METAM Algorithm Implementation and Analysis

as proposed in the paper “Goal-Oriented Data Discovery” [Sainyam Galhotra et al.]

Sofiya Shtetenson and Timna Smadja

1 Introduction

In modern data-driven decision-making, the availability of large and diverse datasets presents both an opportunity and a challenge. While more data can potentially improve predictive performance, integrating heterogeneous data sources in a meaningful way is nontrivial. In this work, we implement and evaluate METAM (Minimal Essential Task Augmentation Mechanism)—a goal-oriented data discovery algorithm that iteratively identifies and joins external candidate datasets to improve a downstream task’s performance (for example - classification accuracy). By selectively augmenting the core dataset with relevant features, METAM automates what data scientists often do manually—searching for beneficial external data, joining it, and then observing performance gains.

2 Background

Traditional data augmentation methods often rely on manual feature engineering or pre-defined heuristics to select supplementary datasets. However, these approaches can be inefficient and may fail to capture the true incremental value of additional data. The METAM algorithm, as described in “METAM: Goal-Oriented Data Discovery” by Sainyam Galhotra et al., introduces a feedback loop in which candidate augmentations are evaluated based on their effect on a downstream utility function. Key aspects of the approach include:

- Data Profiles: Each candidate augmentation is characterized by task-independent measures such as join key overlap, data completeness, and feature richness.
- Adaptive Querying: The algorithm alternates between evaluating individual candidates and groups (via clustering) to efficiently explore the augmentation space.
- Minimal Augmentation Set: Augmentations are only incorporated if they yield a significant improvement in the target task’s utility, ensuring that the final augmented dataset is both parsimonious and effective.

3 Methodology Overview

3.1 Problem Formulation

Given a base dataset D_{base} and a repository of candidate datasets $\{D_1, D_2, ..., D_n\}$, the goal is to select a minimal subset $T \subseteq \{D_i\}$ such that the augmented dataset $\Gamma(D_{base}, T)$ achieves a utility $u(\Gamma(Dbase, T)) \geq \theta$, where θ is a predetermined threshold (e.g., based on accuracy for classification tasks).

3.2 Algorithm Overview

The METAM algorithm proceeds as follows:

1. Candidate Generation:

   Each candidate augmentation is processed to compute a vector of data profiles (e.g., join key overlap, completeness, and normalized feature count). An initial quality score is assigned as the mean of these profile values.

2. Clustering:

   Candidates are clustered based on their profile similarity. This step reduces redundant evaluations by grouping similar augmentations.

3. Adaptive Querying:

   The algorithm iteratively queries the utility function by merging candidate augmentations with the base dataset. Candidates are evaluated individually and as part of groups. Only candidates that yield a measurable improvement in utility (compared to a threshold □ θ) are selected.

4. Minimality Check:

   After candidate selection, a minimality check is performed by attempting to remove each selected augmentation. If removing an augmentation does not reduce the utility below □ θ, it is considered redundant and removed.

3.3 Implementation Details

Our implementation of METAM is in Python and integrates a Streamlit UI for interactive experimentation.

Key implementation details include:

- Utility Function:

The utility function is computed using an 80/20 train-test split and logistic regression accuracy as the performance metric. To ensure compatibility, non-numeric features (and missing values) are handled by filtering and imputation.

- Candidate Profiles: For each candidate, profiles are computed based on:
  - Overlap: The ratio of common join keys between the base dataset and the candidate.
  - Completeness: 1 minus the missing rate in the candidate's join key.
  - Normalized Column Count: The number of columns in the candidate dataset divided by a normalization constant.
- Streamlit UI: The UI allows the user to select from the experiments details in section 4.

4 Experimental Setup

1. Feature Engineering: Boston Housing Prices Prediction Using Partitioned Features

   In this experiment, we adopt a novel approach by taking a single, feature□rich dataset (the Boston Housing dataset) and partitioning it into distinct feature subjects. Specifically, we split the dataset into three groups, then use a reduced version of the dataset as our base. The candidate augmentations, which are the partitioned feature groups, are then merged back into the base data using METAM.

   This experimental setting is significant for the task of feature engineering, in the feature reduction perspective.

   This "reduction" capability highlights METAM's ability to achieve a minimal yet effective augmentation set, a concept discussed in the original paper in the context of minimality of the augmentation set. While the paper primarily focuses on goal-oriented data discovery, **our experiment demonstrates that the same framework can be leveraged to reduce redundancy and perform implicit feature selection**

   - Base Set: 'RM' (average number of rooms per dwelling) and 'LSTAT' (percentage of lower status of the population)—along with a dummy join key "Id" and a binary target "expensive" (1 if MEDV is at or above the median, 0 otherwise).
   - Task: predicting the likelihood of housing to be expensive (greater than the medain)
   - Candidate Augmentations: The remaining features were grouped into three candidate subjects:
     - Crime/Industrial Features (CRIM, INDUS, NOX, AGE, DIS.)
     - Zoning/Tax Features) (ZN, RAD, TAX, PTRATIO.)
     - Structural/Demographic Features (CHAS,B)
   - Results: Selected augmentation: Zoning/Tax Features

- Explanation:

  Initial Base Utility: The base model using only 'RM' and 'LSTAT' achieved an accuracy of 81.37%. Iteration 1: Merging Candidate 1 resulted in a utility of 84.31% (gain of 2.94%). In the grouping step, Candidate 2 from the identified group yielded a higher utility of 85.29%. The algorithm selected Candidate 2 (Zoning/Tax Features), which improved the base utility from 81.37% to 85.29%. A minimality check confirmed that removing this augmentation dropped the utility back to 81.37%, verifying its contribution.

  Final Performance: 85.29% accuracy, with "Zoning/Tax Features" selected as the effective augmentation.

- Conclusion: This experiment demonstrates that when a rich dataset is partitioned into distinct feature groups, METAM can be used for feature reduction—effectively identifying which groups add incremental predictive power and which are redundant. In this case, even though all candidate profiles showed high overlap and data completeness, the algorithm selected the "Zoning/Tax Features" augmentation because it provided a measurable improvement in accuracy. This result aligns with the goal-oriented nature of METAM as described in the paper, emphasizing the minimality of the augmentation set: augmentations are incorporated only when they significantly enhance the utility of the model.

2. The augmentation set is minimal: Seattle Housing Prices Prediction

   - Base set: seattle_housing_prices
   - Augementation candidates: seattle_incomes, seattle_pet_licenses, seattle_crime_rates,
   - Task: predicting the likelihood of housing to be expensive (greater than the medain) based on zip codes
   - Result: no augmentation was selected
   - Explanation: The base model already achieved a very high predictive utility (98.02% accuracy).

     We computed candidate profiles that measured the overlap of join keys, data completeness, and relative feature richness. For instance, the crime rate candidate had a high overlap (0.93) and a quality score of 0.65. Despite these favorable attributes, when we merged the candidate into the base dataset, the observed gain in utility was negligible.

     As we can see, because the base dataset already yielded near-optimal performance, the additional features did not increase the utility sufficiently. Therefore, the algorithm correctly concluded that further augmentation was unnecessary and did not select any candidate.

     This behavior is consistent with the discussion in the paper on the monotonicity of the utility function and the minimality of the augmentation set: when the base data already approaches optimal predictive performance, additional augmentations may be redundant.

3. Goal-oriented nature of METAM - Seattle Pet Ownership

    - Base set: seattle_pet_licenses (containing only “zipcode” and a weak predictor, “total_pets”)
    - Task: predicting the likelihood of having a high proportion of cats per zipcode
    - Result: selected augmentation: seattle_incomes
    - Explanation:

      In this experiment we used exactly the same dataset as in the previous example, but for a different task. The base dataset produced a modest utility of 0.6216. By incorporating the seattle_incomes dataset as a candidate augmentation METAM detected a measurable gain, improving the utility to 0.6410 and thus selected the augmentation.

    These two examples contrast highlights the goal-oriented nature of METAM: when the base dataset is already strong, additional features are unnecessary, but when the base is weak, even modest improvements from candidate augmentations are recognized and selected. This behavior aligns with the paper’s discussion on optimizing query efficiency and achieving a minimal yet effective augmentation set.

5 Conclusion

This report demonstrates that METAM’s goal-oriented data augmentation approach is effective in diverse scenarios. When the base dataset is already highly predictive, as in Experiment 2, no augmentation is needed. Conversely, when the base is weak, as in Experiment 3) even small but measurable improvements lead to the selection of useful augmentations. Moreover, the Boston Housing experiment shows that METAM can be applied to feature reduction tasks—identifying and selecting only the most beneficial feature groups. These findings underscore the algorithm’s potential for improving decision-making in real-world applications.