FDepHunter: Harnessing Negative Examples to Expose Fakes and Reveal Ghosts

Pavel Koupil
Charles University
Prague, Czech Republic
pavel.koupil@matfyz.cuni.cz

Jáchym Bártík
Charles University
Prague, Czech Republic
jachym.bartik@matfyz.cuni.cz

Stefan Klessinger
University of Passau
Passau, Germany
stefan.klessinger@uni-passau.de

André Conrad
University of Hagen
Hagen, Germany
andre.conrad@fernuni-hagen.de

Stefanie Scherzinger
University of Passau
Passau, Germany
stefanie.scherzinger@uni-passau.de

ABSTRACT

Functional dependency (FD) discovery is fundamental in data profiling. Inevitably, existing approaches can return fake FDs that hold only coincidentally. Moreover, these approaches fall short of identifying ghost FDs that would be observable in a clean dataset, but that remain undetected because of outliers in the data. We introduce an interactive method for dependency discovery that augments an Armstrong relation with additional tuples. We rely on artificially generated *negative examples* that emulate real-world tuples to help expose fake FDs. In addition, we rely on domain experts to confirm that *positive examples* indeed reflect the characteristics of the original dataset. Our tool prototype *FDepHunter* thus provides a novel human-in-the-loop workflow where the set of discovered FDs can be iteratively refined.

PVLDB Reference Format:
Pavel Koupil, Jáchym Bártík, Stefan Klessinger, André Conrad, and Stefanie Scherzinger. FDepHunter: Harnessing Negative Examples to Expose Fakes and Reveal Ghosts. PVLDB, 18(12): 5227-5230, 2025.
doi:10.14778/3750601.3750612

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/mmcatdb/fdephunter.

1 INTRODUCTION

Functional dependencies (FDs) are a foundational concept in data management, with applications in schema matching, dataset similarity, outlier detection, and extraction of both integrity constraints and business rules. Several approaches to facilitate FD discovery have been proposed [8, 10, 11], which differ primarily in performance and suitability for specific datasets.

Which FDs are discovered depends on the quality of the input data. In particular, FDs can be classified into three types [3]: (1) *genuine* FDs, which remain valid across all realistic dataset instances and capture inherent dependencies; (2) *fake* FDs, which hold only within the given dataset but fail when additional valid tuples are introduced; fake FDs tend to occur in datasets with large numerical domains, such as medical or transport data, where most values are effectively unique – for example, the US flights dataset[1] contains over 900,000 FDs [9], many of which are probably fake; and (3) *ghost* FDs, which remain undetected due to outliers or anomalies but which would hold in a clean dataset.

A fully representative dataset would help in identifying genuine FDs. To better approximate dataset completeness, we introduce two complementary concepts: *negative* and *positive examples*. The former are artificially constructed to simulate realistic but absent data, allowing the removal of fake FDs. This approach is motivated by Gold's Theorem [6], which highlights the need for negative evidence, such as unseen valid tuples, to infer generalizable properties from incomplete data. The latter reflect tuples of the original dataset and are used to reveal ghost FDs obscured by outliers. *FDepHunter*, the tool presented in this paper, enables users to interactively confirm or reject negative and positive examples.

Most FD discovery approaches return a minimal FD set, i.e., a basis from which other (non-minimal) FDs follow via Armstrong's axioms. When the dataset is extended (e.g., with negative examples), the minimal set may change, and newly added FDs must be validated to ensure their genuineness. Beyond incompleteness, this introduces the challenge of validating such FDs. We propose a human-in-the-loop workflow to discover genuine FDs and lay the groundwork for follow-up research, e.g., on handling this at scale.

Contributions of this paper include the following:

- We present *FDepHunter*, a data profiling tool designed to support domain experts. The tool provides a novel interactive workflow to detect fake FDs and to facilitate the discovery of ghost FDs.
- *FDepHunter* incrementally constructs an Armstrong relation [1], augmented with so-called *negative* and *positive examples*, to approximate the properties of a complete dataset.
- Users iteratively refine the set of FDs: they identify genuine FDs, expose fake FDs through *negative examples*, and uncover ghost FDs via *positive examples*.
- Negative and positive examples are generated to target multiple fake or ghost FDs simultaneously. This allows to efficiently refine multiple dependencies in a single step.

[1] https://www.transtats.bts.gov, accessed July 2025

Proceedings of the VLDB Endowment, Vol. 18, No. 12 ISSN 2150-8097.
doi:10.14778/3750601.3750612

2 CORE CONCEPTS

In this section, we introduce and illustrate the core concepts.

For a relational schema R and sets of attributes $X, Y \subseteq R$, a FD $X \rightarrow Y$ holds if, whenever two tuples agree on X they also agree on Y; c.f. Abiteboul et al. [1] for further details. We call X the left-hand side (LHS) and Y the right-hand side (RHS) of the FD. A FD $X \rightarrow Y$ is *minimal* if there is no strict superset $X' \subseteq R$ of X for which $X' \rightarrow Y$ holds.

We classify FDs by their RHS, grouping FDs determining the same attribute into distinct *classes*. The set of these classes is denoted by C. Within each class $c \in C$, we identify *maximal sets* [8] of attributes on the LHS that do not form a FD for the corresponding RHS. Formally, a set $X \subseteq R$ is a maximal set for the class of A if $X \rightarrow A$ does not hold, but $X' \rightarrow A$ holds for every strict superset $X' \subseteq R$ of X. The set of all maximal sets for a class c is denoted by $\mathcal{M}_c$. The elements of $\mathcal{M}_c$ and the FDs can be illustrated using a *lattice*, where each node represents a subset of attributes and edges indicate the addition or removal of an attribute.

An *Armstrong relation* satisfies exactly a given set of FDs and no others [1]. We artificially construct an Armstrong relation, beginning with a base tuple t_0, where all attributes are assigned a default value. For each maximal set $M \in \bigcup_{c \in C} \mathcal{M}_c$, a unique tuple t_M is added to the relation. In each tuple t_M, attributes belonging to the corresponding maximal set M retain the default value, whereas attributes not in M are assigned distinct real-world values. By substituting abstract identifiers with actual data values, the Armstrong relation mirrors the characteristics of input datasets.

Within the Armstrong relation, rows are categorized as either *positive* or *negative examples*. Both are constructed by modifying a base tuple t_0 using real-world values so that the values of LHS attributes match t_0 while the values of RHS attributes differ. Positive examples collectively uphold exactly the FDs in the input dataset. In contrast, negative examples are artificially generated so that they violate FDs that may hold coincidentally and are likely fake.

Example 2.1. Fig. 1 shows a sample from the real-world IMDB title dataset[2]. Fig. 2 shows an Armstrong relation created from the sample. The positive examples (marked with red squares) form an Armstrong relation that satisfies twelve minimal FDs (e.g., $CD \rightarrow A$, $CE \rightarrow A$, $C \rightarrow B$). By adding the negative example (yellow), the Armstrong relation violates the FD $C \rightarrow B$. Finally, Fig. 3 presents a lattice diagram for the class of A (denoted by c_A), where elements of $\mathcal{M}_{c_A}$ are shown in solid red, their subsets as outlined red nodes, minimal FDs in solid blue, and derived FDs as outlined blue nodes. This visualization provides an intuitive map of the combinations of attributes that determine the RHS within each FD class.

3 WORKFLOW

We now present the workflow of *FDepHunter*, illustrated in Fig. 4.

Initial Data Profiling. As part of the initial phase focusing on the extraction of FDs and maximal sets $\mathcal{M}_c$ for each class $c \in C$, we apply a method that fits the properties of the input data. This work employs an approach inspired by Dep-Miner [8]. For each class $c \in C$, the method initially generates the maximal sets $\mathcal{M}_c$ and subsequently reconstructs the FDs. Alternatively, approaches such as HyFD [11] can be employed to extract FDs first, ensuring that even

[2]https://developer.imdb.com/non-commercial-datasets/, accessed July 2025

tconst	primaryTitle	startYear	runtimeMin.	genres
tt0036443	Titanic	1943	85	Action+Drama+History
tt0079836	S.O.S. Titanic	1979	194	Drama+History
tt0115392	Titanic	1996	87	Action+Drama+History
tt0120338	Titanic	1997	194	Drama+Romance
tt0155274	Titanic	1915	null	History
tt0594950	Titanic	1997	null	Documentary+Short
tt0650185	Titanic Tech	2003	46	Documentary+History
tt0771984	Titanic	2006	51	Documentary
tt0902058	The Titanic	1981	null	Documentary+Drama+Fantasy
tt0929378	Titanic's Ghosts	2002	57	Documentary+History+War

Figure 1: Input dataset.

	A	B	C	D	E	
	tconst	primaryTitle	startYear	runtimeMin.	genres	
	tt0036443	Titanic	1943	85	Action+Drama+History	
	tt0079836	Titanic	1943	194	Drama+History	BC
	tt0115392	Titanic	1979	85	Drama+Romance	BD
	tt0120338	Titanic	1996	87	Action+Drama+History	BE
	tt0155274	S.O.S. Titanic	1997	85	History	D
	tt0594950	Titanic Tech	1943	null	Documentary+Short	C

Figure 2: Positive (red) and negative (yellow) examples.

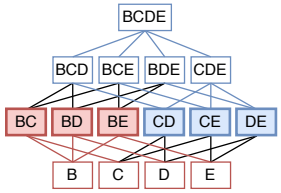

Figure 3: An example of lattice for class c_A (i.e., RHS: A).

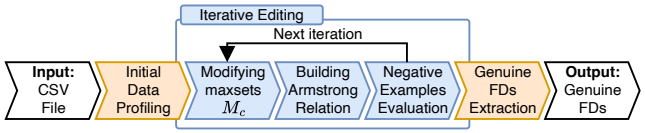

Figure 4: A high-level overview of the FDepHunter workflow.

for large data volumes the system is reactive and prompt, allowing for fluent interaction. Maximal sets are then reconstructed by pruning attributes from minimal LHSs until only maximal subsets that do not form an FD remain.

Iterative Editing of Maximal Sets. In the second phase, the goal is to identify *fake* FDs and to uncover *ghost* FDs, by employing heuristics [3] or by incrementally consulting users; we focus on the incremental approach. Fake FDs are examined by iterating from smaller to larger LHSs (as illustrated in Fig. 5), assuming pessimistically that the extracted dependencies (shown in blue) are fake. Consequently, an affected $\mathcal{M}_c$ is expanded to include a new element M corresponding to the targeted FD (depicted in yellow), and all $M' \subset M$, $M' \in \mathcal{M}_c$, are removed to preserve maximality. This adjustment updates the Armstrong relation to contain one additional row for each newly added element $M \in \bigcup_{c \in C} \mathcal{M}_c$, forming a

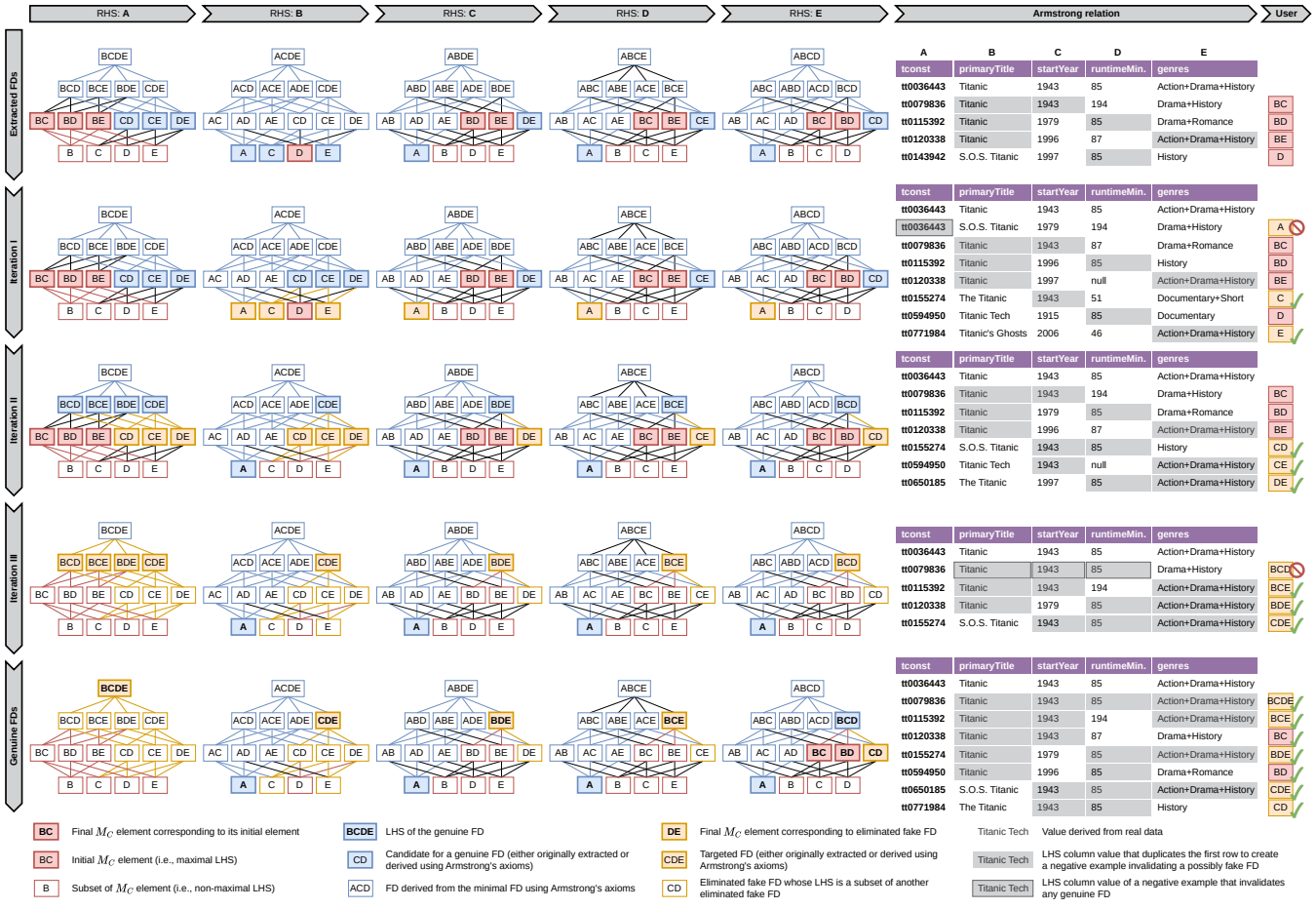

Figure 5: Iterative (top-down) editing of maximal sets and elimination of fake FDs utilizing negative examples.

negative example evaluated by the user. If accepted, the targeted FD is deemed fake; if rejected, it implies that at least one corresponding FD could be genuine, so the matching LHS is removed from affected $\mathcal{M}_c$. If confirmed that all FDs with that LHS are genuine, pruning of LHS stops; otherwise, the algorithm identifies genuine FDs, and continues the search for supersets of LHS that fail to invalidate genuine FDs. If any genuine FD is identified, the M corresponding to its LHS is removed from all affected $\mathcal{M}_c$, and previously removed M' are restored. After all negative examples are processed, the algorithm proceeds to the next iteration, increasing the size of the LHS by one, and terminates when every extracted FD has been classified. With this strategy users do not need to validate the full transitive closure of FDs derived via Armstrong's axioms, but only those with LHSs that are a strict superset of a previously evaluated one.

For ghost dependencies, the process focuses on the original (red) elements M in each $\mathcal{M}_c$, proceeding in reverse (from those containing the largest number of columns to smaller ones). Here too, the user receives pairs of rows – specifically, the first row and one with a red square – to determine whether the latter truly represents a maximal set. If the user accepts the pair, the currently evaluated element M remains valid in each relevant $\mathcal{M}_c$; if rejected, the algorithm presumes an undiscovered FD exists, marks the targeted element as an FD, and continues to target all its immediate predecessors (i.e., subsets of attributes of size one less), verifying them in subsequent iterations. The algorithm terminates once no M (or its subsets) remains to be checked.

Both steps ultimately yield a collection of genuine FDs by incrementally eliminating fake FDs and uncovering ghost FDs. While the worst-case number of examples a user must review is 2^n (lattice size), in our experience this remains manageable in practice, especially for small datasets with many genuine FDs.

Example 3.1. Fig. 5 illustrates the iterative workflow for identifying genuine FDs from the dataset in Fig. 1. The initial phase outputs maximal sets and FDs depicted as a lattice alongside the corresponding Armstrong relation. Each iteration eliminates FDs, from the smallest to the largest LHS. Negative examples targeting $LHS = A$ and $LHS = BCD$ are rejected as A is an identifier and $BCD \rightarrow E$ is genuine (e.g., two series episodes may share genre E but have distinct identifiers A). Finally, the user accepts the remaining maximal sets, ensuring no ghost FDs are in the dataset.

Architecture and Implementation. FDepHunter is implemented as a client-server application. On the server side, it relies on *Java 21* and the *Spring Boot* framework to provide a REST interface. The client employs the *React* library to provide a GUI that displays the current status of the entire workflow.

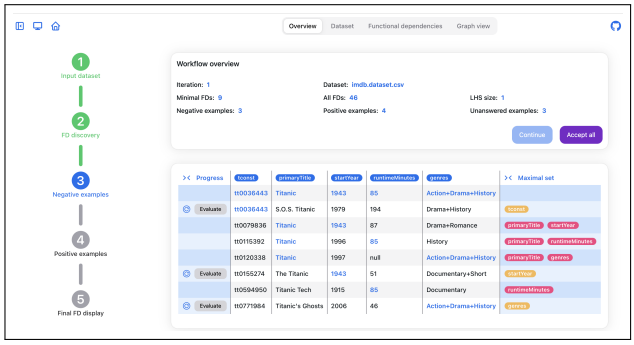

Figure 6: Armstrong relation and workflow status.

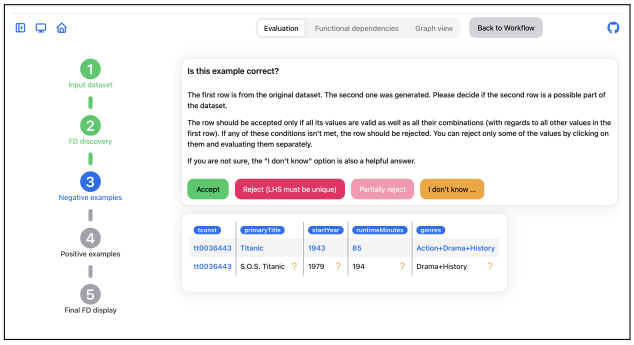

Figure 7: Evaluation of negative example.

4 RELATED WORK

Several studies have evaluated the quality of discovered FDs using scoring functions. Chu et al. [5] introduce *succinctness* and *coverage* to assess the interestingness of denial constraints, a generalization of FDs. Chiang and Miller [4] propose similar ranking functions for conditional FDs. Andritsos et al. [2] rank FDs by their information content, while Wei and Link [13] use data redundancy as a heuristic to identify genuine FDs. Berti-Équille et al. [3] study the impact of missing values on the discovery of genuine, ghost, and fake FDs under various NULL semantics and propose probabilistic scoring.

A separate line of research uses Armstrong relations to identify meaningful FDs [12]. This is highly related to our notion of positive examples. Langeveldt and Link [7] evaluate the usefulness of Armstrong relations for identifying meaningful FDs. They conclude that Armstrong relations are helpful in identifying meaningful FDs that are incorrectly perceived as meaningless, but not vice versa.

Our approach differs from existing work by iteratively refining Armstrong relations through user feedback, and also incorporating artificial negative examples to help eliminate fake FDs.

5 DEMONSTRATION OUTLINE

In our interactive tool demo, users systematically identify genuine, fake, and ghost FDs through the following steps: (1) Users begin by selecting from a curated collection of real-world datasets, which differ in size, attribute count, and exhibit varying numbers of fake FDs and outliers (e.g., IMDb datasets, or Metanome profiling datasets [9]). Alternatively, users can provide their own dataset. (2) Users iteratively refine maximal sets in ascending size by reviewing and validating examples. Each iteration generates an Armstrong relation in which negative examples target fake FDs (see Fig. 6). Negative examples appear sequentially (see Fig. 7), and are either accepted, removing a fake FD, or rejected with justification, as users act as domain experts. (3) In the final stage, users assess remaining positive examples and reject outliers to reveal ghost FDs.

6 CONCLUSION AND OUTLOOK

FDepHunter introduces an interactive workflow to eliminate fake FDs using real-world-like negative examples and to uncover ghost FDs, hidden by outliers, by validating positive examples.

Future work will focus on enhancing the scalability and effectiveness of *FDepHunter*. Specifically, we plan to develop advanced strategies for constructing negative examples with maximal impact by leveraging heuristics to assess the genuineness of FDs. Additionally, we aim to conduct user studies with domain experts to systematically evaluate the tool's usability and its effectiveness in practical applications. A key objective will be to assess the cognitive load perceived by domain experts when evaluating negative and positive examples and investigate whether showing additional context or metadata is perceived as helpful.

ACKNOWLEDGMENTS

This work was supported by GAČR grant no. 23-07781S (P. Koupil), SVV project no. 260 821 (J. Bártík), DFG grant no. #385808805 (S. Klessinger, A. Conrad, and S. Scherzinger), as well as BTHA grant BTHA-AP-2023-21 (S. Scherzinger and P. Koupil).

REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. 1995. *Foundations of Databases.* Addison-Wesley.
[2] Periklis Andritsos, Renée J. Miller, and Panayiotis Tsaparas. 2004. Information-Theoretic Tools for Mining Database Structure from Large Data Sets. In *Proc. SIGMOD.* 731–742.
[3] Laure Berti-Équille, Hazar Harmouch, Felix Naumann, Noël Novelli, and Saravanan Thirumuruganathan. 2018. Discovery of Genuine Functional Dependencies from Relational Data with Missing Values. *In Proc. VLDB* 11, 8 (2018), 880–892.
[4] Fei Chiang and Renée J. Miller. 2008. Discovering data quality rules. *In Proc. VLDB* 1, 1 (2008), 1166–1177.
[5] Xu Chu, Ihab F. Ilyas, and Paolo Papotti. 2013. Discovering Denial Constraints. *In Proc. VLDB* 6, 13 (2013), 1498–1509.
[6] E. Mark Gold. 1965. Limiting Recursion. *J. Symb. Log.* 30, 1 (1965), 28–48.
[7] Warren-Dean Langeveldt and Sebastian Link. 2010. Empirical evidence for the usefulness of Armstrong relations in the acquisition of meaningful functional dependencies. *Inf. Syst.* 35, 3 (2010), 352–374.
[8] Stéphane Lopes, Jean-Marc Petit, and Lotfi Lakhal. 2000. Efficient Discovery of Functional Dependencies and Armstrong Relations. In *Proc. EDBT (LNCS)*, Vol. 1777. 350–364.
[9] Felix Naumann. 2025. Repeatability. https://hpi.de/naumann/projects/repeatability/data-profiling/fds.html. Online resource. Accessed July 2025.
[10] Thorsten Papenbrock, Jens Ehrlich, Jannik Marten, Tommy Neubert, Jan-Peer Rudolph, Martin Schönberg, Jakob Zwiener, and Felix Naumann. 2015. Functional Dependency Discovery: An Experimental Evaluation of Seven Algorithms. *In Proc. VLDB* 8, 10 (2015), 1082–1093.
[11] Thorsten Papenbrock and Felix Naumann. 2016. A Hybrid Approach to Functional Dependency Discovery. In *Proc. SIGMOD.* 821–833.
[12] Antonio M. Silva and Michel A. Melkanoff. 1979. A Method for Helping Discover the Dependencies of a Relation. In *Advances in Data Base Theory*, Vol. 1. 115–133.
[13] Ziheng Wei and Sebastian Link. 2023. Towards the efficient discovery of meaningful functional dependencies. *Inf. Syst.* 116, Article 102224 (2023).