10 more targets in the task ET1 Amazon at 50 sessions. We observe that overall MOO-based exploration policies succeed to find on average 50% more targets compared to SCL, after only 200 sessions. In Amazon, MOO-based policies have higher impact on harder tasks (see Figure 4), as it finds 54% more targets for ET1 (the harder task) and 25% for ET2 (the easier task). SCL has a supremacy in very early stages of the learning process (task ET1 IMDb and task ET2 Amazon) which is soon taken over by its competitor. The reason is that MOO discards iterations drastically at the beginning and hence limits the opportunities of landing on targets while exploring the spaces.

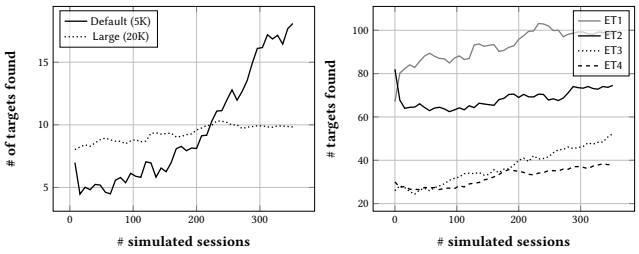

Figure 6: Scalability power of GUIDES by increasing the data size (left) and the complexity of tasks (right).

5.5 Scalability

Finding targets in TIE becomes more challenging by increasing either the data size or the complexity of the exploration task. We examine the scalability power of GUIDES by training policies on IMDb Large dataset and two complex tasks ET3 and ET4.

Increasing data size. Figure 6 left compares the number of discovered targets for the task ET1 IMDb when exploring the default-size and Large datasets. After 350 sessions, the former reaches 18 targets and the latter reaches 10. We observe an overall increasing growth in both datasets with a slower pace for the Large dataset due to the complexity of landing on targets in a larger exploration space.

Increasing task complexity. Figure 6 right compares the number of discovered targets between normal tasks ET1 and ET2, and high-complexity tasks ET3 and ET4, in IMDb. All tasks show relatively similar growth rates, indicating that the system manages to learn complex tasks as well as normal ones. However, we observe a constant gap all along the training between the normal and complex tasks. This illustrates the difference of complexity between the tasks, as the targets in ET1 and ET2 are mostly reviews sharing the same items, making them more grouped hence easier to find than the more diverse and dispersed targets of ET3 and ET4. This shows that the TIE exploration actions are capable of handling more complex tasks to reach targets efficiently.

6 CONCLUSION AND FUTURE WORK

We develop GUIDES, a framework for guided Text-best Item Exploration. We model the TIE problem as finding a policy with the highest utility for a target set of items and propose a non-trivial adaptation of the DQN architecture to accommodate multiple rewards. Our work open several directions, including example-based TIE operators, and the incorporation of user feedback in the reward mechanism.

REFERENCES

[1] Julien Aligon, Enrico Gallinucci, Matteo Golfarelli, Patrick Marcel, and Stefano Rizzi. 2015. A collaborative filtering approach for recommending OLAP sessions. *Decision Support Systems* 69 (2015), 20–30.

[2] Sihem Amer-Yahia, Tova Milo, and Brit Youngmann. 2021. Exploring Ratings in Subjective Databases. In *SIGMOD*.

[3] Enam Biswas. 2021. IMDb Review Dataset. https://doi.org/10.34740/KAGGLE/DSV/1836923

[4] Johannes Bjerva, Nikita Bhutani, Behzad Golshan, Wang-Chiew Tan, and Isabelle Augenstein. 2020. SubjQA: A Dataset for Subjectivity and Review Comprehension. *arXiv preprint arXiv:2004.14283* (2020).

[5] Cristiana Bolchini, Elisa Quintarelli, and Letizia Tanca. 2012. Context Support for Designing Analytical Queries. In *Methodologies and Technologies for Networked Enterprises - ArtDeco: Adaptive Infrastructures for Decentralised Organisations*. Springer, 277–289.

[6] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540

[7] Gloria Chatzopoulou, Magdalini Eirinaki, and Neoklis Polyzotis. 2009. Query recommendations for interactive database exploration. In *International Conference on Scientific and Statistical Database Management (SSDBM)*. Springer, 3–18.

[8] Konstantina Christakopoulou, Alex Beutel, Rui Li, Sagar Jain, and Ed H Chi. 2018. Q&r: A two-stage approach toward interactive recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 139–148.

[9] AWS Public Datasets. 2015. Amazon Customer Reviews. https://s3.amazonaws.com/amazon-reviews-pds/readme.html.

[10] Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. 2016. AIDE: An Active Learning-Based Approach for Interactive Data Exploration. *IEEE Trans. Knowl. Data Eng.* 28, 11 (2016), 2842–2856.

[11] Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander H. Miller, Arthur Szlam, and Jason Weston. 2016. Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[12] Marina Drosou and Evaggelia Pitoura. 2013. YmalDB: exploring relational databases via result-driven recommendations. *International Journal on Very Large Data Bases (VLDBJ)* 22, 6 (2013), 849–874.

[13] Roee Ebenstein, Niranjan Kamat, and Arnab Nandi. 2016. FluxQuery: An execution framework for highly interactive query workloads. In *International Conference on Management of Data (SIGMOD)*. 1333–1345.

[14] Magdalini Eirinaki, Suju Abraham, Neoklis Polyzotis, and Naushin Shaikh. 2014. Querie: Collaborative database exploration. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 26, 7 (2014), 1778–1790.

[15] Ori Bar El, Tova Milo, and Amit Somech. 2020. Automatically Generating Data Exploration Sessions Using Deep Reinforcement Learning. In *International Conference on Management of Data (SIGMOD)*.

[16] Sara Evensen, Aaron Feng, Alon Halevy, Jinfeng Li, Vivian Li, Yuliang Li, Huining Liu, George Mihaila, John Morales, Natalie Nuno, et al. 2019. Voyageur: An experiential travel search engine. In *The World Wide Web Conference*. 3511–5.

[17] Jerome H Friedman and John W Tukey. 1974. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on computers* 100, 9 (1974), 881–890.

[18] Yasuhiro Fujita, Prabhat Nagarajan, Toshiki Kataoka, and Takahiro Ishikawa. 2021. ChainerRL: A Deep Reinforcement Learning Library. *Journal of Machine Learning Research* 22, 77 (2021), 1–14. http://jmlr.org/papers/v22/20-376.html

[19] Liqiang Geng and Howard J Hamilton. 2006. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)* 38, 3 (2006), 9.

[20] David Gotz and Zhen Wen. 2009. Behavior-driven visualization recommendation. In *Proceedings of the 14th international conference on Intelligent user interfaces*. 315–324.

[21] Julia Hirschberg and Christopher D Manning. 2015. Advances in natural language processing. *Science* 349, 6245 (2015), 261–266.

[22] Enhui Huang, Liping Peng, Luciano Di Palma, Ahmed Abdelkafi, Anna Liu, and Yanlei Diao. 2018. *Optimization for active learning-based interactive database exploration*. Ph.D. Dissertation. Ecole Polytechnique; University of Massachusetts Amherst.

[23] Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. 2018. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2107–2116.

[24] Johannes Jahn. 1985. Scalarization in multi objective optimization. In *Mathematics of Multi Objective Optimization*. Springer, 45–88.

[25] Manas Joglekar, Hector Garcia-Molina, and Aditya Parameswaran. 2017. Interactive data exploration with smart drill-down. *IEEE Transactions on Knowledge and Data Engineering* 31, 1 (2017), 46–60.

[26] Magdalena Kaiser, Rishiraj Saha Roy, and Gerhard Weikum. 2020. Conversational Question Answering over Passages by Leveraging Word Proximity Networks. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 2129–2132.

[27] Tom Kenter and Maarten de Rijke. 2017. Attentive memory networks: Efficient machine reading for conversational search. *arXiv preprint arXiv:1712.07229* (2017).

[28] Nodira Khoussainova, YongChul Kwon, Magdalena Balazinska, and Dan Suciu. 2010. SnipSuggest: Context-aware autocompletion for SQL. *Proceedings of the VLDB Endowment* 4, 1 (2010), 22–33.

[29] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[30] Martin Kirchgessner, Vincent Leroy, Sihem Amer-Yahia, and Shashwat Mishra. 2016. Testing interestingness measures in practice: A large-scale analysis of buying patterns. In *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on*. IEEE, 547–556.

[31] Jonathan Koren, Yi Zhang, and Xue Liu. 2008. Personalized interactive faceted search. In *Proceedings of the 17th International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008*, Jinpeng Huai, Robin Chen, Hsiao-Wuen Hon, Yunhao Liu, Wei-Ying Ma, Andrew Tomkins, and Xiaodong Zhang (Eds.). ACM, 477–486.

[32] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *International conference on machine learning*. 1378–1387.

[33] Doris Jung-Lin Lee, Himel Dev, Huizi Hu, Hazem Elmeleegy, and Aditya Parameswaran. 2019. Avoiding drill-down fallacies with vispilot: Assisted exploration of data subsets. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. 186–196.

[34] Chenliang Li, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. 2016. Topic modeling for short texts with auxiliary word embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 165–174.

[35] Jiwei Li, Alexander H. Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. 2017. Dialogue Learning With Human-in-the-Loop. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[36] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards Deep Conversational Recommendations. In *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). 9725–9735.

[37] Yuliang Li, Aaron Feng, Jinfeng Li, Saran Mumick, Alon Halevy, Vivian Li, and Wang-Chiew Tan. 2019. Subjective Databases. *Proc. VLDB Endow.* 12, 11 (July 2019), 1330–1343. https://doi.org/10.14778/3342263.3342271

[38] Bang Liu, Haojie Wei, Di Niu, Haolan Chen, and Yancheng He. 2020. Asking Questions the Human Way: Scalable Question-Answer Generation from Text Corpus. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 2032–2043. https://doi.org/10.1145/3366423.3380270

[39] Patrick Marcel and Elsa Negre. 2011. A survey of query recommendation techniques for data warehouse exploration.. In *Journées Francophones sur les Entrepôts de Données et l'Analyse en ligne (EDA)*. 119–134.

[40] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*. 165–172.

[41] Zhengjie Miao, Yuliang Li, Xiaolan Wang, and Wang-Chiew Tan. 2020. Snippext: Semi-supervised opinion mining with augmented data. In *Proceedings of The Web Conference 2020*. 617–628.

[42] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[43] Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, and Themis Palpanas. 2017. New trends on exploratory methods for data analytics. *Proceedings of the VLDB Endowment* 10, 12 (2017), 1977–1980.

[44] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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)*. 188–197.

[45] Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, and Alexandre Termier. 2015. Interactive user group analysis. In *International Conference on Information and Knowledge Management (CIKM)*. 403–412.

[46] Yongjoo Park, Ahmad Shahab Tajik, Michael Cafarella, and Barzan Mozafari. 2017. Database learning: Toward a database that becomes smarter every time. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 587–602.

[47] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on*

empirical methods in natural language processing (EMNLP). 1532–1543.

[48] Aurélien Personnaz, Sihem Amer-Yahia, Laure Berti-Equille, Maximilian Fabricius, and Srividya Subramanian. 2021. Balancing Familiarity and Curiosity in Data Exploration with Deep Reinforcement Learning. In *AiDM*.

[49] Aurélien Personnaz, Sihem Amer-Yahia, Laure Berti-Equille, Maximilian Fabricius, and Srividya Subramanian. 2021. DORA The Explorer: Exploring Data with Interactive Deep Reinforcement Learning. In *CIKM*.

[50] Sajjadur Rahman, Mangesh Bendre, Yuyang Liu, Shichu Zhu, Zhaoyuan Su, Karrie Karahalios, and Aditya G. Parameswaran. 2021. NOAH: Interactive Spreadsheet Exploration with Dynamic Hierarchical Overviews. *Proc. VLDB Endow.* 14, 6 (2021), 970–983.

[51] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Trans. Assoc. Comput. Linguistics* 7 (2019), 249–266. https://transacl.org/ojs/index.php/tacl/article/view/1572

[52] Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic* 3, 2 (2011).

[53] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[54] Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. 1998. Discovery-driven exploration of OLAP data cubes. In *International Conference on Extending Database Technology (EDBT)*. 168–182.

[55] Mariia Seleznova, Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, and Eric Simon. 2020. Guided exploration of user groups. *Proceedings of the VLDB Endowment (PVLDB)* 13, 9 (2020), 1469–1482.

[56] Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Philipp Eichmann, Navid Karimeddiny, Charlie Meyer, Wesley Runnels, and Tim Kraska. 2021. Davos: A System for Interactive Data-Driven Decision Making. *Proc. VLDB Endow.* 14, 12 (2021), 2893–2905. http://www.vldb.org/pvldb/vol14/p2893-shang.pdf

[57] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. 2016. Effortless Data Exploration with zenvisage: An Expressive and Interactive Visual Analytics System. *Proceedings of the VLDB Endowment* 10, 4 (2016).

[58] Manish Singh, Michael J Cafarella, and HV Jagadish. 2016. DBExplorer: Exploratory Search in Databases.. In *International Conference on Extending Database Technology (EDBT)*. 89–100.

[59] Damiano Spina, Johanne R Trippas, Lawrence Cavedon, and Mark Sanderson. 2017. Extracting audio summaries to support effective spoken document search. *Journal of the Association for Information Science and Technology* 68, 9 (2017), 2101–2115.

[60] Yueming Sun and Yi Zhang. 2018. Conversational Recommender System. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz (Eds.). ACM, 235–244.

[61] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[62] Paul Thomas, Daniel McDuff, Mary Czerwinski, and Nick Craswell. 2017. MISC: A data set of information-seeking conversations. In *SIGIR 1st International Workshop on Conversational Approaches to Information Retrieval (CAIR'17)*, Vol. 5.

[63] Tian Tian and Zheng Felix Fang. 2019. Attention-based autoencoder topic model for short texts. *Procedia Computer Science* 151 (2019), 1134–1139.

[64] Johanne R Trippas, Damiano Spina, Lawrence Cavedon, and Mark Sanderson. 2017. Crowdsourcing user preferences and ery judgments for speech-only search. In *1st SIGIR Workshop on Conversational Approaches to Information Retrieval (CAIR'17)*.

[65] Immanuel Trummer and Christoph Koch. 2014. Approximation schemes for many-objective query optimization. In *SIGMOD*.

[66] Damir Vandic, Steven S. Aanen, Flavius Frasincar, and Uzay Kaymak. 2017. Dynamic Facet Ordering for Faceted Product Search Engines. *IEEE Trans. Knowl. Data Eng.* 29, 5 (2017), 1004–1016.

[67] Manasi Vartak, Silu Huang, Tarique Siddiqui, Samuel Madden, and Aditya Parameswaran. 2017. Towards visualization recommendation systems. *ACM SIGMOD Record* 45, 4 (2017), 34–39.

[68] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. 2015. SeeDB: efficient data-driven visualization recommendations to support visual analytics. *Proceedings of the VLDB Endowment* 8, 13 (2015), 2182–2193.

[69] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya G. Parameswaran, and Neoklis Polyzotis. 2015. SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics. *VLDB* (2015).

[70] Xiaolan Wang, Yoshihiko Suhara, Natalie Nuno, Yuliang Li, Jinfeng Li, Nofar Carmeli, Stefanos Angelidis, Eser Kandogann, and Wang-Chiew Tan. 2020. ExtremeReader: An interactive explorer for customizable and explainable review summarization. In *Companion Proceedings of the Web Conference 2020*. 176–180.

[71] Lanbo Zhang and Yi Zhang. 2010. Interactive retrieval based on faceted feedback. In *International Conference on Research and Development in Information Retrieval (SIGIR)*. 363–370.

[72] Xiong Zhang, Jonathan Engel, Sara Evensen, Yuliang Li, Çağatay Demiralp, and Wang-Chiew Tan. 2020. Teddy: A System for Interactive Review Analysis. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[73] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 177–186.

[74] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 425–434.

[75] Xiaolin Zheng, Weifeng Ding, Zhen Lin, and Chaochao Chen. 2016. Topic tensor factorization for recommender system. *Information Sciences* 372 (2016), 276–293.

[76] Mengyu Zhou, Wang Tao, Ji Pengxin, Han Shi, and Zhang Dongmei. 2020. Table2Analysis: Modeling and Recommendation of Common Analysis Patterns for Multi-Dimensional Data. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[77] Yuan Zuo, Junjie Wu, Hui Zhang, Hao Lin, Fei Wang, Ke Xu, and Hui Xiong. 2016. Topic modeling of short texts: A pseudo-document view. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2105–2114.