Explaining Dataset Changes for Semantic Data Versioning with Explain-Da-V

Himanshu Garg, Sahil Malhotra, Fateen Ahmed

CS520-Data Integration, Warehousing and Provenance

Dr. Boris Glavic

Illinois Institute of Technology

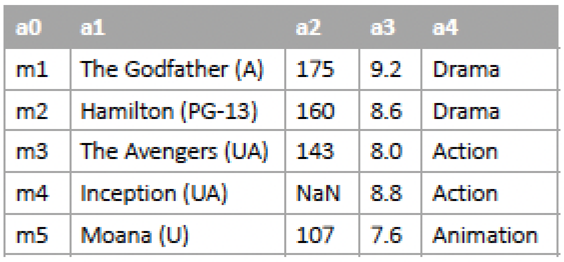
30th November 2023

List of Contents

Introduction

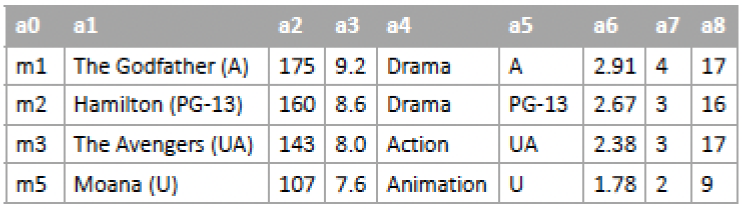
With the ever-increasing demand for AI/ML, the demand for big-data management systems is also growing rapidly. While the currently existing datasets can track dataset changes to some level, a granular approach to track the changes individually by logging their exact transformations is still missing from most platforms.

For projects with multiple collaborating users, documentation is a key aspect of the project, however, oftentimes, the documentation on an enterprise level is either poorly maintained or not generated at all -this leads to eventual miscommunication and confusion. Moreover, the difference in the datasets and the platforms also increases the chances of discrepancies amongst various platforms while noting the metadata and data changes – leading to further reduction of code reproducibility and productivity.



a0	a1	a2	a3	a4
m1	The Godfather (A)	175	9.2	Drama
m2	Hamilton (PG-13)	160	8.6	Drama
m3	The Avengers (UA)	143	8.0	Action
m4	Inception (UA)	NaN	8.8	Action
m5	Moana (U)	107	7.6	Animation

(a) Dataset version created by UserA



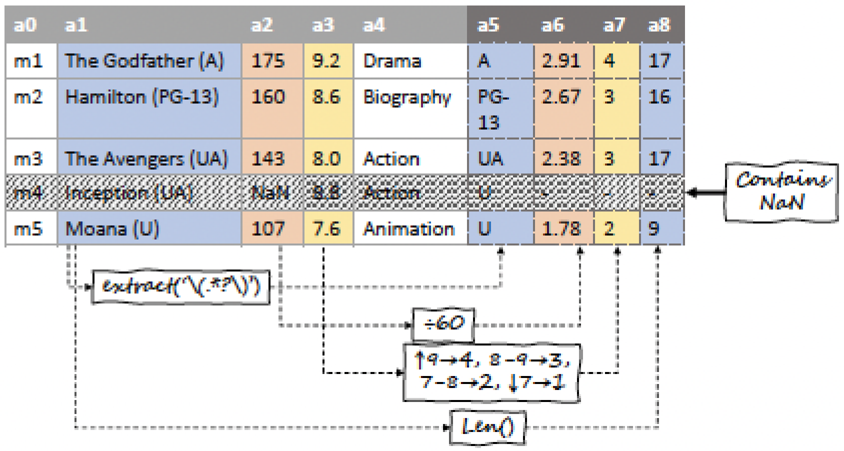
a0	a1	a2	a3	a4	a5	a6	a7	a8
m1	The Godfather (A)	175	9.2	Drama	A	2.91	4	17
m2	Hamilton (PG-13)	160	8.6	Drama	PG-13	2.67	3	16
m3	The Avengers (UA)	143	8.0	Action	UA	2.38	3	17
m5	Moana (U)	107	7.6	Animation	U	1.78	2	9

(b) Dataset version created by UserB

Figure 1: Example dataset versions about movies created by two users. Attribute names are provided in Example 1.

Explanation of Example:

In the aforementioned example, it can be clearly seen that {a5,a6,a7,a8} columns are being added to the second version and {m4} row is being deleted in the second version. While the motivation behind these changes are not evident, we follow a series of steps/functions to explain the exact structural changes in the dataset at each iteration, in a user-understandable manner.



a0	a1	a2	a3	a4	a5	a6	a7	a8
m1	The Godfather (A)	175	9.2	Drama	A	2.91	4	17
m2	Hamilton (PG-13)	160	8.6	Biography	PG-13	2.67	3	16
m3	The Avengers (UA)	143	8.0	Action	UA	2.38	3	17
~~m4~~	~~Inception (UA)~~	~~NaN~~	~~8.8~~	~~Action~~	~~U~~	-	-	-
m5	Moana (U)	107	7.6	Animation	U	1.78	2	9

Figure 2: An interpretation of the changes between the dataset versions given in Figure 1. The columns are colored based on their origin (e.g., a5 is blue because it originates from the blue a1) and annotated column transformations are given at the bottom. The annotated row transformation is given on the right, in this case removing a row, which is also illustrated by diagonal stripes over the row.

Explanation

Through this example, we can see that multiple transformations can be applied on the original dataset. These transformations can be:

Serial Number	Type of transformation
1	Altering integer values (through multiplication or division)
2	Editing the content or editing the number of row and columns
3	Modifying the datatype (for eg. INT -> STRING)
4	Changing columns/rows
5	Change cells/full transformations

Abstract

Akin to software versioning, in this paper, we explore a new concept – dataset versioning. While most companies are moving towards big data, few are concerned about versioning their dataset, which is becoming a necessity with the ever-increasing demand for data transformation and exploration. To counter this issue, the paper explores a tool Explain-Da-V, which aims to simplify the differences between different versions of the same dataset, by studying the functional data transformations reiteratively. The paper also briefs upon the quality assessment parameters, which explain the quality of data transformation clarity. Through this paper, we understand that Explain-Da-V can provide pristine explanations when compared to other methods.

2. Related Work

- Existing tools in the market usually focus more on the initial and final values, rather than noting down the transformation itself for easier backtracking later.

- This paper places more focus on creating an understanding for these "transformations" rather than the initial and final output, to ensure accurate version control.

- This paper works on the intuition that an output table will be created from the source table with a known match of operations between the two. External alterations (eg. Joining or partitioning tables are considered a future scope)

2.1 DATA VERSIONING

With the tracking of all these changes, we encounter another problem – storage of countless versions of the same dataset. While other methods focus more on versioning the table by following a git-akin approach (using tools like DataHub, TardisDB etc), the proposed method focuses more on versioning the data itself. By tracking aspects like content changes, contextual changes, interpretational changes and quality changes, the method tries to keep the storage overhead low by emphasizing on the data versioning itself.

2.2 Data change, difference and integration

Through extensive research in the field of data integration (by studying schema matching and entity resolution), earlier methods follow an educated assumption that two versions of the same dataset will encounter a match between them, which can be leveraged to understand the changes between the datasets. However, our paper places more emphasis on local changes to understand the following:

- Exact version difference (what changed)?
- Timestamp of changes (when did it change)?
- Transformations performed (How did it change)?

2.3 Data transformation by example

Through the concept of "query reverse engineering", we aim to study how the data can be automatically transformed to achieve the desired output.

This can be achieved by two methods:

1. **Programming-by-example**: This method follows a "brute-force" kind of an approach, by generating search spaces of potential operators to be applied on the input table, and explore multiple search algorithms to find the accurate algorithm that can bring us the desired output using the given input. This can be performed through dropping columns, splitting columns by a delimiter or using RegEx expressions.

   Eg. Foofah, CLX, Datadiff

2. **Transformation Repositories**: Instead of focusing on pre-determined set of operations, this method craws sites like GitHub, Stack Overflow, Wikipedia

and other knowledge bases, to generate an intrinsic repo of ranked transformations. This repo is leveraged to identify the appropriate transformations.

Eg. DataX-Former, Proteus

Contrast to the other methods, our focus is to create valid, generalizable and explainable multi-dimensional transformations. These transformations can go beyond simple editing of attributes, we can also address different data types. Going beyond normal string-based transformations, we can also incorporate machine-learning algorithms to identify any forms of transformations. We can also support text-to-numeric and data cleansing transformations. Essentially, other researchers focus on matching input and output, this paper focusses on finding valid and explainable transformations.

3. Semantic Data Versioning

A table dataset (T) consists of a set of attributes (Ta) and tuples (Tr). Each tuple has a tuple identifier, values assigned to attributes and the attribute index. If we are presented with two datasets with derivation between them without any apparent code or documentation, we aim to use this method to understand the transformations. Given two datasets with alignment of their attributes, the tuples with similar identifier represent same real-world entity. The unmatched attributes are defined as those without any alignment in the second table.

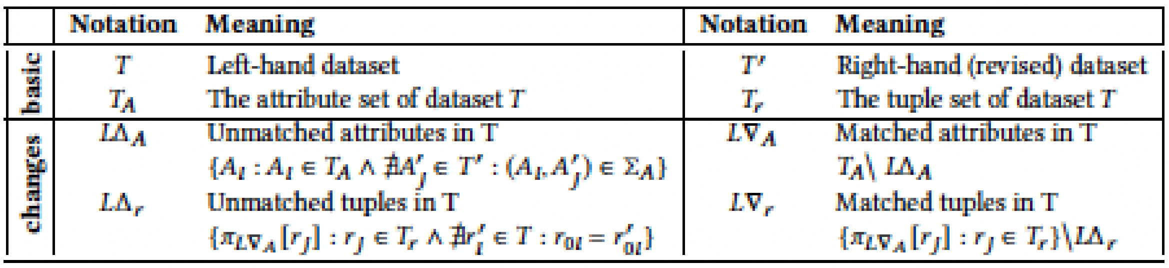
3.1 Explaining dataset changes

The paper discusses two forms of changes: vertical and horizontal. They can be represented as adding something and taking away something. In vertical changes, we focus on the change in the characteristics of the change (eg. INT->String) and in horizontal changes, we focus on change within the items (eg. Adding/deleting new data from the cells). The table includes the following symbols: $L\Delta A$, $L\nabla A$, $L\Delta r$, and $L\nabla r$ – which can be defined as:

<u>Example 4</u>

We consider Figure 1a and 1b. Our goal can be defined as: G = ($a6$, $\pi a6$ [T']) i.e. E$a6$ = ($a2$, $a2 \div 60$)

- This can be represented as: the goal a6 in T’ starts with a2 in T and we then divide it by 60. In this example, our focus is on change sets to recognize different explanations.
- <u>Vertical Addition Explanations ($L\Delta A$)</u>: This attempts to explain why certain characteristics were added.
- <u>Vertical Removal Explanations ($R\Delta A$)</u>: This attempts to explain why certain characteristics were deleted.
- <u>Horizontal Addition Explanations ($L\Delta r$)</u>: This attempts to explain why certain rows were added.
- <u>Horizontal Removal Explanations ($R\Delta r$)</u>: This attempts to explain why certain rows were deleted.



	Notation	Meaning	Notation	Meaning
basic	T	Left-hand dataset	T'	Right-hand (revised) dataset
	T_A	The attribute set of dataset T	T_r	The tuple set of dataset T
changes	$L\Delta_A$	Unmatched attributes in T $\{A_i : A_i \in T_A \wedge \nexists A'_j \in T' : (A_i, A'_j) \in \Sigma_A\}$	$L\nabla_A$	Matched attributes in T $T_A \setminus L\Delta_A$
	$L\Delta_r$	Unmatched tuples in T $\{\pi_{L\nabla_A}[r_j] : r_j \in T_r \wedge \nexists r'_i \in T : r_{0l} = r'_{0l}\}$	$L\nabla_r$	Matched tuples in T $\{\pi_{L\nabla_A}[r_j] : r_j \in T_r\} \setminus L\Delta_r$

<u>Example 5</u>

Explain-Da-V consists of 4 components (highlighted above) to encounter with addition/deletion of attributes and tuples. This forms the data-driven strategy behind this approach. Once we have noted these essential components, we can then move on to the horizontal and vertical modifications.

4. CORE SEMANTIC EXPLANATION METHODS

Following represents the working of the core semantic layer:

1. **Consider the data types**: Initially, we align the data types from our origin (O) to our goal (G).
2. **Traditional database attribute shift**: Instead of focusing on traditional database attribute shift like string, int, float etc, we dive into newer attributes like Numeric, Categorical, and Textual (attained from Machine Learning feature types). We can also consider mixed types that become textual.
3. **Handle core changes**: We address 3 main changes:

   3.1. Numeric

   3.2. Categorical

   3.3. Textual

This helps us to deal with a wide range of transformations.

4. **Version adaptability**: Explain-Da-V is noted as a versatile platform, that can work on numerous pair of versions – regardless of however many transformations have been applied on them.

5. **Unexplained changes**: For any unexplained changes, we label them as "idiopathic" to place special emphasis on them eventually.
6. **Vertical additions**: These are the most commonly found changes and serves as the target base for our primary methods.
7. **Assumption**: Our goal is assumed to be a single attribute with data values.

4.1 NUMERICAL CHANGE EXPLANATIONS

1. **Approach**: The problem is assumed to be a regression task for numerical goal with numerical data.

2. **Regression Framework**: Independent variables are formed from the origin relation tuples. The dependent variables are formed from the goal relation tuples.

3. **Linear regression**: We use linear regression to understand explainable transformations.

4. **Controlling complexity**: Lasso and Ridge techniques are used to maintain model's simplicity and overcome overfitting.

Example 6. A numeric transformation is given in Figure 1b, where explaining a6 can be resolved by fitting a regressor 1/60 .a2

Explanation:

Here, we are working on enhancing the numerical transformations:

1. Limitations of linear functions: The linear functions are unable to cover the numerical transformations appropriately.

2. Expansion of feature space: The feature space (origin) gets expanded to generate additional features for enabling versatile transformations.

3. Polynomial Regression: Polynomial features are added to the origin to enable transformations that are polynomial in nature.

4. Inter-relation Features: Multiplying and dividing different attribute values in the origin help to enhance the transformation capabilities. These can be applied directly on the tuple level.

5. Extensions consecutively: Sequential extensions are applied to enable complex attributes that are difficult to solve by hand (eg. Applying the BMI formula).

6. Mathematical transformations: The mathematical extension of the origin are used to cover common maths formula such as log, squareRoot, reciprocal and exponent.

7. Transformations of the tuple: The tuple-specific maths transformations are applied.

8. Global aggregators: Attributes like sum, mean, max, and min are generated to cover ML transformations like normalization. This value is compounded by the values in the attribute.

9. Regression fitting: Extended feature set fits on a regressor, to assign coefficients on the extended feature set.

10. Learning Transformation: The transformation gets fitted on T and T' without training data.

<u>4.2 CATEGORICAL CHANG EXPLANATIONS</u>

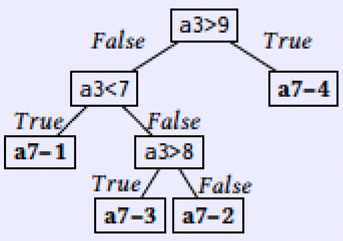
1. **<u>Problem framing</u>**: The problem is defined as a classification task to define categorical goal using an origin relation of numeric data.

2. **<u>Classification Framework</u>**: Explanatory variables are listed as the origin relation's tuples. The goal serves as the output class labels.

3. **<u>Binary/Multi-class output</u>**: Output can be binary (movie is longer than 2 hours or not) or multi-class (grade of a student).

4. **<u>Explainability focus</u>**: We use decision trees to understand the problem statement. This is because decision trees can represent the explanations in disjunctions or conjunctions. The path from root to lead is a conjunction and the tree is a disjunction. They provide a combination of conditions that are interpretable and help understand the categorical goal of the numeric origin.

EXAMPLE 7. *Figure 1b provides an example a categorical transformation, namely,* a7, *which can be resolved with the help of the following decision tree.*



<u>4.3 TEXTUAL CHANGE EXPLANATIONS</u>

1. **<u>The PBE approach for the textual origin</u>:**

When we have to deal with text based data, we employ the PBE (Programming by Example) approach. Eg, Foofah, an existing framework.

2. **<u>Search – based Solution</u>:**

The search algorithm generated by the PBE explores operator space using a **heuristic function** to uncover cost of proposed solution. Here we can also note that pruning the space enhances search speed.

3. **<u>Textual-to-text transformations</u>:**

Conventional NLP steps like **lemmatization**, **removal of special characters** etc can be handled by extending traditional PBE steps. The repository keeps a note of the implemented operators.

4. **Text-to-numeric transformations:**

We use a **meta-operator** to count occurrences of a pattern pat in the value. The operations will be defined and we also count the pre-determined sets of stop-words.

5. **Text-to-categorical transformations:**

A similar meta-operation for pattern existence can be defined and contains_percent = contains‘%′ can be formed using this logic.

4.4 CATEGORICAL ENCODING CHANGE EXPLANATIONS

1. **Handling mixed data types:**

We use one-hot-encoding to deal with mixed data types, specifically textual and categorical based.

2. **Values assignment:**

We assign 1 to specific categorical value and 0 to others to make everything uniform.

3. **Ordinal encoding:**

We use this encoding technique to enhance our transformations and provide more intricate representations.

Example:

We can use this encoding technique to classify prediction of movie ration (eg. "Is_Drama" or "Is_Action") to enhance the learning process.

4.5 Reshaping Change Explanations

Contrary to using “group-by” to simply reshape a table where a direct match between attributes of T and T’ are non existent, we can also use this method to create aggregated attributes based on another field. Numerical change explanations can also be handled this way. During this process, each numerical attribute (Aj) in table Ta has an extended origin in regards to textual or categorical attribute Ai in origin.

We deploy a helper query to create extended group-by features for numerical attribute Aj that uses a categorical attribute Ai. The query given in Figure 3 helps to create additional possible attributes. This is done by joining it with the origin.

Furthermore, we can also reshape our tables using the Figure-3 query and a regressor. Currently, Explain-Da-V solely focuses on reshaping of the attribute-match. It cannot provide support for transpose, pivot or any similar transformations.

```
SELECT A_i, mean(A_j), max(A_j)...
FROM T
GROUP BY A_i
```

Figure 3: Group by Query

4.6 FINDING THE ORIGIN

In order to enhance our search's efficiency, we also consider the possibility of the origin being a subset of T, rather than the previous assumption of origin being the relation T itself. Thus our task becomes to find the origin since we are unaware of the origin that was used to derive a specific goal. Initially, one might consider using all the attributes of T as the origin (O = (TA,T)). However, using all fields of T as the origin can welcome noise into the model through the unrelated attributes, since this presents a much larger search space that can cause unwanted noise. Example, if we try to resolve A5 using A1-A4 then this can create issues of unwanted noise.

If we attempt to create a new attribute using an already existing attribute, we inadvertently create a new functional dependency between the origin and the new attribute. In order to discover this dependency, we need to deploy a new functional dependency algorithm, to recognize attributes where our goal becomes the new dependent set, and the determinant also becomes the origin. Here we also get a case where many attribute sets can be used to recognize the goal, which although produces many origins.

To take care of this situation, we need to study the attribute sets. Section 7.4 helps explain how we eventually choose from these sets using rank determinants of size and cardinality. We can also deploy an early-stop condition via the size or quality of discovered transformations.

5. Explaining vertical changes

Adding or removing an attribute is an integral part of data science, since it forms the basis for exploratory data analysis and feature engineering across all domains in machine learning. Most often, the added attributes are a by-product of applied transformations on existing datasets, and the paper works on explaining these changes in an iterative approach.

We follow the following steps for this:

1. The first step is to identify the origin of our data, which is usually the source through which the very first attribute gets derived. However, finding the origin is a huge task since with many operations, the origin gets buried deep within the level of functional dependencies.

2. Specific to the types of origin and the goal attributes, we deploy different algorithms to recognize the origin.

3. Explain-Da-V works by following a recursive approach to encounter one new attribute at a time. For each newly added attribute, the origin is identified and tailored explanation methods are employed for identifying the origin and the final goal.

4. As is the nature of data iterations, we can have numerous origins for one single newly added attribute with 'n' number of explanations for each goal. This becomes even more prominent when we are dealing with diverse and complex transformations.

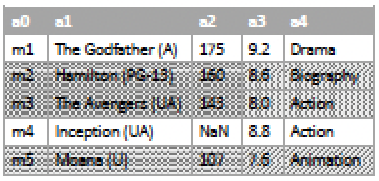
5. Explain-Da-V uses a search strategy to explore these options and select the useful explanations finally.

5.1 Explanation of removals

The concept of removals deals with the removals or deletion of tuples or attributes from a dataset, either in a singular fashion or in a bulk format. It has following types:

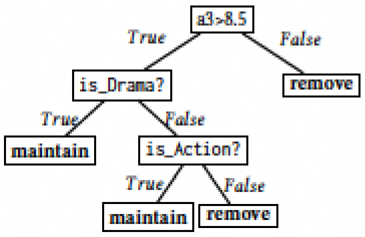
- Superficial removal: Due to some attributes being deemed irrelevant or redundant, these features are removed directly without further analysis into their significance.
- In-depth removal: In cases where the insignificance is not apparent, further analysis is performed on the dataset using graphs and statistical inferences to recognize features and tuples that present no specific use to the overall goal. We can then remove this data after recognizing their insignificance.

- Idiopathic removal: When an attribute contains duplicated information from other columns or rows, it is considered as 'idiopathic' circumstance, and this data is removed from the table.



a0	a1	a2	a3	a4
m1	The Godfather (A)	175	9.2	Drama
m2	Hamilton (PG-13)	160	8.6	Biography
m3	The Avengers (UA)	143	8.0	Action
m4	Inception (UA)	NaN	8.8	Action
m5	Moana (U)	107	7.6	Animation

(a) Dataset version created by UserD over Figure 1a.

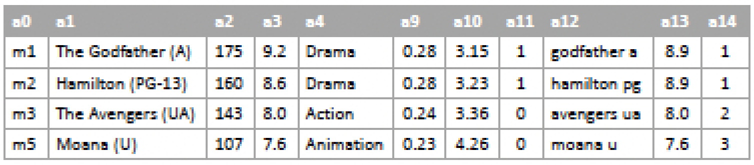


(b) Tree

Figure 5: UserD version of Figure 1a and its explanation.

6. Explaining horizontal changes:

Horizontal changes bring light to the concept of adding or deleting data from individual rows or a bulk of rows together. The paper highlights methods to explain the removal of individual tuples, through predicate logic to ensure systematic explanation for the removals. The paper also briefly discusses addition of tuples due to over sampling and non-idiopathic reasons.



a0	a1	a2	a3	a4	a9	a10	a11	a12	a13	a14
m1	The Godfather (A)	175	9.2	Drama	0.28	3.15	1	godfather a	8.9	1
m2	Hamilton (PG-13)	160	8.6	Drama	0.28	3.23	1	hamilton pg	8.9	1
m3	The Avengers (UA)	143	8.0	Action	0.24	3.36	0	avengers ua	8.0	2
m5	Moana (U)	107	7.6	Animation	0.23	4.26	0	moana u	7.6	3

Figure 4: Dataset version created by UserC over Figure 1a.

6.1 Explanation of Removals

The concept of removals deals with the removals or deletion of tuples or attributes from a dataset, either in a singular fashion or in a bulk format. It has following types:

- Superficial removal: Due to some attributes being deemed irrelevant or redundant, these features are removed directly without further analysis into their significance.
- In-depth removal: In cases where the insignificance is not apparent, further analysis is performed on the dataset using graphs and statistical inferences to recognize features and tuples that present no specific use to the overall goal. We can then remove this data after recognizing their insignificance.
- Idiopathic removal: When an attribute contains duplicated information from other columns or rows, it is considered as 'idiopathic' circumstance, and this data is removed from the table.

Explaining horizontal changes:

Horizontal changes bring light to the concept of adding or deleting data from individual rows or a bulk of rows together. The paper highlights methods to explain the removal of individual tuples, through predicate logic to ensure systematic explanation for the removals. The paper also briefly discusses addition of tuples due to over sampling and non-idiopathic reasons.

Implementation of Explain-Da-V

The tool uses Python, Scikit-learn and Foofah's libraries to execute the regression models and decision trees. Foofah is used as the baseline tool for the efficency's comparison and this tool attempts to extend upon Foofah's capabilities. AutoPandas leverages a search space which performs pruning using deep learning, but this performance is not reported in the paper because of its' inferior results when compared with Explain Da-V.

The paper also mentions usage of a framework SQUAREs for performing the efficiency of the SQL queries, but just like before, its' performance is also not addressed. This led to the baseline implementations being modified to level the playing field for comparison between the baseline model and Explain Da-V.

The version generation's threshold was set at 0.95 and the WINE dataset was used as an example to demonstrate that 1702 changes were logged alone.

7. Evaluating Explanations

We do not only explain the way we found the changes, we also explain the semantics behind these changes.
The explanations are evaluated based on the sense and importance.

Evaluation Methodology

- **Assessing measures:** The paper helps us in deciding the quality of the explanation. These metrics are crafted to evaluate the accuracy, applicability, and clarity of modifications among the different versions created for each given dataset.
- **Empirical Evaluation:** The Explain-Da-V method is tested against other existing methods and parameters already designed.

In the example, we can see how the decision tree can be used to evaluate the changes in the tuples.

7.1 Explanation Validity and Generalizability

- Validity of Transformations(Explanation validity)- This evaluates the effectiveness of a created transformation in reproducing the goal from the original dataset. We use this to find the number of tuples that were involved in the movement of values from start to end.
- Transformation Generalisability- This idea goes beyond validity by assessing whether the transformation can explain similar dataset versions not encountered during its creation, indicating its robustness.

7.2 Problem Definition

- **VEP** challenge entails identifying a collection of transformations that elucidate modifications between different versions of a dataset, encompassing the addition, removal, or transformation of attributes.
  It is proposed that the transformations that occurred need to be taken into account for version changes and should be used in evaluating characteristics such as validity and generalisability.

7.3 Explanation Explainability

The paper emphasizes the importance of explainability in solutions, particularly in attribute addition transformations. It aims to create a standardized measure for explainability across various models, including regressors, decision trees, and programs.Conciseness in a model relates to its simplicity, with fewer components making it more user-friendly. In regression models, it's assessed by the number of coefficients, in decision trees by the number of nodes, and in programs by the lines of code.

Explainability Concentration

Concentration assesses how information is organized in an explanation. Simpler explanations, with fewer distinct elements like mathematical operations or decision tree conditions, are easier for humans to comprehend. Higher concentration scores indicate better explanations.

7.4 On Choosing an Explanation

Explain-Da-V systematically chooses the best explanation for each change by iterating through potential origins and methodologies. The selected explanation optimizes a specific error metric, prioritizing validity and explainability based on the metrics outlined in section 7.3.

8. Empirical Evaluation

The study evaluates Explain-Da-V's performance against baselines using an ablation study and introduces a new benchmark, SDVB, with 342 dataset versions across diverse topics. The paper highlights their focus on measuring explainability, selecting optimal explanations, and empirical evaluation against other methods and benchmarks.

<u>8.1 Experimental Setup</u>

8.1.1 Benchmarks

- **Semantic Data Versioning Benchmark (SDVB):** This new benchmark features 342 dataset versions, including 136 version pairs across five diverse topics with varying data characteristics. Utilizing well-known seed datasets, it ensures comprehensive testing of change dimensions.

- **Benchmark Details:** The document provides a table detailing the benchmarks, including the number of original tuples, attributes, versions, and version pairs for each topic, such as Movies and TV shows (IMDB), NBA Players (NBA), Wines Reviews (WINE), Iris Flowers (IRIS), and Titanic Passengers (TITANIC).

<u>Comparing Explain-Da-V to the baseline tool</u>

We will use the following parameters to compare Explain-Da-V to our baseline tools:

1. **<u>Performance Overview</u>:**

Explain-Da-V surpasses baseline tools to handle different data types transformations (like numeric, categorical, textual)

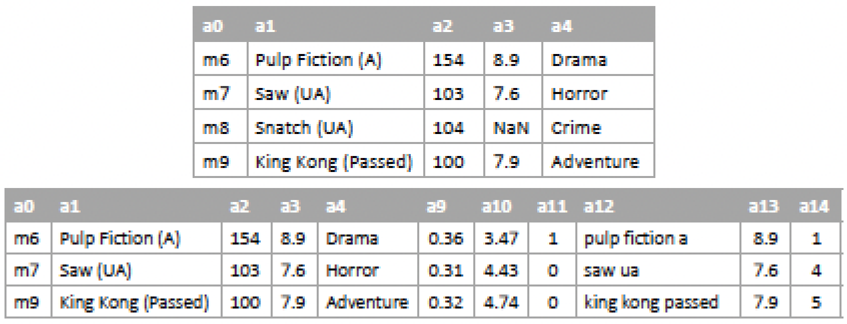
Our tool performed better in the auto-pipeline benchmark but only slightly.

Our tool holds superiority across textual transformations amongst all the alternatives.

2. **<u>Validity and generalizability</u>:**

Despite only checking explanations that are 100% valid and generalizable, the tool has significant improvement over other methods.

However, Foofah+ (a modification of Foofah) showed a small increase in average generalizability and validity (9.5%). This demonstrates the advantages of an extended search space.



a0	a1	a2	a3	a4
m6	Pulp Fiction (A)	154	8.9	Drama
m7	Saw (UA)	103	7.6	Horror
m8	Snatch (UA)	104	NaN	Crime
m9	King Kong (Passed)	100	7.9	Adventure

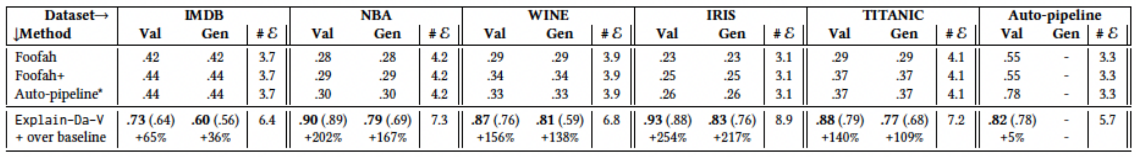
a0	a1	a2	a3	a4	a9	a10	a11	a12	a13	a14
m6	Pulp Fiction (A)	154	8.9	Drama	0.36	3.47	1	pulp fiction a	8.9	1
m7	Saw (UA)	103	7.6	Horror	0.31	4.43	0	saw ua	7.6	4
m9	King Kong (Passed)	100	7.9	Adventure	0.32	4.74	0	king kong passed	7.9	5

Figure 6: Example versions for generalizability

3. **<u>Selection of the explanation</u>:**

Explain-Da-V takes into account almost double the number of explanations when compared to the baseline tools, because of generating an extension to the origin space in the case of number-based explanations.

Table 3: Performance in terms of Validity (Val), Generalizability (Gen) and average number of explanations the method chooses from (# $\mathcal{E}$). For `Explain-Da-V`, we also report (in parenthesis) the proportion of explanations with Val/Gen score of 1.



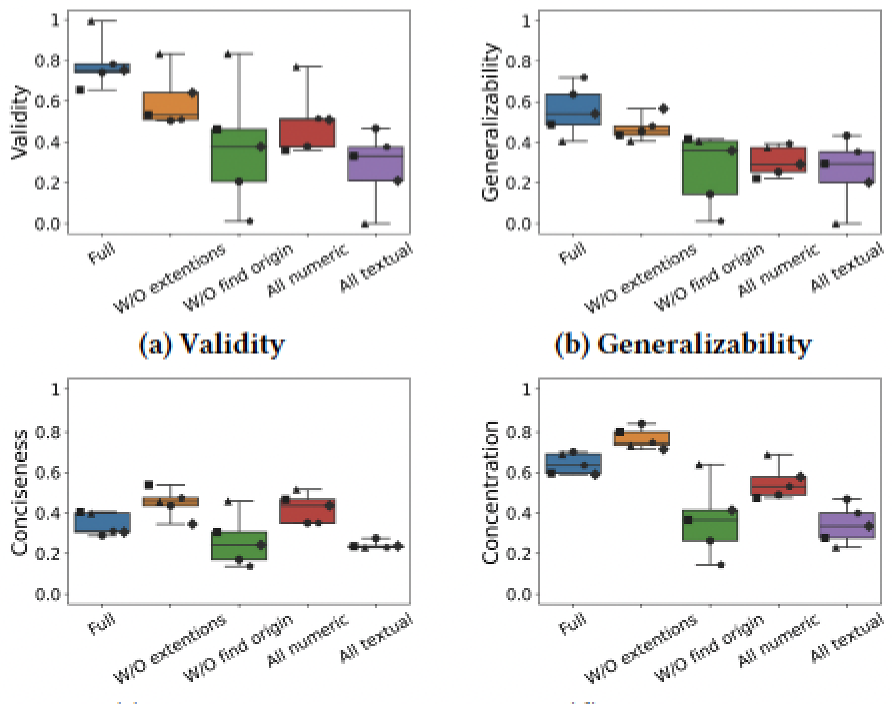
Dataset→	IMDB			NBA			WINE			IRIS			TITANIC			Auto-pipeline		
↓Method	Val	Gen	# $\mathcal{E}$	Val	Gen	# $\mathcal{E}$	Val	Gen	# $\mathcal{E}$	Val	Gen	# $\mathcal{E}$	Val	Gen	# $\mathcal{E}$	Val	Gen	# $\mathcal{E}$
Foofah	.42	.42	3.7	.28	.28	4.2	.29	.29	3.9	.23	.23	3.1	.29	.29	4.1	.55	-	3.3
Foofah+	.44	.44	3.7	.29	.29	4.2	.34	.34	3.9	.25	.25	3.1	.37	.37	4.1	.55	-	3.3
Auto-pipeline*	.44	.44	3.7	.30	.30	4.2	.33	.33	3.9	.26	.26	3.1	.37	.37	4.1	.78	-	3.3
`Explain-Da-V`	**.73** (.64)	**.60** (.56)	6.4	**.90** (.89)	**.79** (.69)	7.3	**.87** (.76)	**.81** (.59)	6.8	**.93** (.88)	**.83** (.76)	8.9	**.88** (.79)	**.77** (.68)	7.2	**.82** (.78)	-	5.7
+ over baseline	+65%	+36%		+202%	+167%		+156%	+138%		+254%	+217%		+140%	+109%		+5%	-	

4. **Database performance:**

IRIS Dataset:

Our tool has the best performance for this dataset (because this dataset has maximum numeric attributes – something over which our tool holds the highest efficiency, which is also something the baselines do not address).

IMDB Dataset: For this dataset, the tool holds the worst performance since this dataset has the highest level of textual columns and values. For example, our tool failed to count the



(a) Validity (b) Generalizability

(c) Conciseness (d) Concentration

Figure 7: Ablation Study over SDVB datasets, namely, IMDB (■), IRIS (▲), WINE (•), NBA (⬣), TITANIC (♦)

number of genres, since this deals with textual data mostly.

5. **Limitations solutions:**

The paper notes that Explain-Da-V struggles in working with complicated transformations (especially those transformations that require a high level of changes, for example, counting delimiters in textual lines).

Ablations

The next section deals with understanding how each sector of Explain-Da-V contributes individually to the tool's success, by removing each tool once and comparing the results with the other performances. We remove the following attributes one by one and check the performance:

1. Not finding the origin: Instead, we use the overall dataset to explain the transformation from the origin to the goal.

2. Not using extensions for numeric-to-numeric transformations.

3. By assuming that all datatypes in the data will be numeric or textual in nature. We do this to check how good will our performance be in these circumstances.

Ablations' findings

1. Fully implementing Explain-Da-V using finding the origin and extensions will yield the best results with the best validations and generalizations.

2. Performance is increased dramatically when we find the origin and use extensions (with improvements ranging from 30%-107% from the baseline model when we do use these methods in contrast to when we don't.

3. Contrary to this, when we end up considering all data types as either numeric or textual, we end up decreasing the validity by 35% and 64%.

Furthermore, the paper also uses conciseness and concentration to further explore interpretability. Despite interpretability being more targeted without using extensions, they also tend to be less valid and generalizable. We derive the following output:

Without finding the origin, explanations become less concise and more vague and variability also increases.

Explanations for numeric data is more easily understandable than textual explanations.

9. Conclusion

The work established a foundation for explaining semantic changes in data versioning and demonstrated the effectiveness of Explain-Da-V against multiple baselines. We realized that the performance of Explain-Da-V on numeric-to-numeric conversion was the highest. However, factors such as numeric-textual and treating attributes regardless of their type brought in a a performance decline.

Verdict- The paper explains sufficiently how well the version changes are explained by this tool, however it needs to perform better in the bad conditions too thereby making it a one-stop solution.