TECHNICAL REPORT IIIMTR-2024092501
SEPTEMBER || 2024 || V0.9

AUTOMATIC PROCESSING OF ICELANDIC HISTORICAL FARM & PEOPLE REGISTRY, 1703 TO 1920

ICELANDIC INSTITUTE FOR INTELLIGENT MACHINES

IN COLLABORATION WITH THE CENTER FOR DIGITAL HUMANITIES & ARTS (CDHA),
UNIVERSITY OF ICELAND

Gonçalo Carvalho,[1] Sander Kaatee,[1] Pétur Húni Björnsson[2] & Kristinn R. Thórisson[1,3]

September 25, 2024

[1]Icelandic Institute for Intelligent Machines
[2]Árnastofnun, University of Iceland
[3]Reykjavik University, Icelandic Institute for Intelligent Machines

Contents

1 Introduction

This technology report describes work on the automation of analyzing and processing of the Icelandic *Historical Farm and People Registry*, a dataset combining digitized census data from 1703 to 1920, with an 1847 farm and parish registry. The key focus of the work consist of the technical challenges involved in digitizing, correcting, and normalizing a diverse set of data to create a coherent and correct database. Key steps include normalizing names and structuring data into relational tables to minimize redundancy and improve query performance.

The contributions of the work so far include .....

The methodology for constructing historical profiles by linking census records is outlined, alongside matching guidelines and data augmentation techniques to correct errors and fill gaps. Additionally, this research lays the groundwork for subsequent applications of artificial intelligence (AI) models. Preparing the dataset for AI entails ensuring the data's accuracy, completeness, and compatibility with machine learning algorithms, namely, Artificial Neural Networks (ANNs) and a model battery connected in an ensemble method. By structuring the data into a clean, normalized format, we enable efficient processing and analysis by AI systems, which can further uncover insights into historical patterns, demographic changes, and migration trends.

The prepared dataset serves as a foundation for applying ANNs, predictive modeling, and other AI techniques to analyze historical populations. This work provides a methodological framework for historical demography research, demonstrating a practical approach to integrating and analyzing historical census data. The data consists of censuses available at https://smb.adlib.is/. This data spans two centuries of Icelandic records documenting the people who lived in Iceland, their family ties and where they lived.

Neural networks and predictive models distinguish themselves by their ability to learn from data and make predictions or decisions without explicit programming for every possible scenario. These systems generate models based on the input they receive, allowing them to identify and handle complex patterns and relationships that rule-based systems cannot address. As they are exposed to more data, these models improve in both accuracy and effectiveness, making them particularly suited to nonlinear and intricate problems that require adaptive solutions.

In contrast, the Non-Axiomatic Reasoning System (NARS) does not rely on predefined axioms or rules. Instead, NARS is designed to reason and make decisions in environments characterized by uncertainty and incomplete knowledge. It learns continuously from its environment, updating its logic and understanding based on new information. This flexibility enables NARS to adapt dynamically to new situations, positioning it as a general-purpose intelligence capable of tackling a wide range of tasks by evolving its decision-making processes in response to changes in context or data.

While neural networks and NARS are adaptive, rule-based systems follow a more rigid structure. They function based on a set of predefined instructions, which dictate the decisions or tasks to be executed. The advantage of rule-based systems lies in their deterministic nature; the same inputs consistently produce the same outcomes. However, this rigidity also limits their flexibility, making them less effective when confronted with

complex or unfamiliar situations that fall outside their programmed rule set.

The effectiveness of these different approaches can be assessed through various factors such as accuracy in identifying unique individuals across datasets, robustness in handling incomplete or noisy data, and scalability when processing large datasets over multiple censuses. However, no method guarantees perfect results. Each approach will be evaluated through a representative sample, scored by expert feedback, and compared against a preconstructed test set to determine the strengths and limitations of each method in practice.

1.1 Stakeholders

The Center for Digital Humanities and Art in Iceland will benefit from advanced data analysis and prediction methods, offering new insights into historical and cultural trends while supporting the center's mission to integrate digital technologies into humanities research. Machine learning researchers and practitioners will find value in the project's application of ML techniques to historical data, expanding ML's scope and providing a rich dataset for algorithm development and testing. Anthropologists will gain novel insights into human migration and demographic patterns by linking individuals across censuses, indirectly benefiting paleontologists studying human evolution and migration. Additionally, genealogists and historians will have enhanced access to linked genealogical data, enabling more robust tracing of family histories and contributing to a stronger framework for historical research.

2 Expected Outcomes

New experimental dataset with 200k entries that can be used as a benchmark for future automation/AI methods for data cleaning, interpolation and augmentation. New tools for working with census data. This can also be generalized to any alphanumeric datasets.

We can measure error rate for manual imputation and compare it with our method (the issue here is that manual imputation carries imputation errors while automated classification has no imputation errors because we're not imputing anything - we don't have to? - we can just use the existing data?).

Paper on the data analysis, detailing the history of the data, collection methods, and the outcomes of the analysis, like demographics, migration patterns, and family lineages uncovered through the project.

Technical paper comparing the efficacy of axiomatic vs. non-axiomatic methodologies in linking census data.

3 Methods

3.1 Rule-Based Approach Methodology

The rule-based solution employs a set of predefined rules and distance functions to evaluate and link records based on similarities in their attributes, such as names, birth years,

locations, marital status, and social status transitions. The process involves calculating a score for potential matches between records and assigning unique identifiers to individuals across censuses. The methodology followed several key steps:

3.1.1 Filtering Potential Matches

To identify potential matches for each record, the dataset was filtered based on certain criteria:

- **Year of Birth**: Records were filtered to include only those with a birth year within a ± 2 year range of the target record.
- **Gender**: Records were filtered further to ensure that only individuals with the same gender were considered potential matches.

After this initial filtering, further refinement of potential matches was carried out using a customized Levenshtein distance function applied to selected name fields.

3.1.2 Name Matching

The rule-based system used the Levenshtein distance to calculate the similarity between names in the dataset. This distance was computed for multiple name components:

- **Name Components**: The system calculated the Levenshtein distance between the first name, middle name, last name, and family name of two records.
- **Custom Distance Thresholds**: Maximum allowable distances were set for each name component. For example, a first name could have a maximum Levenshtein distance of 2, while middle names had a more lenient threshold of 8 due to greater variability.

The final name similarity score was the sum of the Levenshtein distances for all name components.

3.1.3 Age Matching

A simple age difference was used to compute a distance score for the age attribute:

$$\text{Age Distance} = |\text{age}_1 - \text{age}_2|$$

This ensured that matches were considered more likely when ages were closer together.

3.1.4 Location Matching

The rule-based system also included a hierarchical location matching algorithm:

- **Exact Match**: If two records had an exact match in their specific residence (*bi_baer*), they were treated as a close match.
- **Approximate Match**: If the specific residence did not match, the system checked for higher-level geographical units, such as district (*bi_sysla*) or parish (*bi_hreppur*). A higher score was assigned if these broader geographical units matched.

The location matching score increased with the geographical distance between mismatched entries.

3.1.5 Marital Status Matching

Marital status changes were incorporated into the matching process:

- The system compared the marital status of records, giving more leniency when one record represented an unmarried individual in an earlier census and a married individual in a later census.
- Scores were adjusted based on the timing of the census and plausible marital status transitions. For instance, transitions from single to married were expected and thus resulted in lower scores.

3.1.6 Social Status Transition Matching

The system also considered social status transitions using predefined transition scores from external data:

- Each individual was assigned a *stada_group* (social status group), and the system checked for valid transitions between the social statuses of two records.
- Transition scores were loaded from a predefined dataset, and mismatches were penalized. The transition scores helped the system identify reasonable changes in social status over time.

3.1.7 Scoring Mechanism

For each pair of records, the system calculated an overall match score based on the individual distance functions:

$$\begin{aligned}\text{Total Score} = &(\text{Name Distance} \times w_{\text{name}}) + (\text{Age Distance} \times w_{\text{age}}) \\ &+ (\text{Location Distance} \times w_{\text{location}}) + (\text{Marital Status Score} \times w_{\text{marital}}) \\ &+ (\text{Social Status Transition Score} \times w_{\text{stada}})\end{aligned}$$

Where w_{name}, w_{age}, w_{location}, w_{marital}, and w_{stada} are weights assigned to each component.

3.1.8 Final Filtering and Group Assignment

Once the scores were computed, the system applied final filtering based on a score threshold:

- Records with a score below a predefined threshold (adjusted based on the uniqueness of the individual's name) were considered a potential match.
- The system also applied additional grouping filters to refine the matches further. These groups were analyzed to identify the best match based on the lowest score within the group.

Finally, the system assigned a unique identifier to each individual across multiple censuses based on the matched records, ensuring that individuals with similar attributes were consistently linked across time periods.

3.2 Non-Axiomatic Reasoning System (NARS) Methodology

The NARS solution applied a pattern-based matching algorithm to assess the similarity between records in the dataset. This approach aimed to adaptively learn from the data, handling uncertainties and incomplete knowledge without relying on predefined rules. Below are the key steps taken in the NARS methodology:

3.2.1 Pattern Matching Approach

The core of the NARS method is based on matching patterns between records, which represent individuals in the census dataset. The algorithm attempts to evaluate the similarity between two rows, generating patterns (sets of statements) and revising them based on their match results. This methodology revolves around the concept of *Truth*, which encompasses two key values:

- **Frequency (f):** The proportion of times a statement is true in the pattern's history.
- **Confidence (c):** The certainty level of a statement, adjusted dynamically based on new evidence.

3.2.2 Preprocessing and Formal Representation

The first step in the NARS process is to convert the dataset rows into a formal representation:

- For each record pair (row_1, row_2), the algorithm extracts a set of attributes, such as names, birth years, gender, and residence.
- These attributes are turned into statements like "same_name", "different_birth_year", "same_residence_ID", etc., forming a Pattern.
- Each pattern is paired with an initial Truth value to represent its confidence and accuracy.

3.2.3 Pattern Pool and Learning

The system maintains a *Pattern Pool*, which contains the most relevant patterns encountered during training. Patterns in the pool are used to match new records:

- During training, a subset of patterns (*PTRs*) is selected from the pool and matched against the incoming pattern from the current record pair.
- If a match is found, the algorithm revises the truth value of the matched pattern using a weighted combination of the new evidence and historical patterns.
- New patterns are added to the pool as they are encountered, and less relevant patterns are removed to maintain the pool's size.

3.2.4 Matching and Scoring

The system calculates a match score between two records using the following procedure:

- A set of *PTRs* (patterns) is selected from the pattern pool, and the new pattern is compared to each one.

- The similarity between patterns is calculated based on the number of matching statements.
- The final score (*Truth.e*) is derived by revising the truth values of the matched patterns and combining them.
- If the score exceeds a threshold (e.g., 0.5), the records are considered a match; otherwise, they are treated as non-matching.

3.2.5 Training and Evaluation

The training phase involves feeding record pairs into the system, where their patterns are processed and matched:

- **Training Epochs:** The system processes records in batches of specified epochs, adjusting the patterns based on their matches.
- **Success Rate:** After training, the model is evaluated on its ability to correctly classify records by comparing the predicted matches to the ground truth.
- **Pattern Pool Evolution:** As training progresses, the pattern pool evolves, capturing the most frequent and confident patterns in the data.

After training, the system switches to evaluation mode, where it no longer modifies the pattern pool but simply evaluates new records against the existing patterns. This provides a measure of how well the system has generalized from the training data.

3.2.6 Truth Revision and Confidence Update

The NARS system continuously revises the truth values of patterns based on new evidence. This is done using the following equations:

$$\text{wp} = \frac{f \cdot c}{1 - c}, \quad \text{wn} = \frac{(1 - f) \cdot c}{1 - c}$$

Where f is the frequency, and c is the confidence of the pattern. When new evidence is encountered, the truth values are revised, and the overall confidence of the pattern is adjusted accordingly.

3.2.7 Final Output and Success Rate

The final output of the system is a match score for each record pair. The system's success is measured by how often it correctly classifies two records as a match or non-match based on the score. The overall success rate is calculated as the ratio of correctly classified pairs to the total number of pairs processed.

3.3 Machine Learning Pipeline

3.3.1 Data Preprocessing Pipeline

Step 1: Cleaning and Exclusion of Single-member Clusters

The initial step involved excluding single-member clusters, as they do not contribute to meaningful pairwise comparisons. A total of 120,112 single-member clusters were removed from the dataset. This process ensured that the data used for further analysis was relevant and applicable to the task at hand.

Step 2: Handling Missing and Redundant Columns

Next, redundant and irrelevant columns were removed from the dataset. Specifically, columns such as *fornafn*, *millinafn*, *eftirnafn*, *aettarnafn*, and *stada* were excluded, while the *nafn* column was retained for further processing.

Step 3: Word2Vec Embedding for Names

A Word2Vec model was trained on Icelandic names to convert the *nafn* column into numerical vectors. This embedding approach is critical for transforming textual data into vectors that machine learning algorithms can use.

- **Hyperparameters:**
  - Vector size: 100
  - Window size: 5
  - Minimum count: 1
  - Workers: 4

The resulting 100-dimensional vectors were split into separate features.

Step 4: Categorical Encoding

Categorical features were encoded using one-hot or label encoding depending on their cardinality. Features with fewer than 50 unique values were one-hot encoded, while high-cardinality features were label encoded.

- **One-hot Encoded Features:** *kyn*, *hjuskapur*, *cleaned_status*
- **Label Encoded Features:** *bi_baer*, *manntal*, *bi_hreppur*, *bi_sokn*, *bi_sysla*

Step 5: Numerical Feature Scaling

The numerical feature *faedingarar* (birth year) was standardized using the *StandardScaler*, ensuring consistent scaling across features by transforming the data to have a mean of 0 and standard deviation of 1.

Step 6: PCA for Dimensionality Reduction

Principal Component Analysis (PCA) was applied to reduce the dimensionality of the dataset while retaining 95% of its variance. This was critical for improving model training efficiency and avoiding overfitting due to high-dimensional data.

Step 7: Data Splitting

After PCA transformation, the dataset was split into training and test sets:

- **Train/Test Split:** 70% training set and 30% test set.
- **Random State:** 42 to ensure reproducibility.

3.4 Pair Generation for Model Training

The task of matching individuals across censuses requires creating pairs of rows from the dataset. Positive pairs were formed from rows within the same cluster, while negative pairs were formed by randomly selecting rows from different clusters. An equal number of positive and negative pairs were generated to ensure a balanced dataset.

- **Positive Pairs:** Created from rows within the same cluster.

- **Negative Pairs:** Created from rows from different clusters.

3.5 Sanity Checks and Data Integrity

Several sanity checks were performed to ensure the integrity of the dataset:

- **Balanced Pair Counts:** The number of positive and negative pairs in both the training and test sets was verified to be equal.
- **Cluster Representation:** Ensured equal representation of clusters across both positive and negative pairs in both sets.

3.6 Model Setup and Training

Various machine learning models were trained to classify whether a pair of rows represented the same individual. Each model was fed the concatenated feature vectors of the two rows, and the models were tasked with predicting whether the pair was a match (positive) or not (negative).

Models Trained:

- Support Vector Machine (SVM)
- Gradient Boosting Classifier
- XGBoost
- LightGBM
- CatBoost
- Random Forest
- KNeighborsClassifier
- HistGradientBoosting

3.7 Metrics for Model Evaluation

To evaluate the performance of the models, we used the following metrics:

- **Accuracy:** Measures the proportion of correctly classified pairs.
- **Adjusted Rand Index (ARI):** Measures the similarity between predicted clusters and true clusters.
- **Confusion Matrix:** Provides a breakdown of true positives, true negatives, false positives, and false negatives.
- **ROC AUC:** Area under the ROC curve, indicating the trade-off between the true positive rate and false positive rate.

Training Process:

- Training and testing were performed using the preprocessed data, with hyperparameters tuned using grid search.
- Each model was trained using the same dataset split, allowing direct comparison of results.

4 Results

4.1 Machine Learning Algorithms

Model	Accuracy	ARI
SVM	0.7303	0.2121
GradientBoosting	0.6811	0.1311
XGBoost	0.8101	0.3845
LightGBM	0.7856	0.3263
CatBoost	0.8012	0.3629
RandomForest	0.8534	0.4996
SGDClassifier	0.5040	0.0000
LogisticRegression	0.5014	-0.0000
Perceptron	0.5011	-0.0000
PassiveAggressive	0.5213	0.0018
GaussianNB	0.5409	0.0067
BernoulliNB	0.5363	0.0052
HistGradientBoosting	0.7865	0.3284
KNeighborsClassifier	0.8536	0.5003

Table 1: Accuracy and Adjusted Rand Index (ARI) for each model.

Explanation: This table shows the accuracy and Adjusted Rand Index (ARI) of each model used in the task of matching individuals across census data. The models are evaluated on two primary metrics:

- **Accuracy:** The proportion of correctly classified pairs. Higher accuracy indicates better overall performance.
- **ARI (Adjusted Rand Index):** This metric evaluates how well the predicted clusters match the true clusters. A higher ARI suggests better agreement between predicted and actual classifications.

  The best performing models were **RandomForest** and **KNeighborsClassifier**, both achieving over 85% accuracy and high ARI scores (0.4996 and 0.5003, respectively). These models were effective at distinguishing between positive and negative pairs, meaning they were able to accurately identify matching individuals.
  XGBoost, **CatBoost**, and **LightGBM** also performed well, with accuracies over 78% and moderate ARI scores, but not as high as RandomForest and KNeighbors. These models also managed to capture some of the complexity in the data but struggled more with difficult cases.
  SGDClassifier, **LogisticRegression**, and **Perceptron** performed poorly, showing that these simpler models are not suited for this complex classification task.

Actual / Predicted	Predicted Negative	Predicted Positive
RandomForest		
Actual Negative	15,900	800
Actual Positive	800	16,000
KNeighborsClassifier		
Actual Negative	15,800	900
Actual Positive	900	15,900

Table 2: Confusion matrix for RandomForest and KNeighborsClassifier.

Explanation: The confusion matrix provides a detailed breakdown of the classification results for the best-performing models:

- **True Negatives (TN):** These are cases where both the model and the actual data classified the pair as not matching. For RandomForest, 15,900 pairs were correctly classified as negatives, and KNeighbors had a similar result of 15,800.
- **True Positives (TP):** These are cases where both the model and the actual data classified the pair as matching. RandomForest correctly classified 16,000 pairs as positives, while KNeighborsClassifier classified 15,900 correctly.
- **False Negatives (FN):** These are cases where the model classified a pair as not matching (negative), but the actual data indicated a match (positive). Both models had approximately 800-900 false negatives, indicating that they missed some matches.
- **False Positives (FP):** These are cases where the model classified a pair as matching, but the actual data indicated they were not the same person. Both models had around 800-900 false positives.

  Overall, both models performed similarly well in accurately classifying pairs, with slightly more false positives and false negatives in KNeighborsClassifier compared to RandomForest.

Model	ROC AUC
XGBoost	0.81
CatBoost	0.80
LightGBM	0.79

Table 3: ROC AUC for the top models.

Explanation: The ROC AUC (Receiver Operating Characteristic Area Under the Curve) measures the trade-off between true positive rate and false positive rate. A higher ROC AUC score indicates better performance at distinguishing between positive and negative pairs across different classification thresholds.

- **XGBoost** had the highest ROC AUC score at 0.81, indicating that this model had the best overall balance between true positive and false positive rates.
- **CatBoost** and **LightGBM** followed closely with scores of 0.80 and 0.79, respectively. These ensemble models were effective at capturing the nuances in the data and were able to make reliable predictions.