# Combining Active and Transfer Learning for Entity Resolution

Master Thesis

presented by
Jonas Kaendler
Matriculation Number 1419369

submitted to the
Data and Web Science Group
Prof. Dr. Christian Bizer
University of Mannheim

July 2020

# Combining Active and Transfer Learning for Entity Resolution

Jonas Kaendler

## Abstract

In this thesis, the problem of transferring knowledge from a source domain to resolve entities in a target domain within the same semantic topic was researched. Experiments have shown that learning only on labeled source domain data and applying the model on the target domain data (i.e., naive transfer) achieves better results with ensemble methods and can outperform unsupervised matching in the target domain. However, in most cases, the naive transfer results are significantly worse than learning on labeled data from the target domain. Merely incorporating all labeled source domain data into the labeled set of an active learning method has shown to hinder the active learning model from adapting to the target domain. Based on the initial experiments' findings, an existing active learning method for entity resolution was modified to incorporate source domain knowledge and address the identified pitfalls. The final method, called ATLX, was tested on data from three different semantic topics with varying difficulty. ATLX has shown to outperform its baseline most of the time for the two difficult topics and all the time for the easier one. ATLX leads to better models in terms of the quality and robustness of the model learned. The method, thus, successfully combines active and transfer learning for entity resolution. So far, this has only been done in entity resolution research with the use of deep learning.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Nowadays, collecting massive amounts of data is common in many businesses, government agencies, and scientific research projects. As many different institutions collect data within the same semantic topic, this data can be integrated to create a more holistic view of each topic and gain meaningful insights. Nevertheless, data integration comes with many challenges. One of the major challenges here is called Entity Resolution (ER), which is about identifying different entity representations that refer to the same real-world entity across different data sources and resolving these matches in order to not have duplicates in the integrated database [15]. This challenge is tricky as often, no unique identifier for real-world entities across different sources exists. Additionally, these matches are often fuzzy, due to different data representations, different attributes describing these entities, misspellings, just to name a few reasons [48, 15]. Hence, ER is a popular research topic that has been tackled by many different research areas for the last decades. That is also why many different synonyms for ER exist in the literature, like Identity Resolution, Data Matching, Record Linkage [48, 15].

In recent years many methods based on supervised machine learning were proposed in order to learn effective classification models from labeled training data that can be used to classify a record pair (i.e., a record from one source and a record from another source that potentially refer to the same real-world entity) to a non-match or a match. In general, though, obtaining labeled data to train a model is for many real-world applications, a complicated and expensive process [6]. That is also the case in the area of ER [75]. Hence, concepts from the machine learning area that try to mitigate the need for expensive labeled data, like transfer learning and active learning, are gaining attention in recent ER research. Transfer learning, in general, is about training a model to learn a *learning task* (i.e., classifying a record pair to a match or a non-match, as in ER) on data from a so-called *source*

*domain*, where lots of labeled data is available, to then transfer learned knowledge from the trained model to a related *target domain* and *learning task*, where only a little or no labeled data is available [55, 80]. In contrast, active learning tries to reduce the label-complexity (i.e., the need for labeled training data) by querying only the labels of the most informative examples [6, 64].

## 1.1 Problem Setting and Contributions

This thesis aims to research how existing knowledge can be incorporated into an active learning method that learns to resolve entities between two data sources. If labeled record pairs from another domain but the same semantic topic (the source domain) are available, this knowledge can be reused to boost an active learning process that learns a classification model that resolves entities in another domain (the target domain). The following research questions are addressed:

- How does learning a classification model only on labeled record pairs from the source domain perform in resolving entities in the target domain?
- How can the knowledge in the form of labeled record pairs from the source domain be incorporated into an active learning process?

This thesis's main contribution is the adaptation of an existing active learning method for ER [62] to incorporate knowledge from another domain. The final method thus combines active and transfer learning to resolve entities between two data sources while achieving superior performance than suitable baselines and reducing the label-complexity. The combination of these two concepts for ER has been achieved in earlier work [32] only through deep learning [39]. The method is evaluated on data from three different semantic topics of different difficulty. The whole experimental setup, including data sets, implementations, and results, are made publicly available on GitHub[1].

## 1.2 Structure of this Thesis

In chapter 2, chapter 3, and chapter 4 background knowledge regarding ER, active learning, and transfer learning including related work sections is provided. The methodology of this thesis is described in chapter 5. The data profiling and the experimental results are listed in chapter 6 and chapter 7, respectively. Additional material is provided in Appendix A and Appendix B.

[1] https://github.com/JayKay0104/ma-atl-for-er

# Chapter 2

# Entity Resolution

Entity Resolution (ER) is about identifying records within a collection of entities that describe the same real-world entity [25, 15]. Ironically, ER has many different synonyms like Identity Resolution, Duplicate Detection, Record Linkage, and so forth [48, 15]. Although the specific application may differ slightly between them, they all refer to the main concept of identifying records that refer to the same real-world entity. Thus, they all require a similarity measure to compare records and a mechanism to determine whether or not the records describe the same real-world entity [48].

## 2.1 Motivation and Challenges

It is evident that combining data from different sources can lead to a more holistic view of a particular semantic topic. However, the more, the better is not necessarily always the case. More critical than possessing much data is the quality of the data and the knowledge that can be retrieved from the data.

Suppose data from different sources are combined without resolving entity descriptions that refer to the same real entity. Doing this would result in duplicate, incomplete (i.e., one entity description would lack specific information that would be present in another description of the same entity) or incorrect information (i.e., to take the number of records as the number of all entities contained in the integrated collection would be incorrect) in the integrated data set, to name just a few problems. There is no denying that integrating data without ER will hinder the extraction of meaningful insights from the data, and the benefits of data integration will be lost.

In the case where a unique identifier exists for all entities across all sources, ER is a reasonably simple task, since all entities can easily be linked together based on

the exact similarity of the unique identifier. However, in most cases, such a unique identifier is not provided. In addition, the descriptions of the entities are often fuzzy due to spelling mistakes or errors in general, different attributes describing an entity (i.e., heterogeneous schemata), different formats (e.g., one source uses abbreviations, and another does not), or one source simply contains outdated data [48].

Nevertheless, the problem that descriptions of the same entity are not necessarily identical is not the only challenge faced in ER. Another challenge derives from the question of which entries to compare. After all, a comparison of each entry with every other entry would increase quadratically in the number of entries and thus quickly lead to a too extensive computational effort. Because of this, most ER approaches apply so-called blocking, filtering, or indexing at first. These techniques mitigate this impracticable computational effort by reducing the comparisons to only relevant candidates of record pairs. The reduced set of record pairs can then, in a subsequent step, be compared in detail and divided into matches and non-matches [56, 16]. The challenges in ER can, therefore, be divided into two subsequent steps: (1) selecting pairwise comparison candidates (the selection step) and (2) the actual comparison step to determine whether they refer to the same real-world entity (the determination step) [56].

This work focuses mainly on the second step. However, since the first step is a prerequisite when working with data sources containing more than a few thousand records, a short introduction to the selection step is given before focusing on the determination step. It is essential to highlight that in this thesis, the problem of resolving entities across two data sources with aligned schemata is considered.

## 2.2 The Selection Step

In principle, ER can be performed in a way that can be referred to as *exhaustive search*. There each entity described in an entity collection gets compared with each entity description in another entity collection. However, as mentioned before, simply performing an *exhaustive search* does not scale to large data sets due to the quadratic time complexity. For this reason, in order to perform ER when working with large data sets, the number of pairwise comparison candidates is significantly reduced in the selection step before the actual comparison of the entity descriptions with each other.

### 2.2.1 Blocking

One way to do this is called blocking, which groups similar[1] entities into blocks so that only entries within the same block need to be compared. The goal is to place every pair of matching entities in at least one common block while minimizing the number of unnecessary comparisons. Often a transformation function maps an entity description to a set of so-called blocking keys [56]. A blocking key can be considered a signature of the entity description, and based on its blocking key value, a record gets assigned to one or more blocks from an assignment function [56]. Note that a blocking key from an entity is also called an inverted index in the literature. Therefore, the whole process is also referred to as indexing [56, 15, 16]. For the concept of blocking, scientists have proposed a vast amount of techniques in the last decades. The interested reader is referred to a recent survey of Papadakis et al. [56] for more information about these techniques and their differences.

### 2.2.2 Filtering

Another approach to reducing the number of comparisons is called filtering, which is less generic than blocking. It assumes that entity descriptions are based on strings or sets and that they match if the strings or sets exceed a certain similarity threshold [56]. One common technique used in ER for filtering are *string similarity joins* [56]. The core of *similarity joins* is the similarity measure. Here, either two strings are compared by considering them as a *sequence of characters* or as *sets of tokens*. The former analyzes how many characters need to be changed to get the same string as the one being compared. In the latter, the similarity is measured based on the match of elements in the two sets (i.e., common tokens) [56]. Tokenization is used in order to transform strings into sets of tokens. A fundamental example of tokenization would be the extraction of all substrings separated by a space, or better, extraction of so-called q-grams. Here $q$ refers to the length of the desired tokens. Suitable set-based similarity measures can then be applied, and a certain threshold value determines whether the data set pairs are passed on to the next step of the ER, the determination step.

### 2.2.3 Evaluation

Even though it is often crucial to apply one of these concepts at the start of ER, there is also the potential adverse effect of missing true matches because they got filtered or blocked prior to the more detailed comparison. Therefore, the selection step already contributes to the overall performance of ER that is often measured

[1] similarity depends upon the characteristics of the data to be matched

based on its effectiveness and efficiency. Effectiveness refers to the portion of identified actual matches to the total number of true matches. Efficiency, on the other hand, refers to the computing effort required to determine these matching entities, which is often measured by the amount of performed comparisons [56].

It is quite apparent that there is a trade-off between effectiveness and efficiency [56]. Performance metrics that are commonly used in order to evaluate a blocking or filtering strategy are *Pair Completeness (PC)* and *Reduction Ratio (RR)*. The former indicates the effectiveness of a blocking strategy. It estimates how many matches of all existing comparisons with the selected pairwise comparisons can be detected. The efficiency of a blocking strategy, on the other hand, is commonly evaluated using the *Reduction Ratio (RR)* that measures the reduction in the number of pairwise comparisons compared to the exhaustive search approach. Both performance metrics take values in the interval $[0, 1]$ with higher values indicating higher effectiveness or efficiency.

## 2.3 The Determination Step

After the number of pairwise comparisons is reduced, more detailed comparisons can be conducted in the so-called determination step [56]. In the literature, this is also called simply String Matching (i.e., if the entity is described using strings) or more generic Entity Matching or Data Matching [56, 24]. The concepts needed for the determination step of ER are similarity measures and a mechanism to classify record pairs into matches and non-matches, which will be explained in the subsequent subsections. It is important to note that ER is considered a binary classification problem in this thesis. However, in the literature, it is sometimes seen as a multi-class classification problem. For example, the classification of record pairs could be made not only into matches or mismatches but also into potential matches [15]. Though, the concepts mentioned below apply to any type of classification.

### 2.3.1 Similarity or Distance Measures

Similarity or distance measures are needed in order to compare two entities with each other. While a similarity measure outputs a value in the range of zero and one, where a higher value indicates that two strings are more similar than a lower value, a distance measure is not bound to this range. For distance measures, smaller values indicate greater similarity [24].

Different categories for similarity measures exist. According to Doan et al. [24] the most common similarity measures can be classified into four groups.

**Sequence-based Similarity Measures** calculate a cost between two strings based on the number of insertions, deletions, and replacements of characters required to convert one string to the other [24, 48]. Measures that belong to this group are Levenshtein or also called Edit-distance, Needleman-Wunsch, Affine Gap, Smith-Waterman, Jaro, and Jaro-Winkler [24].

**Set-based Similarity Measures** consider strings as sets of tokens and use set-related characteristics to calculate similarity values. Measures are Overlap, Jaccard, and TF/IDF [24].

**Hybrid Similarity Measures** combine the benefits of the two previously mentioned groups of similarity measures. Measures are Generalized Jaccard, Soft TF/IDF, and Monge-Elkan [24].

**Phonetic Similarity Measures** are based on a completely different approach. Theses measures compare strings based on their sound [24]. Here the most common one is called Soundex. For Soundex, many different variations or extensions like Phonex, Phonix, NYSIIS, Double-Metaphone, and Fuzzy Soundex exist [14].

Explanations of all these measures can be found in textbooks about ER or data integration in general [24, 48, 15]. Additionally, more specific similarity measures exist, e.g., for personal names where the work by Peter Christen [14] provides a good overview. The similarity measures or distance measures used for the experiments in this thesis are explained in the following.

### Levenshtein

Also called the edit distance $dist_{levenshtein}(v_1, v_2)$ computes the minimal cost of transforming $v_1$ into $v_2$, where three different operations, namely *delete a character*, *insert a character*, *substitute a character with another*, each with a cost of one can be performed [24]. So for two rather dissimilar values more such operations are required and hence a higher $dist_{levenshtein}$ is the result. The distance measure $dist_{levenshtein}$ can be converted into a similarity measure with values ranging from zero to one by simply calculating the following:

$$sim_{levenshtein}(v_1, v_2) = 1 - \frac{dist_{levenshtein}}{max(length(v_1), length(v_2))} \text{[24]} \tag{2.1}$$

### Jaccard

For the Jaccard similarity measure, the two strings are converted into sets (e.g., by extracting tokens of a specific length like trigrams, simply extracting words separated by white space or any other tokenization technique). The formal definition of the Jaccard similarity measure is:

$$sim_{jaccard}(v_1, v_2) = \frac{|common\_tokens|}{(|tokens_{v1}| + |tokens_{v2}|) - |common\_tokens|} \text{[24]} \tag{2.2}$$

The resulting $sim_{jaccard}$ is in the interval $[0, 1]$ with one indicating exact similarity.

### Relaxed Jaccard with inner Levenshtein

This similarity measure creates tokens and is calculated in the same way as Jaccard. However, tokens count as common tokens if the calculated Levenshtein similarity across two tokens exceeds a specified threshold.

### Containment and Exact Similarity

Containment has a similarity score of one if the two strings share at least one word and zero otherwise. The exact similarity is one only if the two strings do not differ at all and zero if they already differ at one character.

### Cosine Similarity with TF/IDF Weighting

The intention behind this measure is to consider two strings as more similar if the words or terms they share are rather uncommon [24]. TF, which stands for term frequency measures how often a particular term occurs in a string, and IDF, which stands for Inverse Document Frequency is a measure to determine if a particular word is common or not. In order to calculate IDF, all values for the corresponding attribute across both sources are considered.

Even though similarity measures are required to define how similar two entity descriptions are, they cannot be applied alone when resolving entities. There is always either a mechanism or person (f.i. a domain expert) required that classifies with the help of similarity scores record pairs into matches and non-matches. Therefore, many different solutions exist in the literature. Most of them can be categorized into four groups: Rule-Based Matching, Supervised Matching, Unsupervised Matching or Clustering, and Probabilistic Approaches [24, 15]. Note that the names of the categories in the literature sometimes differ, although they have the same meaning.

### 2.3.2 Probabilistic Approaches

These approaches for ER make their matching decisions based on probabilistic reasoning, where some of them require besides domain knowledge the need of training data and some, not [24]. Note that this group is therefore linked to Unsupervised Matching and Supervised Matching, and in some cases, there is no complete distinction. However, all approaches within this group are based on probabilistic models, but could still be assigned to the other two groups. The pioneer works in ER by Newcombe et al. [50] and Fellegi and Sunter [26] in the late 50s and 60s, respectively, fall into this category but also more recent works for probabilistic approaches for ER exist. Examples are graphical probabilistic models like Bayesian networks, the EM algorithm, and other generative models for ER [33, 65, 41, 42]. More information and explanations of some of these approaches can be found in [24].

### 2.3.3 Rule-Based Matching

As the name of this category implies, approaches that fall into this group use handmade rules to classify record pairs written by a domain expert. Here one possibility is to create a linear combination of the similarity scores for some selected attributes and similarity measures [24]. So the similarity of the attribute values from the record pairs is calculated using the selected similarity measure, and then a weight is assigned to the score. It is ensured that the weights sum up to one so that the final similarity score that results from the linear combination is at most one for greatest similarity. If the scores are all zero for all attributes in the rule, the overall score will be zero. That indicates no similarity at all for the record pairs that are compared. The classification of matches and non-matches is done by setting a threshold. For instance, a threshold at $0.7$ can be set. All record pairs with an overall score below that threshold are classified as non-matches and all greater or equal than $0.7$ as matches [24]. Due to that fact, this kind of approach is also called threshold-based matching [15]. A similar approach does relax the constraint that the weights need to sum up to one by using the logistic function. The logistic function ensures that the overall score at the end is still in the range of $[0, 1]$. This approach achieves that the overall score increases less once a certain threshold is exceeded [24]. Of course, also approaches that create more complex rules exist and fall into this category [24]. However, the problem with the approaches described here is that it is often hard to find the right weights, and it is also problematic in domains that are rather complicated [24].

### 2.3.4 Unsupervised Matching or Clustering

Into this category fall all approaches that use traditional clustering techniques in order to cluster the matching record pairs into the same clusters but also approaches which do not apply any clustering technique but instead apply unsupervised learning differently. Examples for approaches using clustering are described in the work of McCallum et al. [45] or Cohen and Richman [18]. In general, unsupervised learning means that no information about the class or label of the class is required. Hence, the learning algorithm does learn alone from the characteristics of the record pairs. Therefore, clustering techniques also fall into this category. In this thesis, one very recent approach for unsupervised matching is used as a baseline. One could consider it a threshold-based approach to ER, but the threshold is not defined by a domain expert but instead learned from the data in an unsupervised way [62]. The method is explained in subsection 3.4.3.

### 2.3.5 Supervised Matching

This category refers to all approaches that apply supervised learning to learn the matching rules automatically from training data. Here basically, everything which is considered relevant for the matching task can be considered a feature. To train a supervised classification model for a subset of record pairs, not only the feature values are provided but also the correct labels that define whether a record pair is a match. The learning algorithm can then learn from these examples and build a classification model that can be applied to unseen data to let the model decide whether a certain record pair is a match or not.

The advantages of supervised learning in ER compared to defining a handmade rule is that the learning algorithm automatically selects the features that it considers as essential based on what it has learned from the training data [24]. In rule-based learning with handmade rules, this needs to be done by a domain expert. However, this is labor-intensive and, as mentioned before, rather complicated depending on the complexity of the domain [24]. Additionally, a learning algorithm can detect much more complicated patterns from the training data and hence also can build more sophisticated matching rules [24]. For this reason, most supervised learning approaches to ER have shown to outperform non-learning approaches [38].

Nevertheless, supervised learning algorithms have one major disadvantage: the need for a sufficient amount of labeled training data, which is difficult and time-consuming to obtain. Besides that, the performance of the classification model learned will depend highly on the quality of the labeled data used for training. This fact is precisely referring to the problem statement of this thesis. The subsequent chapters (chapter 3 and chapter 4) will focus on learning techniques that try

to reduce the amount of labeled training data required while still achieving good performance results.

In the subsequent subsections, prerequisites for supervised learning and existing learning algorithms used in this thesis are briefly explained.

### Performing Supervised Learning

A supervised learning algorithm needs labeled training data from which it can learn a model. On the other hand, a test set is needed to validate how the final model performs on unseen data (i.e., if it generalizes well). Here the test set must be kept entirely out of the training phase. One possible way for splitting the whole data into training and test set is using cross-validation.

In cross-validation, the whole data is split into $k$ folds. Then $k-1$ folds are used for training, and the $k^{\text{th}}$ fold is used for testing. That is done in a round-robin fashion until each fold was used once for testing. Then the result of each run can be averaged [47]. This approach is often also used for model selection as one can test different hyperparameter settings and choose the best performing one [31].

Another approach is using a dedicated hold-out test set for validation [31]. Note that the cross-validation approach provides a better estimation of the generalization capability of the model learned as the performance of the model in the hold-out set approach highly depends on which examples are included in the training set and which in the test set [31]. Additionally, using less data for training also makes the model weaker as most often, the more data used for training, the better the final model [31].

In general, it can be said that strictly speaking, there are slightly different approaches for carrying out the learning, optimization, and evaluation steps. However, they all adhere to the rule that disjoint samples must be used to construct (training set) and evaluate the model (test set). To compare two different models, they must also be tested on the same test set.

### Logistic Regression

Logistic Regression is a parametric learning algorithm that is linear in the parameters and used for binary classification instead of regression. Hence, the name Logistic Regression can be a little bit misleading. Other names for Logistic Regression are among others maximum-entropy classification or log-linear classifier [57]. Logistic Regression follows the discriminative approach to fit a model of the form $P(y|x)$ directly instead of the generative approach where the joint probability $P(y, x)$ is modeled and then $P(y|x)$ can be derived from there by conditioning on $x$ [47]. The variable $y$ is a binary output with $Y \in \{0, 1\}$ (i.e., match or non-match

in ER) and $x$ is the input or predictor(s) (i.e., in case of ER a record pair of two records coming from two different data sources represented by $n$ numeric features like similarity scores. Out of simplicity, it is assumed that a record pair is described by only one predictor $x$ that can be used to predict $y$.

Thus, instead of directly modeling whether a record pair is a match, Logistic Regression models the probability that $x$ is a match. In any case, the probabilities can be easily converted into the two classes by setting a threshold, for example, at $0.5$ and classifying $x$ as a match if $p(x) > 0.5$ and as a mismatch, if $p(x) \leq 0.5$. Still, first a function that maps $x$ to $p(x)$, which as it is a probability, strictly needs to be in the range of zero and one. In Logistic Regression a function called *logistic function* (also called sigmoid function) is used, which maps every real-valued number into a value that lies in the range of 0 and 1 [31]:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}} \tag{2.3}$$

This equation can be rewritten to:

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0+\beta_1 X} \tag{2.4}$$

Here the resulting value of $p(x)/(1 - p(x))$ is called the odds that can take on every value in the range of zero and $\infty$, where a value close to zero indicates that the record pair is a match with only very low probability and vice versa [31]. When taking the logarithm of both sides the following equation is derived

$$\log(\frac{p(x)}{1 - p(x)}) = \beta_0 + \beta_1 X \tag{2.5}$$

This equation is called the log-odds which, as the equation shows, is linear in $x$. This is also the reason why a Logistic Regression in the general case, as described here, is belonging to the family of linear models. The weights or also called coefficients $\beta_0$ and $\beta_1$ must be learned from the training data with a method called *Maximum Likelihood*. This method, in turn, must be estimated using certain optimization algorithms that are well-explained in the textbook by Kevin Murphy [47]. In *Maximum Likelihood*, the weights shall represent the training data in the best possible way so that record pairs within the training data that are actual matches are getting a probability close to one with the weights estimated and non-matches a probability close to zero. For making predictions on unseen data the values of the input variables (recap in case of ER, i.e., the similarity scores of a particular record pair and in the example explained above only one similarity score is used) as well as the coefficients estimated based on the training data need to be plugged

into Equation 2.3 to get the probability of that record pair belonging to the class of matches. If the probability is close to one, the model is confident that this record pair is a match. If the probability is close to $0.5$, the model is somewhat less confident, and one also says the record pair is close to the decision boundary.

In this thesis also LogisticRegressionCV is used from the scikit-learn package (introduced in section 2.3.5). The CV stands for cross-validation. It means that this particular version of a Logistic Regression automatically searches for the best hyperparameters.

### Support Vector Machines

A support vector machine (SVM) is an extension of a support vector classifier (SVC), which in turn, is a generalization of the maximum margin classifier [31]. The Maximal Margin Classifier looks for a hyperplane that separates the two classes and has the maximum distance to the nearest training points [31]. Therefore, it assumes that the data is linearly separable, and since this is not often the case in real-world data sets, this simple classifier often cannot be applied [31]. For this reason, SVC circumvents the limitation that the data must be linearly separable by introducing so-called slack variables that penalize any data point on the wrong side of the hyperplane when calculating the margins [31].

SVM is often used when talking about SVC. However, as mentioned above, the two concepts are not the same, but SVM is instead an extension of SVC that addresses the problem when the decision boundary is non-linear. With the help of a kernel, it offers the possibility to transform the data into a higher-dimensional space where a linear separation can be found [31].

### Decision Trees

Based on labeled training data, a Decision Tree Classifier generates a model in the form of a tree data structure that humans can (most of the time) interpret. A particular classification decision follows a clear path through the decision tree down to a leaf node representing one label.

A decision tree consists of a root node characterized by the fact that it only has outgoing edges and no incoming edge. In contrast, an internal node has at most one incoming edge and at least two outgoing edges. A leaf node, which provides the final decision or class, only has one incoming edge. Internal nodes and the root node are representing one feature[2] that gets split in a certain way. Each leaf node represents a label.

[2] one feature per node is at least the most common approach

For learning a decision tree, many different algorithms exist (e.g., CART, Hunt's algorithm, ID3, C4.5)[3] that use a greedy strategy as finding the optimal decision tree is known to be NP-hard [57]. Besides choosing which algorithm to use, one needs to specify how the split shall be conducted (f.i. binary split or multi-way split) at each node (except leaf node) and how the best split is determined. Additionally, for specific tree-building algorithms, one can choose if the tree shall be fully grown or if it shall be pruned (i.e., the splitting procedure shall stop even though the leaf nodes are not pure).

The purity of the resulting nodes is evaluated concerning the classes they contain to determine the best split. Standard metrics used to measure the quality of a split are the classification error rate, the Gini index for measuring the Gini impurity, and Entropy for measuring the information gain [31]. Considering $p_{mk}$ as the share of record pairs of class $k$ (i.e. either match or non-match in ER) at a specific node $m$.

**Classification Error Rate** is simply the share of record pairs at node $m$ that are not from the most common class $k$ [31]. It is calculated by [31]:

$$E = 1 - \max_{\{k\}}(p_{mk})[31]. \tag{2.6}$$

In practice, however, the most common measures used are Gini index and Entropy as they appear to be better suited for tree-growing [31].

**Gini index** is calculated with the following formula [31]:

$$G = \sum_{k=1}^{K} p_{mk}(1 - p_{mk}). \tag{2.7}$$

It takes on small values if $p_{mk}$ is close to one for $k = match$ while $p_{mk}$ is close to zero for $k = non - match$ or vice versa. So in other words the Gini index is small as purer the resulting nodes are. This is also the case for the entropy measure [31].

**Entropy** is defined by [31]:

$$D = -\sum_{k=1}^{K} p_{mk} \log p_{mk}. \tag{2.8}$$

The problem with decision tree classifiers is that they tend to overfit too much to the data. That can easily be mitigated if the tree is pruned, which can be assessed with one of the above measures. Besides, a small change in the data can

[3] scikit-learn, which is the library used in this thesis uses an optimized version of CART [57]

have an enormous impact on the final tree that is constructed. Therefore it can be said that decision tree classifiers, as described above, are affected by high variance [31]. However, as mentioned before, the advantage is that a human can easily interpret them. Additionally, the feature importance can be calculated. Feature importance represents how much that feature reduces the impurity [31]. It is additionally weighted by the probability of reaching the node, which represents the feature [57].

### Ensemble Methods

Random Forests and XGBoost belong to the family of ensembles. The idea of ensembles is to use various weak learners to solve the same problem and combine their results based on a certain heuristic (e.g., majority vote in case of classification) into a single outcome. A weak learner does not necessarily refer to a bad model. In general, weak learners are models that would not perform very well by themselves because they could suffer from high bias or too much variance. The advantage is that if one learner is wrong, other learners may not, and hence the overall prediction will be better. By doing this ensemble methods tend to overfit less to the data and thus reduce variance [31, 69, 10]. The point is to have many different learners that do not make the same mistakes as the other ones, and therefore, together, they can perform quite well.

To accomplish this, Random Forest and XGBoost use two different concepts. One of them is called bagging [10], which in general refers to creating samples by randomly selecting examples from the data set with replacement (also called bootstrapping) in order to learn a model on each sample and aggregate the predictions of each model [31, 69]. The other concept is called boosting, which is a bit more complicated. It incorporates training a set of classifiers one after another, where subsequent classifiers focus more on examples misclassified by previous classifiers (e.g., through weighting). One type of learner that applies boosting is a Gradient Boosting Machine (also known as Gradient Boosting Tree). XGBoost is a particular implementation of a Gradient Boosting Machine using decision trees, optimized to be highly effective [12]. The interested reader is referred to the paper introducing XGBoost by Chen and Guestrin [12] or the work by Friedman [29] for a detailed explanation of XGBoost and Gradient Boosting Trees.

In Random Forests [11] bagging comes into play which is slightly adapted with additional constraints in order to improve simple bagging [11, 31]. A Random Forest, as the name implies, consists of several decision trees that are learned on different random samples of the data set, same as bagging [11, 69]. However, they additionally only analyze a random subset of features for each split instead of all features [11]. This approach ensures that each tree grown does not focus on the

same feature, which is considered the strongest predictor. Otherwise, it would lead to a strong correlation between the predictions of each tree within the forest and would not reduce variance [31]. Besides that, each tree is often left unpruned (i.e., there are no constraints set for how big a tree can grow). It is important to note that this is not necessarily the case as certain settings can easily be changed, and still, one would consider it as a Random Forest.

As Random Forests contain a collection of many trees, they are harder to interpret [31]. However, the feature importance can be calculated by taking the Gini index's decrease for a specific feature averaged over all bagged trees in the forest [31].

#### Scikit-learn: Machine Learning in Python

Scikit-learn is a Python package that provides many of the most advanced machine learning algorithms for supervised and unsupervised problems. The first public release of the scikit-learn library was in 2010. It is mainly written in Python, but some parts are written in Cython, C, and C++, due to performance reasons.

The focus is on usability, performance, documentation, and API consistency. Besides, there are few dependencies, and it is distributed under the simplified BSD license [57]. That has led to broad adoption in both the academic and commercial domains. It is open-source and maintained by a team of mainly scientists and developers.

The implementations of the learning algorithms Logistic Regression, Logistic Regression CV, Support Vector Machines, Decision Tree, and Random Forest, used in this thesis and explained in the previous subsubsections, are all from the scikit-learn machine learning library.

### 2.3.6 Evaluation

Regardless of which approach is chosen for ER, it is always necessary to evaluate how good the classification capabilities are since most of the approaches mentioned so far are approximate or based on heuristics. Therefore, it is not guaranteed how well they perform in a given domain. As already mentioned in subsection 2.3.5, a test set is needed on which the performance can be measured. In ER, this is also referred to as a so-called gold standard which contains all true matches that exist between two data sources [46]. This gold standard needs to be created by humans, or sometimes, if this is too labor-intensive, an exhaustive search algorithm is used to resolve the entities, and the result is then considered a gold standard [46]. Once a gold standard is at hand, the selected approach's performance can be measured by applying it to the domain and comparing the results to the gold standard. For that,

| | | Classification | |
|---|---|---|---|
| | | Match | Non-Match |
| Actual | Match | True Positives (TP) | False Negatives (FN) |
| | Non-Match | False Positives (FP) | True Negatives (TN) |

Table 2.1: Confusion Matrix for Binary Classification of ER

a so-called confusion matrix (see Table 2.1) can be used. The confusion matrix is a common practice for evaluating classification models as it provides a good overview of the errors the model is doing.

Specific metrics can be derived from the confusion matrix, which will be explained in the following. Those are the ones commonly used in ER and also used for the experiments in this thesis.

**Recall** (R) is considered as the amount of documents retrieved in order to answer a search query in information retrieval [73]. In the binary classification of ER for the class matches, the amount of classified matches correctly identified is divided by the amount of actual true matches in the record pairs' collection.

$$R = \frac{TP}{TP + FN} \tag{2.9}$$

However, this measure does not consider how many record pairs the system classified as matches that are no actual matches. For this precision is needed.

**Precision** (P), in the context of ER, is the share of actual matches divided by the amount of all record pairs the classifier classified as matches.

$$P = \frac{TP}{TP + FP} \tag{2.10}$$

**f1 score** (F1) or f-measure is the harmonic mean of Precision and Recall

$$F1 = \frac{2 * P * R}{P + R} \tag{2.11}$$

With the f1 score, a classifier gets a worse result if, e.g., R is 100% (i.e., FN = 0), but P is only 10% (i.e., a high number of FP; of all matches returned, only every tenth record pair is an actual match) as this results in an f1 score of about 18%. However, if the arithmetic mean is taken, a much higher score of 55% would be the result. Intuitively, the harmonic mean makes more sense for measuring a classifier's performance since a high R is often not desirable when P is significantly

worse. Nevertheless, for some specific use cases, one of the measures could be considered as more important.

However, if data records from different data sources are consolidated (e.g., in a new database) and a low P is achieved, the resulting information would not be trustworthy because descriptions from different entities are mixed. A similar problem occurs if R of the classification model learned is low as many record pairs that are referring to the same real-world entity stay undetected. This issue would mean that duplicates exist in the consolidated database, which leads to false statements when discussing the overall amount of real-world entities in the integrated data set.

Another critical point is that generally in ER, the non-matches are much more common [68]. For this reason, it would be misleading to consider TN for assessing the performance of an ER model. Examples of quality metrics that consider the TN rate are Accuracy, False Positive Rate, and True Negative Rate. Therefore, looking at the F1 for an ER classification model is reasonable and provides a good trade-off between the errors the model is doing.

A further metric that considers TN but still is not sensitive to the imbalance problem is the Matthews Correlation Coefficient (MCC) [7]. MCC is calculated by:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2.12}$$

The MCC result is in the range of $-1$ and $1$. A value of $1$ stands for a perfect prediction and $-1$ for an inverse prediction. A value of $0$ stands for a random prediction. Nonetheless, f1 score, precision, and recall are the metrics commonly used to evaluate ER, not MCC. However, MCC is used in this paper to measure the domain relatedness between source and target domain, which is used as a data profiling dimension. Due to the confusion matrix in Table 2.1 it is added to this subsection.

## 2.4 Profiling Data for ER Research

In ER research, the evaluation results based on the metrics mentioned in subsection 2.3.6 are sometimes insufficient to understand the performance of an ER method. Therefore, data used for the evaluation needs to be profiled to get a better understanding of the performance of the ER method. Standard profiling dimensions are characteristics like the size of the data sources or the amount of shared true matches. However, more advanced dimensions are needed to understand the difficulty of the matching task. In this thesis, the term candidate set is often used,

which is the collection of record pairs from two data sources that need to be resolved. The characteristics of the candidate set, therefore, refer to the matching task. In a very recent work by Primpeli et al. [61], five profiling dimensions for a matching task are proposed, which are explained in the following subsection. Four of them are used in this thesis.

The first one is *Schema Complexity (SC)*. It indicates how many attributes of the shared schema are required for successfully resolving entities in the candidate set. The authors argue that candidate sets with large SC are more difficult because more parameters need to be fit while learning the model, and the potential of overfitting rises [61]. To calculate SC, the authors propose a heuristic to identify relevant features and the corresponding attributes. Their heuristic suggests training a Random Forest Classifier using cross-validation with four folds on all features to retrieve the f1 score $f1_{all}$. The feature importance is then measured, and the features are returned in descending order based on the feature importance scores. After that, in a first iteration, the feature with the highest feature importance is selected. A Random Forest classifier with the same parameter setting and four-fold cross-validation is trained only on that feature. If the result with this single feature $f1_1$ is below $f1_{all}$, the feature with the second-highest feature importance score is included. Hence, in each iteration, the next highest feature is added until the f1 score achieved is the same or better than $f1_{all}$. The attributes associated with the features that are enough to achieve f1 score at least as good as $f1_{all}$ are considered *relevant attributes* [61]. Additionally, the authors suggest another heuristic to define *top relevant attributes*. They propose considering the attributes associated with the features that achieve $95\%$ of $f1_{all}$ when applying the approach mentioned above. However, for calculating SC, they suggest using the number of relevant attributes (i.e., not only top relevant attributes) [61]. Nevertheless, the top relevant attributes are highlighted in bold in the candidate set-level profiling tables.

Another profiling dimension is called *Sparsity*, which is the share of missing values in the whole candidate set. A high *Sparsity* of a matching task often signifies that it is more challenging for the learning algorithm [61]. Besides that, *Textuality* is proposed as a further profiling dimension that shall provide insight into the length of the attribute values contained in the candidate set [61]. The authors point out that the *Textuality* plays a crucial role in the design of the feature vectors and propose calculating it as the average length of the attribute values from the top relevant attributes [61].

The two last profiling dimensions are *Size* and *Corner Cases*. *Size* is the number of positive and negative record-pairs in the candidate sets [61]. *Corner Cases* refers to the number of hard positive cases and hard negative cases. Hard positive cases are true matches with a low overall similarity score, and hard negative cases are true non-matches with a high similarity score. The amount of corner cases is

an important indicator of the difficulty of the matching task. If a candidate set does not contain any corner cases, it would be straightforward for a learning algorithm to differentiate between the two classes. It only needs to find the right threshold to separate them [61].

The authors propose the following heuristic to calculate the share of corner cases in the candidate set. From the top relevant features, the non-missing values are averaged, and in steps of $0.01$, a linear threshold is searched, which maximizes the f1 score [61]. After the optimal threshold is encountered, the amount of matching record pairs below the threshold (i.e., false negatives (FN)), as well as the amount of non-matching pairs above the threshold (i.e., false positives (FP)), is measured [61]. The final ratio of corner cases (CC) is calculated by [61]:

$$CC = \frac{|FN| + |FP|}{|true\_matches|}. \tag{2.13}$$

# Chapter 3

# Active Learning

Machine learning (ML) refers to the field of computer systems that learn from data and improve with experience (supervised learning that was mentioned before is a specific type of ML). Active Learning (AL) is a concept within ML that aims to mitigate the need for many labeled instances from which the learning algorithm can learn. In traditional ML, when compared to AL, also called passive learning, a learning algorithm needs a sufficient amount of labeled data from where it can learn.

In the ER field, we can provide a learning algorithm with record pairs of two data sources labeled to a match (i.e., they refer to the same real-world entity) or a non-match. The learning algorithm can then learn how to discriminate between these two classes (as described in subsection 2.3.5). It was already mentioned that acquiring labeled data is labor-intensive and a tedious task. Therefore applying AL also to the problem of ER is gaining more and more momentum in recent years.

The following will give an overview of AL and its components before a summary of recent works that combine AL and ER is provided. Most of the information provided here is based on the AL survey of Burr Settles [6] if not cited differently.

## 3.1 Motivation and Challenges

AL is motivated by the hypothesis that if a learning algorithm can select instances where it wants to learn from by its own, it will require far less training data while performing at least as good as passive learning methods that are trained on a subset of data drawn at random from the whole population [6].

One challenge that arises within AL is how to initialize (i.e., also called bootstrap) the active learning process when not having any labeled instances at hand. The problem is also called *cold-start problem*. Another challenge is how to se-

lect the most valuable instances for labeling so that the labeling costs can be kept low while still achieving excellent performance. Of course, there are further challenges. For instance, when to stop querying instances (i.e., the stopping criteria) is not well researched yet in the literature. However, in this thesis, the focus is only on the first two challenges.

## 3.2 AL Settings

The process of selecting instances from the data to retrieve the label for them is the heart of AL. The overall setup can be classified into three different approaches that will be explained in the subsequent subsections.

### 3.2.1 Query synthesis

Query synthesis refers to the setting where the learning algorithm generates the instance that it wants to get labeled by itself. The learning algorithm only needs to know the dimensions of the features and the ranges for that. However, the problem here is that the learning algorithm may not know the underlying distribution, so it potentially queries instances that are not representative of the problem at hand [6].

### 3.2.2 Stream-based selective sampling

Stream-based selective sampling is based on selective sampling [4, 19], assuming that retrieving an instance from the underlying distribution is cost-effective. Therefore, the learning algorithm can simply request one instance at a time and decide if it wants to know the label or not [6]. The decision on whether to request the label will be based on the informativeness of the instance. The informativeness, in turn, is calculated based on a certain heuristic [6].

### 3.2.3 Pool-based sampling

Pool-based sampling [40] assumes that there is a pool containing all unlabeled data for which at each iteration, the informativeness is measured on all instances (not necessarily all) within the pool and the most informative is selected. This approach is not necessarily restricted to only one instance at a time, but also $n$ most informative instances can be queried at once [6]. This setting differs from the stream-based one as it considers the informativeness of all instances and not individually for one instance at a time. Therefore, for scenarios where devices with limited resources are used, the stream-based approach could be more suitable

[6]. Nevertheless, the most attractive setting in the active learning community is the pool-based one.

Different heuristics for measuring the informativeness exist, and in the literature, the name query strategy is often used to refer to them. Different query strategy approaches will be covered in the next section. However, the focus will be on the Pool-based sampling setting, as this is the setting used in the method proposed in this thesis. Besides that, the focus remains on the binary classification, which is the case for the whole thesis.

## 3.3 Query Strategies

In this section, popular query strategy approaches are explained. This overview is not complete, and the interested reader is referred to the survey from Burr Settles [6] to read about a few more approaches and a more in-depth explanation of the ones mentioned here. Essentially, the goal of a query strategy is to select the most informative instances and request the label for them, as mentioned before already. Because of this, the upper bound that a query strategy has to outperform is random sampling.

### 3.3.1 Uncertainty Sampling

For this type of query strategy, the assumption is that the learning algorithm can omit to query the instances it is already very confident about but focus more on the confusing ones [6]. The uncertainty in the binary classification case can be measured using the distance of the instance to the decision boundary. That means that simply the instance closest to the decision boundary can be queried. Since it is straight-forward and also fast, uncertainty sampling is one of the most popular approaches [6]. However, the problem is that the uncertainty at each iteration is only based on one single model (i.e., also referred to as hypothesis) created by a single learning algorithm, which additionally is learned from little data [6]. Other approaches exist that focus more on reducing the set of possible hypotheses (i.e., eliminating bad hypotheses) that will be learned in subsequent iterations. Hence, these kinds of approaches do not only consider the informativeness of a single instance alone but also try to gain information about the learning algorithms' hypothesis space [6]. One of them, called Query By Committee, will be explained in the next subsection. Other approaches can be found in the survey by Burr Settles [6].

### 3.3.2 Query By Committee

Query By Committee belongs to the class of disagreement-based query strategies and requires a method to retrieve hypotheses that will be part of the committee (i.e., committee members) and a method for measuring disagreement among them [6]. One example of retrieving the committee's hypothesis would be to use a single learning algorithm with different parameterization or to use different learning algorithms (i.e., heterogeneous committee) as members of the committee. Here, the literature suggests many different approaches to selecting committee members. For example, the use of an ensemble method would also fall under this category [6]. Still, Query By Committee differs from the Query By Disagreement strategy, which is another disagreement-based approach. The main difference is that Query By Committee does not consider all possible hypothesis but only a subset. Therefore, Query By Committee is applicable to the pool-based setting [6]. For the number of committee members to use, no clear rule exists [6]. However, in the context of AL for ER, a heterogeneous committee with four members has shown to perform well [13].

The second essential component for Query By Committee strategies is, as mentioned above, a heuristic to measure the disagreement among the committee members [6]. The two most popular measures are Vote Entropy and Kullback-Leibler Divergence [36].

Vote Entropy (VE) would select $x^*$ based on the following calculation [6]:

$$x^*_{VE} = \underset{x}{\operatorname{argmax}} - \sum_y \frac{vote_c(y,x)}{|C|} \log \frac{vote_c(y,x)}{|C|} \tag{3.1}$$

Here $y$ represents one class of $Y$ (i.e. $Y$ = {match,non-match}) and $vote_c(y,x)$ represents how many committee members predict the class $y$ for the record-pair $x$. $|C|$ stands for the amount of committee members [6]. Another option is to calculated a slightly different version of VE, called Soft Vote Entropy (SVE), that considers the confidence of each committee member [6]:

$$x^*_{SVE} = \underset{x}{\operatorname{argmax}} - \sum_y P_c(y|x) \log P_c(y|x) \tag{3.2}$$

Measuring disagreement based on the Kullback-Leibler Divergence (KL) does follow a different approach. KL is a way to measure the difference between two probability distributions [36, 6]. The disagreement among the committee members is measured by averaging KL between each members prediction $\hat{y}$ and the consensus $C$ [6]:

$$x^*_{KL} = \underset{x}{\operatorname{argmax}}(\frac{1}{|C|} \sum_{\hat{y} \in C} KL(P_{\hat{y}}(Y|x)||P_C(Y|x))) \tag{3.3}$$

KL is defined by [6]:

$$KL(P_{\hat{y}}(Y|x)||P_C(Y|x)) = \sum_y P_{\hat{y}}(y|x) \log \frac{P_{\hat{y}}(y|x)}{P_C(y|x)} \tag{3.4}$$

## 3.4 Related Work

This section provides an overview of existing work from the literature that applies AL to the ER problem. The approaches that will be explained in the following subsections all follow a Query By Committee approach in the Pool-based setting. Because of this, they are similar to the approach proposed in this thesis with the work by Primpeli et al. [62], explained in subsection 3.4.3, being the most similar one as the method proposed in this thesis uses the same setting, but TL instead of unsupervised matching. Therefore, this method [62] is also used as the baseline for the final evaluation. Besides, their unsupervised matching serves as a baseline for one of the initial experiments.

Nevertheless, there exists more work in the field of ER, which uses AL. However, they follow a different approach or a slightly different setting. For instance, some use AL in a committee-based setting but in combination with genetic programming and apply these methods to ER [23, 30, 53, 54, 60].

In other works [3, 8, 17, 27, 63, 75] the authors propose solutions using AL that are focusing on learning matching rules that guarantee high precision. These approaches are rather different from the approach considered in this thesis. The same applies to the work of Ngomo et al. [51]. However, this approach is also different from other approaches mentioned so far as it focuses on the link discovery across knowledge bases with large and diverse schemata.

### 3.4.1 Two Pioneer Works on AL in ER

The papers presented in this subsection do follow not only a similar AL setting as the method proposed in this thesis but also are the first two works using AL to tackle ER. They were published in the early 2000s by Tejada et al. [71] and Sarawagi and Bhamidipaty [68]. Both approaches are similar in that they both apply a pool-based setting using a Query By Committee strategy consisting of decision tree learners.

To be more precise, the former applies a query by bagging approach (bagging is explained in section 2.3.5), which they bootstrap with examples picked based on their aggregated similarity scores [71]. For each attribute value-pair, they first apply different transformation functions and calculate the similarity score based on a customized TF/IDF weight combined with cosine similarity [71]. After that,

they sum the similarity scores to an overall score by weighting each score with the uniqueness of the corresponding attribute, which is a heuristic of how important the attribute is [71]. As a final step to select the initial examples, they group the instances based on the aggregated similarity score into as many groups as attributes. They assume that the group with the highest total similarity scores is the smallest and contains mostly positive examples (i.e., first group) [71]. From all groups, they randomly sample an equal amount of instances [71]. With those examples, they bootstrap their committee after a human annotator provides the correct labels for them [71]. They choose the examples to be queried based on the votes of the bagged trees. However, they do not provide a number of how many trees to use. Even though not explicitly mentioned in their paper, they measure disagreement among the votes using vote entropy. Besides that, they select the one with the highest total similarity score if more than one example achieves the largest vote entropy. They assume to pick more of the rare positive examples by doing this [71].

In the work of Sarawagi and Bhamidipaty [68], they do not use bagging in order to build the committee. Instead, they tested different approaches in order to create a diverse committee. According to their experiments, the best performing one introduces randomization by not picking the attribute with the highest information gain at each node but randomly choosing an attribute with an information gain close to the best one [68]. Besides that, they select the threshold, at which to split a continuous attribute at a node, uniformly and at random within the range of the attribute values [68]. In an experiment, they have shown that the committee's size can be kept below five members without hurting performance much [68]. In order to measure the uncertainty for each example, they use vote entropy [68].

These two pioneer works about AL in ER suffer from one major problem. Sarawagi and Bhamidipaty [68] assume to have a small number of labeled instances (i.e., around ten) of matches and non-matches in order to bootstrap the labeled set [68]. However, this is not realistic because, as they point out themselves, when justifying why they use f1 score as a measure, ER is very much skewed towards the class of non-matches [68]. Therefore, a random selection of matches is not possible while keeping the cost of labeling low.

In the work of Tejada et al. [71], they consider the problem of additional labeling costs, but they do not ensure that the amount of matches and non-matches is balanced; moreover, they do not even describe an approach that ensures that both classes are present in the bootstrapping sample at all. As Chen et al. [13] show in their recent study, this problem results in too little matching record-pairs within the bootstrap sample and, therefore, often leads to bad performance [13]. The experiments of Chen et al. [13] also indicate non-satisfying overall results for both approaches described above. For the method of Tejada et al. [71], they suggest that

the approach of always picking the one with the highest total similarity score for instances that share the largest vote entropy within the query strategy is potentially problematic [13]. However, it is essential to mention here that Chen et al. [13] did modify the originally proposed approach of Sarawagi and Bhamidipaty [68] by using SVMs instead of decision trees due to implementation issues [13]. Because of this, they also changed the committee members of the method of Tejada et al. [71] from decision trees to SVMs and justified this with the comparability of the results [13]. Nevertheless, the problem with the bootstrapped sample remains as the committee members do not influence this.

### 3.4.2 HeALER

The approach from Chen et al. [13], which they call HeALER, tries to tackle the problems identified in subsection 3.4.1. To retrieve a balanced bootstrapping sample, they also aggregate the similarity scores of attributes for each record pair, but they do not use any weighting for that but simply sum up all scores [13]. After sorting all record pairs based on the total score in descending order, they bin them into $k$ groups with equal width (f.i. if the amount of similarity scores is ten then the values range from zero to ten and could be classified in ten bins with equal width) [13]. These $k$ groups are then categorized further into three different zones, which shall represent the likeliness of containing matches or non-matches. They call these zones matching zone, mixed zone, and non-matching zone [13]. The matching zone represents the record pairs within the groups with the highest overall similarity score, followed by the mixed zone in which both classes could be represented. The non-matching zone should be the largest, containing far more non-matches than matches, if any. For defining the size of the matching and mixed zone, they point out that there is no exact method. However, one heuristic would be (in the case of matching entities across two data sources) to choose the smallest data source size because this represents the maximum of possible matches [13]. To distinguish between the matching zone and the mixed zone, they merely point out that the matching zone has the largest total similarity scores [13]. They also assume that by randomly selecting from the matching zone, mostly matches can be retrieved. By randomly selecting from the mixed zone, the same amount of informative non-matching pairs can be fetched. They consider the size of the bootstrapping sample as a preset parameter $n$, and in order to retrieve a more or less balanced sample, they pick $n/2$ from the matching zone and $n/2$ record pairs from the mixed zone [13].

Apart from their proposed approach to creating a balanced bootstrapping sample, they suggest using a heterogeneous committee, taking into account the efficiency and interpretability of the learning algorithms used [13]. With their ex-

periments, they show that this outperforms committees based on a single learning algorithm [13].

### 3.4.3 Unsupervised Bootstrapping of AL for ER

An even more recent work that utilizes AL for ER in the pool-based setting with a Query By Committee strategy is that of Primpeli et al. [62]. Here, the authors use a newly proposed unsupervised matching approach to bootstrap the labeled set on which the committee can be trained on right from the beginning and incorporate knowledge into the active learning model and query strategy [62].

For the proposed unsupervised matching approach, they aggregate the similarity scores for every record pair. However, each similarity score is weighted by the density of the corresponding feature (a feature is a certain similarity measure applied on the value pairs of a certain attribute) [62]. As they represent missing values (i.e., at least one attribute value from the two data sources is missing) by a similarity score of $-1$, this refers to the share of values other than $-1$ for that particular feature. After that, the cumulative distribution of the total similarity scores is used to calculate the so-called elbow point [62].

According to Primpeli et al. [62], the value of the elbow point can be estimated as the point on the cumulative distribution of similarity scores with the maximum vertical distance to the line connecting the first and the last point of the distribution. All record pairs with an aggregated similarity score above the elbow point value are considered a match and vice versa [62]. They use this unsupervised approach to retrieve a label (i.e., a noisy label) for each record pair and assign confidence weights based on the normalized distance of the total similarity score to the elbow threshold [62]. Record pairs with a high or very low overall similarity value are weighted more heavily since they are less likely to be incorrectly classified by the unsupervised matching method.

The record pairs with the noisy label are used to train an initial Random Forest of ten estimators, using the confidence weights as sample weights for the training [62]. They expand this initial Random Forest model at each iteration by adding two more trees trained on the labeled set of the current iteration [62]. To bootstrap the committee, they take the most confident positive and negative record pair and incorporate the noisy labels from the unsupervised matching as an additional vote into the Query By Committee strategy [62]. Their committee consists of five models, namely Logistic Regression, Linear SVM, Decision Tree, XGBoost, and Random Forest [62].

### 3.4.4 Libact: Pool-based Active Learning in Python

Libact is an open-source Python package designed to facilitate pool-based active learning for developers or scientists. The package implements many popular active learning strategies. Besides, the package provides a standardized interface for implementing additional active learning strategies. It is also easy to include new active learning models and customized labelers [77]. The ability to be easily adaptable makes it very valuable for the experimental framework of this work.

# Chapter 4

# Transfer Learning

Another concept within ML which tries to circumvent the need for acquiring a vast amount of labeled data exist. The idea is to use existing (labeled) data from a related source to improve the learning process for the problem at hand, which contains no or little labeled data. In other words, it is a transfer of knowledge from one domain (the source) to another (the target), and therefore this concept is generally called transfer learning (TL).

In the following, first, some notations are provided in order to explain TL and to be able to introduce the different categories. After that, the focus is set on the category of TL, which is used in this thesis.

## 4.1 Notation and Basic Concept

The following definitions are taken from the recent TL survey by Zhuang et al. [80], which in turn are from the first comprehensive TL survey by Pan and Yang [55], but have been slightly modified to cover the progress made in this research area in the meantime.

When talking about transferring knowledge from one domain to another in order to solve a task, domain and task needs to be further defined. A domain consists of a feature space $\mathcal{X}$ and a marginal distribution $P(X)$ with $X$ denoting an instance set containing $n$ instances, that is defined as $X = \{x_i \in X, i \in 1, 2, ..., n\}$ [55, 80]. In ER, the feature space corresponds to the similarity measures applied to certain value pairs (also known as comparison vector or simply feature vector) and an instance to a record pair. A task $T$, on the other hand, consists of a label space $Y$ and a decision function $f$, that matches an instance $x \in X$ to a label $y \in Y$ (i.e. $f(x) = y$). From a probabilistic point of view, this can also be considered $P(y|x)$, which is the probability of $x$ belonging to class $y$.

In the ER context, the task is the previously mentioned problem of classifying a record pair into a match or a non-match ($Y \in \{0\,;1\}$ with 0 = non-match and 1 = match). The decision function $f$ represents the classification model learned from the training data. In traditional ML, the source and target domains (i.e., $D_S$ and $D_T$ with subscript $S$ denoting source and $T$ target), as well as the source and target tasks (i.e., $T_S$ and $T_T$, respectively), are considered to be the same.

So when performing traditional ML, one generally assumes that the training data comes from the same domain as the test data. In this case, one can view the training data as the source and test data as the target domain. Therefore, one can say $\mathcal{X}_S = \mathcal{X}_T$ and $P_S(X) = P_T(X)$. Besides that, the task that needs to be resolved is the same across training and test data. One uses the training data to learn the classification model or decision function, which maps an instance to a label from the label space. This means $T_S = T_T$, with $Y_S = Y_T$. However, in TL, some of the assumptions do not hold, and the formal definition of TL is:

**Definition 4.1.1** *Transfer Learning: Given $D_S$ and $T_S$ as well as $D_T$ and $T_T$, transfer learning is about using knowledge from $D_S$ and $T_S$ to improve the learning of the decision function $f_T$ for $T_T$ in $D_T$, where $D_S \neq D_S$* ***or*** *$T_S \neq T_T$ [55].*

## 4.2 TL Settings

From the feature or label space point of view TL can be classified into *homogeneous TL* if $\mathcal{X}_S = \mathcal{X}_T$ and $Y_S = Y_T$. If at least one of the two conditions does not hold, it is considered as *heterogeneous TL* [80]. In this thesis, the same similarity measures are used, and the schemata are aligned across both domains. Therefore, the feature space $\mathcal{X}$ is the same across $D_S$ and $D_T$. Besides that, in both domains, the label for the record pairs is either match or non-match (i.e., $Y_S = Y_T$). Because of this, the focus will only be on *homogeneous TL* in this thesis. More information about *heterogeneous TL* can be found in the survey by Day and Khoshgoftaar [22] or Weiss et al. [76].

Another possible categorization of different TL settings can be done from the label-setting viewpoint [55, 80]. If label information from $D_T$ is available it is regarded as *inductive TL*. Here it is also assumed that $D_S$ and $D_T$ are the same and only $T_S$ and $T_T$ are different but related [55]. If label information is only available from $D_S$ it is called *transductive TL* in the literature. Besides that, it is additionally assumed that $D_S$ and $D_T$ are only related with each other and $T_S$ and $T_T$ are the same for *transductive TL* [55]. For the case, that label information is unknown in $D_S$ as well as in $D_T$ and the domains, as well as the learning tasks, are only related to each other the term *unsupervised TL* is used [55].

The TL setting applied in this thesis belongs to *transductive TL* as labeled data is available from $D_S$, but before the active learning approach starts, no labeled data is available from $D_T$. From the feature or label space perspective, the setting in this thesis refers to *homogeneous TL* as feature and label space are the same across both domains. However, as the record pairs differ across $D_S$ and $D_T$ the marginal distributions $P_S(X)$ and $P_T(X)$ are different. According to Pan and Yang [55] this specific setting is also associated in the literature with the covariate shift [70], the sample selection bias [78] or the domain adaptation [21].

However, one often encounters minor differences when reading about the term domain adaptation in the literature. That is the case in a quite recent study about domain adaptation from Kouw and Loog [35]. Here the authors consider domain adaptation as a branch of TL where differences in the feature space or label space are possible. They state that domain adaptation describes the process of using labeled source domain data to make predictions in the target domain where only unlabeled data is available [35]. In the work of Daumé and Marcu [21] the authors state that the term domain adaptation is also known as "transfer" [21, p.1]. In the survey about TL from Zhuang et al. [80], the term is simply used when talking about the process of adapting a domain to the other one. Weiss et al. explicitly mention in their survey [76] the confusion about the term domain adaptation in the literature and state that they use the term domain adaptation to describe the process of adapting the source domain to the target domain in order to perform TL [76]. That is also the terminology used for the term domain adaptation in this thesis.

Additionally, Weiss et al. [76] point out that there is also confusion regarding the classification of TL based on the label-setting point of view in the literature. Here minor differences to the categorization from Pan and Yang [55], as mentioned above, can be found in other works [76]. Therefore, it is always important to state from which domain labeled data is used.

To highlight the setting used in this thesis: For the transfer of knowledge, only labeled data from the source domain is considered because TL is used before the start of the active learning process. Then, labeled data from the target domain is acquired. Besides that, the use of domain adaptation techniques without considering the labels across both domains is researched. For this, the term unsupervised domain adaptation will be used, and in subsection 4.3.1, the focus will be mainly on this particular setting.

## 4.3 Related Work

Although TL is a vibrant research area, the scientific community has not yet proposed much work that applies TL to ER. Before related work, which applies TL to

ER, is introduced, three different approaches for TL stemming from the literature are explained. These approaches to TL are used in this thesis.

### 4.3.1 Naive Transfer and Domain Adaptation Techniques

A straightforward approach for TL is to train a learning algorithm only on the labeled source domain data without any adaptation to apply it to the unlabeled target data. This approach was called *naive transfer* by Thirumuruganathan et al. [72]. The same term is used in this thesis when referring to this particular approach.

A significant concern of TL is the problem of potential negative transfer [76], which occurs when the source domain data negatively affects the performance of the classification model on the target domain because of the domains not being sufficiently related [66, 76]. However, Weiss et al. [76] point out that the problem of negative transfer is not yet well researched. For instance, it is hard to find a good measure that precisely accounts for the relatedness of two domains [76]. That is amplified by the fact that differences across two domains can have various causes where some of them simply cannot be measured. For instance, if only labeled data is available from the source domain differences in the conditional distributions or potential class imbalances cannot be known. This issue will also be problematic if little labeled data is available in the target domain since the underlying probability distributions are generally unknown. However, besides differences in the marginal distributions across the source and target domains, these differences are potential causes for a negative transfer [76].

In this thesis, certain unsupervised domain adaptation techniques are tested to mitigate the risk of negative transfer. Those techniques are about weighting the source instances based on the relevance to the target domain and are explained in the following. Note that more domain adaptation techniques exist in the literature that can be applied to this thesis's problem setting. For more information about them and also TL approaches that are not suitable for the problem setting at hand, the interested reader is referred to existing TL surveys [22, 35, 55, 76, 80].

Unsupervised domain adaptation techniques that rely on the weighting of source domain instances assume that the conditional probability distributions in the source and target domains are equivalent [35]. Therefore, a further assumption is that when considering the marginal distributions of both domains, source examples that are more likely to come from the target distribution are considered more valuable. When training a learning algorithm using sample weights for the training instances, higher weights increase and smaller weights decrease the loss. That ensures that the learning algorithm adapts more to the instances with higher weights. A higher weight should, therefore, be given to the more important source samples [35].

Different approaches exist for estimating the weights of the source samples.

So-called indirect weight estimators estimate the marginal distributions of both domains and use the ratio of them as weights (i.e., $weight_x = \frac{p_T(x)}{p_S(x)}$ with x being a source instance) [35]. Another group does estimate the weights directly by considering it as an optimization problem. Here discrepancy measures are used, and the weights of the source samples are optimized so that the discrepancy between source and target domain is minimized [35]. Different discrepancy measures are used in different techniques, but according to Kouw and Loog [35], they adhere all to the constraints that all weights are greater or equal to zero and have a mean of one [35]. More information on these two groups can be found in their survey [35].

Additionally, there is a technique that does not first estimate weights for the source samples, which can be used as sample weights for the final classification model but does optimize the weights of the final classifier directly [9]. The weight estimation techniques considered in this thesis are direct weight estimators, but unlike those mentioned above, they are not based on optimization. They are explained in the following.

#### Weighting based on Logistic Regression

This method proposes to train a Logistic Regression model that can discriminate between source and target instances and use the predicted probabilities from the source samples belonging to the target domain as the weights [35]. In this thesis, the method is implemented in the following way. Source and target domain data are combined, and for each instance, its provenance is used as the target label for the logistic regression model that is trained on the combined data.

#### Nearest-neighbour-based Weighting

This approach was proposed by Marco Loog [44]. It is based on the allocation of the feature space in Voronoi cells. Each source sample's weights are obtained by counting the target samples in the Voronoi cell of that source sample. For this method, no hyperparameter tuning is required, and counting of the target samples can be done using a nearest-neighbor algorithm [35]. However, Laplace-Smoothing, which adds plus one to each Voronoi cell, can be used additionally [35].

### 4.3.2 TRANSFER - TL for multiple-source ER

Early work that applies TL to ER is the one from Negahban et al. [49], which considers the multiple-source ER problem setting with aligned schemata across the sources. Multiple-source ER refers to resolving entities across many sources of the

same semantic topic. The authors claim that, for achieving similar performance results across all source combinations, labeling costs would increase quadratically with the number of sources [49]. They argue that common approaches to ER with supervised learning build a model for each pairwise combination of data sources and would treat each learning task separately. However, some combinations share a common source [49]. Therefore, they propose a new method, which they call TRANSFER. This method jointly learns a linear classification model on all source samples in which common patterns that are shared across all source combinations are included in the learned weights, along with characteristics for each specific pairwise combination. They have shown that their method achieves superior results than three baselines with regards to precision and recall and the amount of labeled data required [49]. Their baselines are learning a linear model for each pairwise source combination separately, combining all source samples to learn a single model for all combinations without any shared component in the coefficients, and a non-linear baseline which also learns for all source samples at once. For the non-linear baseline, the instances' source origins are encoded into the feature vector, so that it can also learn both general patterns and specific patterns for each source combination [49]. However, they only test their method on one semantic topic as well as synthetic data. Besides that, their setting requires labeled data for each source combination. Therefore the problem differs from the one considered in this work. In this thesis, it is assumed that labeled data from a different source combination within the same semantic topic are available, but no labeled data in the source combination in which the entities are to be resolved.

### 4.3.3 TL for Link Discovery

Another work by Ngomo et al. [52] provides a formal definition of how TL can be conducted in the setting of link discovery across knowledge bases, which is a special use case within ER. In this work, the authors decompose the problem of transferring matching rules to three simpler problems. First of all, they state that the transferability of a matching task depends on how similar the source and target domains are, whether the feature space of the source can be mapped to the feature space of the target, and how accurate the source matching task is. Based on these points, they suggest using an existing matching task that comes from the most similar domain and feature space and has good results in its domain as background knowledge for the target domain. To measure this, they propose a heuristic for each of the three problems and an equation on how to combine them to calculate a total score. This total score indicates how suitable an existing matching task is for the matching task in the target [52]. The authors do not provide any experimental validation of their work as it is only a formal definition of how TL could be applied

to link discovery. Besides that, the problem setting of link discovery differs from the setting considered in this thesis.

### 4.3.4 Reuse and Adaptation for ER through TL

Another work that is more recent than the ones mentioned so far is the one from Thirumuruganathan et al. [72]. In this work, the authors identify four different real-world scenarios for ER that could arise when applying TL. The scenarios are based on the amount of labeled data available in the source and the target domain. The first scenario considers that sufficient labeled data is available from the source domain, but no labeled data from the target domain. They call this scenario (Adequate, Nothing) with adequate referring to a sufficient amount of labeled source data. With sufficient, they mean that an ER classifier can be learned with this amount of labeled data [72]. The other three scenarios are (Adequate, Limited), (Limited, Limited), and (Adequate, Adequate) [72]. For all these scenarios, they provide a specific approach for applying TL. Besides that, they propose an approach that aligns the record pairs from different domains to the same feature space to deal with heterogeneous schemata across data sources. They recommend not considering each attribute separately but combining them to one single sentence and applying word embeddings on this sentence to convert each word into a high dimensional vector [72]. For a record pair, they calculate the difference between the two vectors as a similarity measure. They also describe some practical issues when applying TL for ER and conduct experiments to verify their points.

As the scenario (Adequate, Nothing) is exactly the one considered in this thesis (i.e., before the start of the AL process), more details for how this scenario is tackled in the work of Thirumuruganathan et al. [72] is provided. For this scenario, they define two different approaches, which were already explained in subsection 4.3.1. The first one is the naive transfer (see subsection 4.3.1), and the second one is applying unsupervised domain adaptation to retrieve importance weights for the source domain data by using a similar approach as the one explained in section 4.3.1. Here the similarity of the two approaches is the use of a Logistic Regression model to differentiate across source and target domain. However, they propose a formula of how to calculate the importance weights for the source domain data. Instead of using the predicted probability of a source instance coming from the target domain as weights, they advocate calculating the weight $w$ for a source instance $x_i$ in the following way [72]. Assuming class one is representing the source domain as the origin.

$$w(x_i) = (\frac{1}{p(y = 1|x_i)}) - 1 \tag{4.1}$$

They state that if the probability for $x_i$ coming from the source domain (i.e., class one) is high, then a weight close to zero should be assigned and if the classifier is not sure from where $x_i$ is coming from (i.e., $p(y = 1|x_i) \approx 0.5$) a higher weight close to 1 should be assigned [72].

Even though the intention behind this weighting formula seems to be appropriate, there is one consideration missing. If much more unlabeled data is available from the target domain than from the source domain, but still all unlabeled data from both domains are combined to train a Logistic Regression model that discriminates between both domains, the learned model might tend to predict the majority class (i.e., the target domain) all the time.

This case would mean the predicted probabilities of class one would be close to zero, and therefore large weights (i.e., significantly higher than one) would be the result. This issue could potentially have a negative effect, and therefore it should be considered when applying this weighting scheme. In their paper [72], the authors mention dataset imbalance (i.e., differences in the number of instances across the source and target) as an issue when applying TL for ER. However, they only consider the differences in the amount of labeled data between source and target and not the issue with the importance-weighting that considers only the unlabeled data. In the setting considered in this thesis, using AL, a few labeled instances from the target domain can be acquired. Therefore, weights of the source domain data higher than one could mitigate the influence of the little labeled target domain data even more. It is possible to simply map the weights to values in the range of zero and one. However, using the predicted probability of the target domain class already produces values in the range of zero and one which can be assigned as weights to the source domain data. Thus, the approach proposed by Thirumuruganathan et al. [72] is not used in this thesis. As they report improved performance of their weighting scheme compared to naive transfer based on experiments with data sets from different semantic topics, unsupervised domain adaptation seems appropriate for this scenario [72].

### 4.3.5 Further Work applying TL to ER

There is more work that applies TL to solve ER like the work by Kasai et al. [32] or Zhao and He [79]. However, these works use deep learning [39], which is out of scope for this thesis. However, it is important to mention that Kasai et al. [32] even combines AL and TL for ER. Still, it is the only work in the ER research field that combines these two concepts to tackle ER, and to the best of my knowledge, there exists no work in the literature that combines these two concepts without using deep learning.

# Chapter 5

# Methodology

In this thesis, ER is considered a binary classification problem. The goal is to learn whether two records from two different data sources on the same semantic topic refer to the same real-world entity. To learn the classification model, as little labeled data as possible should be used for training. It is also assumed that within a semantic topic, at least three different data sources with aligned schemata, ds1, ds2, and ds3, exist. For two of them, the entities are already resolved (e.g., ds1 and ds2). Therefore, positively and negatively labeled record pairs are available for these two data sources (referred to as source domain). The goal is to resolve the entities between ds1 and ds3 and ds2 and ds3 (referred to as the target domains). In the target domain, it is assumed that no labeled data about matches and non-matches are available. However, using AL, few labeled data can be acquired.

Additionally, it is desired to boost AL using the knowledge in the form of labels from the source domain so that less unlabeled data from the target domain needs to be labeled by a human annotator. That combination of TL and AL should achieve comparable performance results to an ER model trained on more labeled target data. Before conceptualizing this method, two initial experiments are conducted, which shall provide insights on how to combine AL and TL.

In this chapter, the steps conducted in this thesis are explained in detail. It includes the process of filtering potential correspondences, creating feature vectors (i.e., similarity measures applied on the attribute value-pairs of each record pair), as well as creating training and test sets that are needed for training and testing the learned models.

Besides that, the experimental setup of the initial two experiments is described in detail. The whole framework for the experiments was developed in Python [67]. It is made publicly available on GitHub[1] along with the data used. Therefore,

[1] https://github.com/JayKay0104/ma-atl-for-er

the exact implementation of the experimental framework can be found online, and everyone interested can rerun the experiments.

After that, the methodology of the proposed method that combines AL and TL for ER is explained, and the baselines used to evaluate it are listed. In the last section, the data profiling dimensions to profile the data used for the experiments is provided.

## 5.1 Filtering and Creation of Candidate Sets

For the semantic topics, books and kitchen product offers only the data sources accompanied by a unique identifier were available. Therefore, for these topics, filtering had to be applied for all data source combinations to filter out apparent non-matches. Otherwise, too many record pairs would have made it impossible to work with due to immense computational complexity. Filtering or blocking is an essential first step in ER and, therefore, a quite active research field, as explained in section 2.2.

In this thesis, however, the intention to filter the data was somewhat differently motivated than is usually the case in ER. Typically, filtering or blocking is applied in ER to minimize a large number of record pairs so that fewer pairs that are more likely to relate to the same real-world entity need to be resolved. Here, one must consider that a filtering strategy could filter out real matches if it is too strict. Though, for the experiments conducted in this thesis, both training data and test data have to be created for a learning algorithm, as explained in subsection 2.3.5.

In addition to the true matches (i.e., those already resolved due to the unique identifier), this should contain pairs that do not refer to the same real-world entity, but for which it is difficult to decide whether they do. These cases will be referred to as hard negative cases in the following. Hard negative cases are just as important as hard positive ones (i.e., pairs that refer to the same entity but have different values for different reasons or are only slightly similar) to create a good training set. Such examples contain more interesting information than regular instances. Therefore, they make the learned model more robust and better prepared for unseen data. Together such instances are also called corner cases. However, a training set should not consist exclusively of such cases as the data used for training shall also represent the characteristics encountered in practice [37].

According to these considerations, for this thesis, it is not needed to apply the best state-of-the-art filtering or blocking strategy, which ensures to filter out as many non-matching record pairs as possible while filtering out as less true matches as possible. Because of this, in the experimental setup explained in this chapter, simple string similarity joins using Jaccard (explained in section 2.3.1) as similar-

ity measure are used, which are applied on a domain-specific attribute. For that, the Python library py_stringsimjoin, which is part of the Magellan Entity Matching System [34], is used. In the appendix section A.1, the exact settings for the similarity joins are listed.

#### Heuristic for Creating Candidate Sets

After the number of record pairs is heavily reduced due to filtering, a sampling heuristic is applied to create even smaller candidate sets (i.e., a collection of record pairs that includes true matches and hard negative record pairs). These candidate sets serve as the basis for the training and test sets used in the experiments. For that, the following heuristic is used. At first, similarity scores for each attribute value-pair of the matching and the non-matching record pairs are defined. The similarity measures used are the same for the feature vectors and are explained in section 5.2 in more detail. After the similarity scores are calculated, they are aggregated to one score for each pair of data sets.

Theses scores are binned into ten bins of equal width, and for each bin, it is tried to randomly sample as many negative record pairs as matches so that at best, each bin contains as many positive as negative cases. If there are not enough negative pairs for a particular bin, they are randomly sampled from the entire data set, regardless of which bin they belong. That is actually to be expected for bins with high similarity values since hard negative cases occur less frequently than easy positive cases. This heuristic not only creates candidate sets with as many negative as positive pairs but also ensures that hard negative cases are included. As mentioned before, hard negative cases are essential to learning a robust machine learning model [37].

This thesis aims to research if incorporating labeled training data from a related domain can boost active learning in another domain. Thus, it is considered necessary that the classes' distribution does not differ significantly across the candidate sets of one semantic topic. The different characteristics that can be found in each candidate set are investigated in the data profiling section, and there influence on the naive transfer is part of the examination. For the topic authors, existing candidate sets were available as this topic was used in a previous work [62]. Consequently, the whole process could be skipped for this semantic topic.

## 5.2 Creation of Feature Vectors

The problem setting considered in this thesis assumes that the schemata are aligned across the data sources of each topic. If one source does not provide a particular

attribute, it will be added to the schemata even though it is only filled with missing values. That is only the case for one data source, namely bookcrossing from the topic books, where the attribute *pages* is not provided at all.

The feature vectors are build by calculating similarity or distance measures on the value-pairs for each attribute of the record pairs. The approach used for creating the feature vectors is similar to the one applied in [62]. The data types contained in the data are strings, dates, numbers, and binary values (i.e., it only provides exactly two unique values). For strings, it is additionally differentiated between long strings and short strings. If an attribute column contains more than six words on average, long string is assigned if not short string. If at least one data source contains long strings for an attribute, all other sources' values are considered long strings as well. For all other differences in the data types for an attribute, all attribute values are automatically regarded as strings to ensure comparability. Similar to [62] for every data type, a set of similarity measures are applied that are listed together with the preprocessing steps in the following. Data type-specific similarity measures are applied instead of topic-specific ones to enable comparability of the results across the different semantic topics. Even though in practice, this can lead to worse results than topic-specific feature engineering.

All similarity scores are rescaled to be in the range of zero and one, with values close to one indicating highest similarity according to the similarity measure's criteria and vice versa. Because of this, distance measures are additionally reversed so that they align with this logic. If at least one value from the value-pair is missing, the similarity score is set to $-1$ no matter which data type the attribute contains. By doing this, there is no need to drop rows that contain missing values.

### 5.2.1 Long and Short Strings

First of all, before applying any similarity measure on the pairs of string values, the values are preprocessed in a standardized way. That means everything other than letters, digits, or white spaces is removed from the strings. Besides, the strings are converted to lower case, and any spaces to the left or the right are deleted. For long strings, English stop words are also removed, and stemming based on the Porter Stemmer [59] is applied. The following similarity measures are used:

- **Levenshtein** as explained in section 2.3.1.
- **Jaccard using trigrams with padding** (explained in section 2.3.1).
- **Jaccard with white space tokenization**.
- **Relaxed Jaccard with inner Levenshtein** as explained in section 2.3.1. A Levenshtein threshold of $0.7$ is used.

- **Containment** and **Exact Similarity** as explained in section 2.3.1
- **Cosine Similarity with TF/IDF Weighting** as explained in section 2.3.1. This similarity measure is only applied to the long string attributes.

### 5.2.2 Date

The Python Programming Language [67] does not automatically recognize dates. If the type for a specific attribute is considered a string and not a long string, potentially dates can be encoded in the strings. Therefore, it will be checked if the strings represent a date based on a list of conventional format strings used for dates (i.e., not complete but 23 standard date formats are recognized). If this is the case, the data type of the attribute is considered as date and will be converted to a Python Datetime Object [1] so that the pairs of a date attribute can be compared easily. For the date object, three distance measures are applied.

- Absolute difference in **days**
- Absolute difference in **months**
- Absolute difference in **years**

If a date only contains the year, the date will be set to the 1st of January of that year; the same logic applies if only month and year are provided to be still able to calculate the difference in days.

### 5.2.3 Number

For numbers, only the absolute difference is calculated, rescaled, and reversed into a similarity score, as mentioned above.

### 5.2.4 Binary

For binary attributes, all data sources can only contain exactly two unique values. If this is the case, only the exact similarity is measured (explained in section 2.3.1). Only the topic books with the attribute *binding* has this data type, which additionally contains a lot of missing values.

## 5.3 Creation of Training and Test Data

To perform the experiments, not only training data for the supervised learning algorithm is needed but also test data on which the learned model can be evaluated.

As mentioned in section 2.3.5, different approaches like cross-validation or using a dedicated hold-out test set can be used.

However, when incorporating labeled source data in the training phase but testing the model's performance on the target domain to compare it to the baselines, only the approach of using a hold-out test set from the target domain makes sense. Because of this, for the experiments, the candidate sets were split into training and test set. For that, a random split with $77\%$ for training and $33\%$ of the data for testing is performed. The split is conducted in a stratified fashion, which ensures that the class distribution is the same across training and test set. For every experiment involving the same domain (i.e., for TL, it is tested on the target domain), the same test set was used.

### Further Restrictions on the Data

For the initial two experiments, there are additional critical requirements for the data sources which needed to be considered. First of all, a learning algorithm should not be able to easily differentiate across matches and non-matches (i.e., achieving an f1 score close to $100\%$).

Secondly, a single attribute should not be sufficient to learn a perfect model, while the other attributes are not considered relevant to the matching task. If these requirements are not met, the data is not useful for the experiments due to its simplicity.

For the topic kitchen product offers the attributes *title* and *description* led to $\sim 100\%$ f1 score when training a Random Forest Classifier with $100$ estimators and default parameter setting using k-fold cross-validation with $k = 4$ and stratified splits on all instances. The remaining attributes, however, did not play a role. The following strategies were tried to mitigate the influence of the *title* and *description* attribute and make the data more difficult for the learning algorithm. First of all, the attribute *description* was removed from the candidate sets. Here the values were almost the same for both records of a matching record pair.

As a second step, it was tried if excluding examples where the classifier achieved a predicted probability greater than $99\%$ for being a match, and the number of missing values among the remaining attributes (i.e., the ones except *title*) was higher than $50\%$, leads to worse results and more influence of the remaining attributes.

However, this did not lead to the desired results either. By inspecting the *title* attribute values in more detail across the record pairs, it was encountered that a product code was included in the *title*. That was the case for all sources and the reason the learning algorithm only focused on the *title*. Therefore, the only option was to create some noise in the *title* by removing the leading two words from the string, which included the product code. After that, the influence of the remaining

attributes could be increased, and the overall performance dropped significantly.

Due to this approach, the kitchen product offers data could have been used in the experiments. Please refer to the section 6.3 for the final profiling results of the kitchen product offer data.

## 5.4 Experimental Setting

To evaluate which factors need to be considered to develop an ER method that combines TL and AL, the following experiments were conducted on two semantic topics. The experiments are only briefly introduced here, and a more detailed description of each experiment can be found in the dedicated subsections.

First of all, TL experiments were conducted. It was evaluated how a classification model learned on the source domain performs when applied to the test set from the target domain (i.e., naive transfer) using different learning algorithms and two different selections of features. The first feature set is using all features. The second feature set is only using features that have high density in both domains (i.e., excluding features that have only little missing values in one domain but almost only missing values in another domain).

Secondly, based on the results of the TL experiments, it was researched if incorporating all labeled source domain data in the labeled set of AL can help to achieve better performance compared to AL with only target domain data. Additionally, two different unsupervised domain adaptation techniques (explained in subsection 4.3.1) were tested to evaluate if the classification model learned on the source domain data can be adapted to the target domain before transferring knowledge without the need of labeled target domain data.

According to the findings of the experiments, a method was developed, which mitigates the identified risks. This methodology was then evaluated on the data from the different semantic topics and compared to the AL method for ER proposed by Primpeli et al. [62]. This method follows a similar AL approach but without incorporating knowledge from another domain but by leveraging an unsupervised matching technique [62] as explained in subsection 3.4.3.

#### Combinations of Source and Target

In each experiment, there is always one candidate set acting as the source domain, for which labeled data is available, and one target domain involved. For the target domain, it is assumed that no labeled data is available. For the AL experiments, though, labels for some instances of the target training set can be acquired during the experiments. The performance is always measured on the respective test set

from the target domain. For the source domain, always the whole candidate set with labeled record pairs is incorporated.

At least three different data sources with matching entities are available for each semantic topic. All candidate sets that share at least one data source with the target domain are considered potential source domains. Therefore, for the topic books with a total of four different data sources, six different candidate sets (i.e., combinations of data sources) are available. For this reason, 24 experiments were conducted with different source-target combinations, with one candidate set acting as the source and another as the target. It should be noted that the source and target roles can be reversed, which means another experiment.

For the other two topics, kitchen product offers and authors, only three data sources are available. Therefore, only three different candidate sets were created, and a total of six different source-target combinations were performed.

### 5.4.1 Naive Transfer Learning (NTL) Experiments

The setting of the naive transfer experiments is visualized in Figure 5.1. With a naive transfer, it is meant that a learning algorithm only learns from the labeled data of the source domain and is applied without any adaptation (i.e., hence naive) on the target domain (explained in subsection 4.3.1). The naive transfer is pictured in the upper part of Figure 5.1. As can be seen in Figure 5.1, three different baseline methods are used to put the results of naive transfer into context.

**Baseline 1**: The first baseline is unsupervised matching, as introduced in subsection 2.3.4. The method used is the unsupervised matching approach proposed by Primpeli et al. in [62]. The method is explained in subsection 3.4.3. The authors use this method to bootstrap their AL method for ER. Therefore, this baseline is considered the lower bound for the naive transfer results; if the naive transfer results are below this baseline, it is considered a negative transfer, which most likely harms AL on the target when incorporating this knowledge.

**Baseline 2**: As the second baseline, the averaged target results for random samples of the training data from the target is used. That means that random samples of size $x$ are drawn from the target domain's training data and used for training a learning algorithm. The learned model is then applied to the test set of the target. The results are averaged over ten random samples for each sample size $x$ with values in the range of 10 to 500[2]. This baseline indicates how many target instances are roughly needed to achieve results similar to those of a naive transfer.

[2] x = [10, 14, 20, 24, 28, 32, 38, 44, 50, 60, 70, 80, 90, 100, 120, 140, 160, 180, 200, 300, 500]

**Baseline 3**: The last baseline is passive learning, which means the learning algorithm learns on all labeled instances from the target domain and is applied to the test set from the target. This baseline can generally be regarded as an upper bound. One cannot expect from naive transfer to produce better results than this baseline. However, it should serve as a benchmark of how far naive transfer results are away from training on the target domain rather than the source domain.

The learning algorithms used for the naive transfer and the learning-based baseline methods are Logistic Regression, Logistic Regression CV (i.e., cross-validated logistic regression with tuning the C hyperparameter), SVM (with a linear kernel), Decision Tree, Random Forest, and XGBoost. All learning algorithms are explained in subsection 2.3.5. The parameter setting is provided in Figure A.3. All baseline methods and NTL are evaluated on the same test set from the target domain using the f1 score measure. All the baseline methods do not incorporate any source domain knowledge.

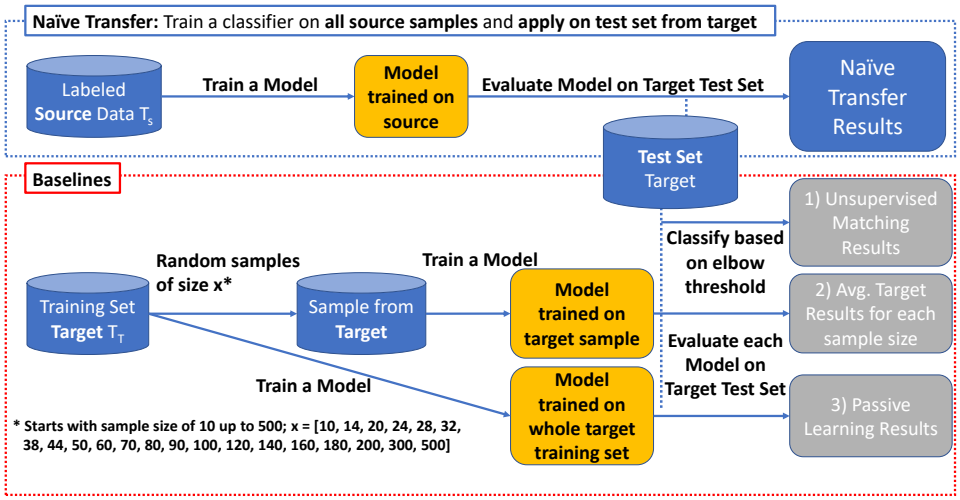


Figure 5.1: Naive Transfer Learning Experiment

### Research Questions

The NTL experiments shall mainly provide an initial understanding of potential pitfalls when incorporating knowledge from another domain, and the following research questions are asked.

- RQ1.1: Which type of learning algorithm achieves better results for naive transfer?
- RQ1.2: Do substantial differences in the density of the features across the source and target domain influence the performance of naive transfer?
- RQ1.3: For which combinations does naive transfer perform worse than unsupervised matching and what are potential causes for this?

### 5.4.2 Active Transfer Learning (ATL) Experiments

The second set of experiments is about incorporating the labeled source domain data into the pool-based AL setting by adding the whole labeled source data into the labeled set before starting the AL process.

The whole experimental setup is shown in Figure 5.2. Additionally, it is researched whether performing unsupervised domain adaptation (as described in subsection 4.3.1) in the form of weighting the labeled source domain data based on their estimated importance for the target domain can increase performance. That is illustrated in the lower part of Figure 5.2 and is especially interesting for the source-target combinations that are suffering from the negative transfer, as shown in the NTL experiments. As unsupervised domain adaptation techniques, Nearest-neighbor based Weighting (NN) (see section 4.3.1) and the customized weighting based on the predicted probability of a Logistic Regression model discriminating between source and target samples (lr_predict_proba) (see section 4.3.1) is used.

As mentioned before, the labeled set of the pool-based active learning is bootstrapped with all labeled source data. Subsequently, one target instance per iteration, up to a maximum of 100 target instances, is queried from the pool of unlabeled record pairs.

These unlabeled instances are stemming from the training set of the target. A Query By Committee strategy (explained in subsection 3.3.2) with heterogeneous committee members is employed to retrieve the most informative record pair per iteration. That has shown to perform better than homogeneous committees where the members only differ in their parameters [13]. In total, four members are in the committee with two linear and two non-linear ones, namely Logistic Regression, SVM (linear kernel), Decision Tree, and Random Forest. Each of them is trained on the labeled set of the current iteration and makes its prediction on the unlabeled pool of target instances.

The instances with the highest disagreement in the committee members' predictions are selected as the most informative record pairs. The disagreement is measured using vote entropy. From the most informative record pairs, one is randomly selected and provided to the oracle (i.e., a perfect labeler; in a real-world

setting, this could be a human annotator) to retrieve the corresponding label. After that, the newly labeled record pair is added to the labeled set. As a final step of one single iteration, the AL model is trained on the labeled set. The performance of the AL model is measured at each iteration on the target's test set using the f1 score. Every run of 100 iterations is repeated five times to account for the random selection of the most informative record pairs that share the same disagreement among the committee members.

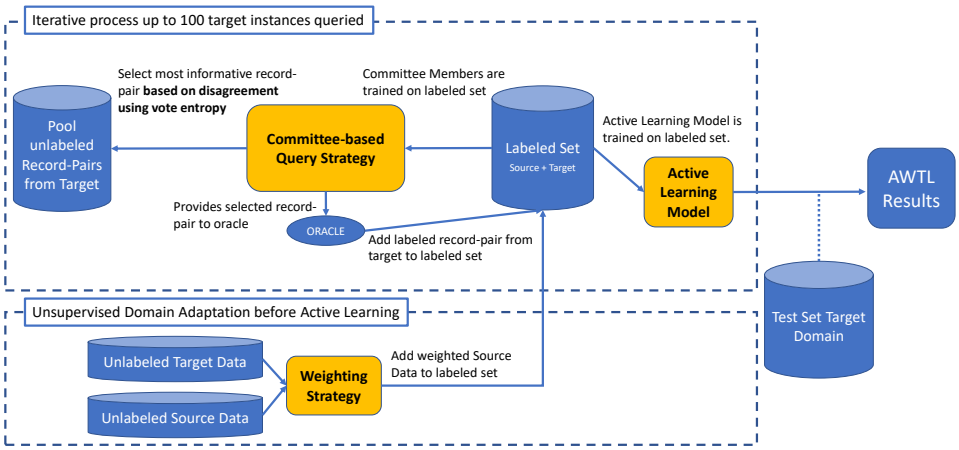


Figure 5.2: Active (weighted) Transfer Learning Experiment

The following baselines are used in order to compare the performance results of the ATL experiments.

**Baseline 1** Random Sampling. Here the same setting is used, but instead of picking the most informative sample from the unlabeled pool based on the committee, random sampling is applied. This baseline shall provide insight into whether the query strategy is not misled by the labeled source samples in the labeled set.

**Baseline 2** AL without Source Data. The overall results are compared to AL in the target domain without incorporating the labeled source data. The initially empty labeled set is bootstrapped by randomly sampling from the target data until at least one positive and one negative record pair is added to the labeled set. It is important to note that this baseline does not tackle the cold-start problem very well, as randomly sampling until one positive and one negative sample is included could lead to an increased labeling cost. As a baseline, however, it serves its purpose.

**Research Questions**

The active transfer learning experiments build upon the naive transfer results and aim at answering the following questions.

- RQ2.1: Does the query strategy perform better than random sampling?
- RQ2.2: How does this simple active learning setting with transfer learning perform compared to the active learning baseline?
- RQ2.3: Leads Unsupervised Domain Adaptation to better performance for source-target combinations that suffer from negative transfer?

## 5.5 The proposed ATL method for ER (ATLX)

Based on the NTL and ATL experiments' findings, a final method was developed that tries to account for the weaknesses and pitfalls identified when incorporating source domain knowledge into an AL setting for ER. First, the methodology of ATLX will be explained in subsection 5.5.1 in detail. In subsection 5.5.2, it is explained which baselines methods are used to evaluate ATLX.

### 5.5.1 Methodology

The final method, combining AL and TL for ER, considers the implications of the NTL and ATL experiments (see subsection 7.2.5). That is, the final method helps the AL model to better adapt to the target domain. It also minimizes the influence of source domain data in the query strategy. A similar AL approach as proposed by Primpeli et al. [62], which is explained in subsection 3.4.3, is followed.

The proposed method starts with training a Random Forest classifier (with 100 estimators) on the labeled source domain data (source model). This classifier has shown to achieve good naive transfer results in the NTL experiments, as shown in the results in section 7.1. Additionally, domain adaptation techniques can be used to train the source model if the estimated domain relatedness of both domains is rather low. The source model is used to predict the labels for the unlabeled target domain data, creating a pool of labeled target domain data with some degree of noise (the noisy pool).

Besides that, instead of adding any labeled source domain data into the labeled set the record pairs from the target domain where the source model has the highest confidence (i.e., highest predicted probability) are added to the labeled set. The preset number of record pairs from the target used to bootstrap the labeled

set can be chosen based on the source and target domain's estimated domain relatedness. As higher the estimated domain relatedness, the more record pairs for the bootstrapping sample. A requirement is to use as many positive as negative record pairs to balance the bootstrapping sample. Besides that, the bootstrapping instances' predicted probabilities need to be above $90\%$ to mitigate the risk of *false positives* and *false negatives*. Note that even for the most confident record pairs, the predicted labels may suffer from noise. Therefore, a more conservative approach uses only the most confident positive and negative record pair as a bootstrapping sample (i.e., only two instances for bootstrapping).

The committee consists of two linear and three non-linear models: Logistic Regression, SVC (linear kernel), Decision Tree, Random Forest, and XGBoost. The same committee members are used in the method of Primpeli et al. [62] and have shown to perform well. Because of the bootstrapping sample, the committee can be trained on the target domain data from the start without manually labeling any target domain data. It, thus, successfully tackles the cold-start problem.

Additionally, the labels in the noisy pool count as one additional vote. That means that per iteration, the query strategy selects one record pair from the pool of unlabeled target data, which accounts for the greatest disagreement among the four committee members' votes plus the vote from the source model. The disagreement is measured using vote entropy. From the record pairs with the highest disagreement, one instance is randomly selected.

The AL model, a Random Forest classifier with ten trees and warm_start parameter set to true, is also initialized before the actual AL process starts. Instead of training the initial AL model on the bootstrapped labeled set, the whole labeled source domain data is used.

After that, the approach of expanding the initial random forest with two more trees per iteration trained on the current labeled set is the same as in the method proposed by Primpeli et al. [62], as explained in subsection 3.4.3. This approach ensures that more and more trees are added to the random forest, trained only on labeled target domain data, while the trees learned in previous iterations, including the initial ten trees learned on the source domain data, remain. This procedure ensures that the labeled source domain data have more influence on the active learning model at the start where still too little target domain data is labeled. However, once more target data is labeled, the model adapts more to the target domain.

This approach has shown to facilitate high quality and stable models already after a few iterations [62]. To conclude, the only difference to the method proposed in the work of Primpeli et al. [62] is that they incorporate unsupervised matching knowledge instead of knowledge from another domain.

The whole proposed method, called ATLX (with X referring to the e**X**panding random forest), is depicted in Figure 5.3.

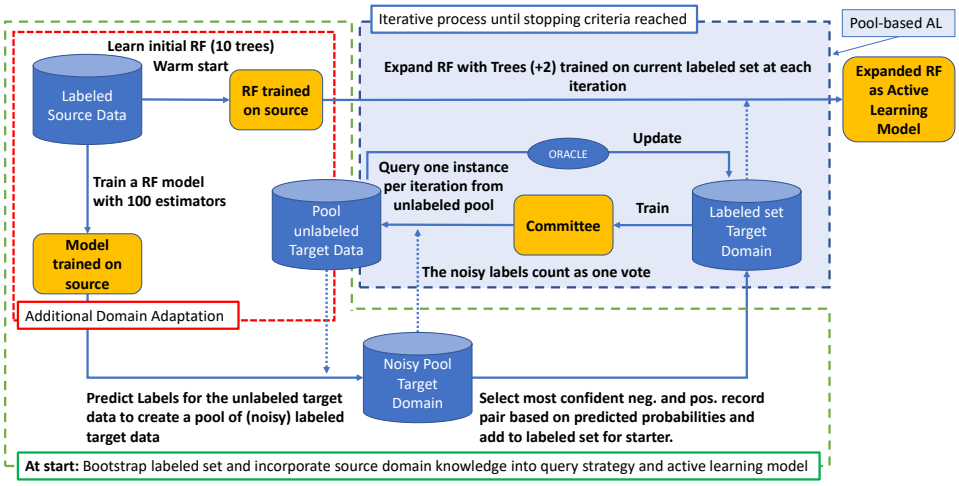


Figure 5.3: ATLX - The Proposed Active Transfer Learning Method for ER

### 5.5.2 Evaluation

The following explains how ATLX is evaluated on the three different semantic topics books, kitchen product offers, and authors (see chapter 6). Three baselines are considered:

**Baseline 1** Random Sampling. This baseline shows, as in the initial ATL experiments of subsection 5.4.2, whether the query strategy is capable of selecting valuable record pairs from the pool of unlabeled target instances. Therefore, this baseline has the same AL setting, except that instead of using the query strategy, an instance is randomly selected from the pool of unlabeled instances at each iteration. If random sampling achieves better results than ATLX, the query strategy cannot select instances that help the active learning model better generalize to unseen data.

**Baseline 2** Active learning method with unsupervised bootstrapping [62]. This baseline is explained in detail in subsection 3.4.3. As mentioned in subsection 5.5.1, the only difference from ATLX is that it does not use TL to bootstrap the query strategy committee and kickstart the AL model, but an unsupervised matching technique. This baseline provides information about whether reusing existing knowledge from another domain is superior to bootstrapping AL with unsupervised matching knowledge.

**Baseline 3** Passive learning results. This baseline can be considered as the upper bound. In the best case, ATLX leads to comparable results as this baseline.

## 5.6 Data Profiling Dimensions

Profiling the data used in the experiments is essential to understand the results better. Especially for the NTL experiments, the profiling dimensions explain why the naive transfer between two domains did not work. The profiling dimensions used can be divided into three categories. The first category, data source-level, presents profiling results for each data source. In contrast, the second category, candidate set-level, profiles the candidate sets that contain the record pairs stemming from two data sources. The creation of the candidate sets is explained in section 5.1. The last category examines the source-target combinations that are considered in the experiments.

### 5.6.1 Data source-level

On the data source-level the main profiling dimensions are *Attribute Density*, *Value Overlap*, *Attribute Length/Format*. *Attribute Density* is the percentage of non-missing values. A density of $100\%$ signifies that no missing values exist for this attribute. A *Value Overlap* of $0\%$ implies that this attribute only has unique values. Hence, a higher percentage in *Value Overlap* for a certain attribute means that many entity descriptions in the data source share the same value (i.e., overlapping values). For *Attribute Length/Format* the distribution of the attribute length for strings is visualized in a boxplot (i.e., data types *string* and *date*). The distribution of the values from attributes of type *number* is also shown in a boxplot.

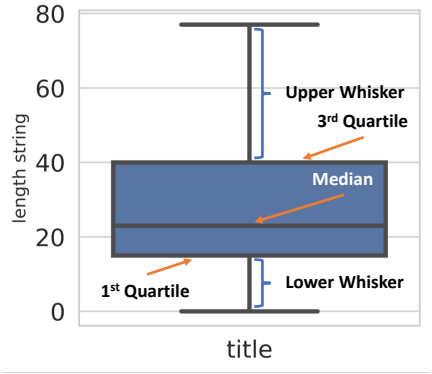


Figure 5.4: Example Boxplot

In Figure 5.4 an example boxplot is shown. A boxplot provides an overview of the distribution of numeric data. The upper bound from the box plot shows the

third quartile, which means $25\%$ of the data is greater than this value. In contrast, the bottom of the box refers to the first quartile, which means $25\%$ is smaller than this value. Therefore, the box corresponds to the zone in which the middle 50% of the data is located. The length of the box corresponds to the interquartile range (IQR), which indicates the dispersion of the data. A line within the box represents the median, which refers to the middle of a data set (i.e., $50\%$ of the data is greater than this value and $50\%$ smaller). Additionally, the minimum and maximum values (i.e., excluding outliers) are shown with a horizontal line at the end of the so-called whiskers (denoted as lower and upper whiskers in Figure 5.4). The length of the whiskers is determined to be at most $1.5 * IQR$. However, the potential differences in the length of the upper and lower whiskers originate from the fact that the largest or smallest data point within the range of $1.5 * IQR$ defines the whiskers' end (i.e., the horizontal line). Normally, data points below the lower or above the upper whisker are considered outliers and are visualized using dots in a boxplot. In Figure 5.4 no outliers are visualized. That is also the case for the boxplots used to visualize the *Attribute Lenght/Format* profiling dimension. However, the mean is additionally visualized using a small white square with a blue outline.

The aligned attributes, data types, and the size of each data source are shown in Table 6.1, where an overview of all semantic topics is given.

### 5.6.2 Candidate set-level

On the candidate set-level, the profiling dimensions *Schema Complexity (SC)*, *top relevant attributes and relevant attributes*, *Size (S)*, *Corner Cases (CC)* are taken from a very recent work from Primpeli et al. [61] which are explained in section 2.4. Another profiling dimension also proposed by Primpeli et al. [61] is called *Sparsity* and is also explained in section 2.4. In this thesis, however, this is reversed and called *Density* (i.e. $1 -$ *Sparsity*). It does provide the same information but aligns with the *Attribute Density* dimension used in the data source-level profiling.

Besides these profiling dimensions, the similarity values are aggregated for each attribute, and the distribution is visualized using a boxplot (example boxplot in Figure 5.4), separated by the label (i.e., match or mismatch). Thus, the differences in the similarities of each attribute between matches and mismatches can be seen for each candidate set. A similar distribution of matches and mismatches for a given attribute indicates that this attribute cannot play a decisive role in distinguishing between a match and a mismatch. However, if an attribute mainly shows a high similarity for matches and a low similarity for non-matches, it is a reliable indicator for a match and a non-match. This visualization should reflect the *relevant*

*attributes* and *top relevant attributes* dimensions.

Additionally, to the average density of the whole candidate set, which is one of the profiling dimensions mentioned above, a heat map is provided. In that heat map, the density for every single attribute and candidate set is visualized.

It is important to know that the proportion of positive and negative record pairs (i.e., matching and non-matching) in the candidate set must be known. Within the candidate sets for the two topics *books* and *kitchen product offers* the two classes are balanced (as explained in section 5.1). However, the *authors* topic was used in another work [62], and the candidate sets were taken as they are. For them, the two classes are not balanced, and therefore the ratio of positive matches is additionally given as a candidate set level profiling dimension.

### 5.6.3 Source-target combination-level

This category profiles the source-target combinations of the experiments. Here the domain relatedness between the candidate set that acts as the source domain and the candidate set that acts as target domain is estimated. This profiling dimension shall provide insight if a potential transfer of knowledge across two domains can work well.

For calculating the domain relatedness, the two candidate sets are merged into one data set. A column is added to that data set, which indicates to which candidate set a record belongs. This column is used as the target label for a Logistic Regression model (the Logistic Regression CV classifier from scikit-learn is used). Additionally, the class_weight parameter of the Logistic Regression classifier is set to *balanced*. That setting is important as for the topic books, a candidate set which is much larger than the remaining ones exists. The model is trained using five-fold cross-validation, and the performance is measured using MCC, as explained in subsection 2.3.6.

The idea behind this method for estimating the domain relatedness is similar to the intention behind the importance weighting technique described in section 4.3.1. If the logistic regression model can correctly differentiate across the two candidate sets, they differ in their marginal distributions and therefore are not very related to each other. In an article by Francisco J Martin [28], the author proposes a similar method in order to detect covariate shift, which represents a shift in the marginal distributions (as explained in subsection 4.3.1). According to the author, a rule of thumb says that if MCC is below $0.2$, the two data sets come from a somewhat similar distribution and are more related to each other [28].

Note that the domain relatedness is only calculated on the features, and the label information is not incorporated. Therefore, the domain relatedness can always be measured between a source and target domain, even if only unlabeled data is

available from the target domain. The aim is to see if harmful performance in the naive transfer can be explained using this domain relatedness measure.

The domain relatedness is measured on two different feature sets, that are also considered for the experiments. The first feature set contains all features (denoted as *all*), and the second feature set excludes features associated with attributes that have a very low density in at least one candidate set (denoted as *dense*).

# Chapter 6

# Data Profiling

In this chapter, the profiling results of the data used for the experiments are listed. In total, three different semantic topics were used. Two of them were used for the initial experiments, but all three were used to evaluate ATLX. Before the profiling results and the data collection process for each semantic topic are presented, an overview of all topics is given.

## 6.1 Overview Semantic Topics

In Table 6.1 all three semantic topics are listed. The data sources from where the data is stemming from and the attributes with data types are provided. Additionally, the size of each data source is listed.

| Topics | Data Sources | Size | Attributes (Data Type) |
|---|---|---|---|
| Books | Barnes & Noble (ban)<br>Bookcrossing (bx)<br>Half Price Books (half)<br>Wordery (wor) | 17,692<br>270,948<br>3099<br>48,792 | authors (string)<br>binding (binary)<br>title (string)<br>pages (number)<br>pubdate (date)<br>publisher (string) |
| Kitchen Product Offers | culinarydepotinc (cdi)<br>katom<br>rewonline (rewo) | 7677<br>25,635<br>82,385 | base (string)<br>brand (string)<br>capacity (string)<br>category (string)<br>color (string)<br>finish (string)<br>height (string)<br>material (string)<br>shape (string)<br>style (long string)<br>title (long string)<br>product_type (string) |
| Authors | DbPedia (dbp)<br>DNB<br>VIAF<br>Wikidata (wiki) | 31,604<br>9998<br>10,186<br>31,480 | birthdate (date)<br>deathdate (date)<br>gender (string)<br>name (string)<br>work (long string) |

Table 6.1: Overview Semantic Topics

## 6.2 Topic: Books

In this section, the data acquisition of the topic books is briefly explained before the profiling results are presented. The data sources with their respective size and the aligned attributes are listed in Table 6.1.

### 6.2.1 Data Acquisition

The setting researched in this thesis requires at least three data sources for each semantic topic with a sufficient number of matches among them and aligned schemata. Therefore, for the topic books, data was gathered in the following way.

In the entity resolution data repository of AnHai Doan's group [20] several

book datasets can be found. There are datasets from Amazon, Half Price Books, Barnes and Noble, and Goodreads with information about books and common attributes. When for one source, more than one dataset was available, the schema was aligned, and they were merged while removing duplicates based on the ISBN. Here the record that provided most information was kept. Additionally, a further source with book information from www.bookcrossing.com (BX) was found on the internet [81]. All five datasets come with ISBNs that can serve as a unique identifier to identify positive correspondences among them correctly.

Although many sources share positive correspondences with at least one more source, not all of them share sufficient true matches with all of them. Because of this, another source was selected (namely www.wordery.com). Here book data was crawled not only based on the ISBNs that exist in the other datasets but also based on author names so that also new books are discovered with the crawl and not only matching books. In the end, six different sources with data about books were available. However, as the experimental setting only requires at least three different data sources for one topic, it was evaluated which sources share the highest number of positive correspondences. It was also checked which common attributes can be considered across the best four sources so that the amount of missing values is kept low. Here title, author(s), binding (with values 'Hardback' or 'Paperback' if available), pages, publication date (pubdate), and publisher was selected. If for a particular source, this information was not provided, an empty column was created so that this attribute is available (albeit only with missing values). However, this was only the case for BX. The final four sources for the topic books are Half Price Books, Barnes and Noble, Wordery, and BX, as shown in Table 6.1. More experiments can be conducted with four data sources than with only three as more source-target combinations are available.

### 6.2.2 Profiling Results

In this subsection, the profiling results on the data source-level, candidate set-level, and source-target combination-level for the semantic topic books are provided.

#### Data source-level

The *Attribute Density* and *Value Overlap* for each attribute and data source is visualized in Figure 6.1. The attribute *title* has, in all data sources, a very low value overlap. Because titles of books are most of the time unique and each single data source is deduplicated, this makes sense. Possible reasons for value overlap in the title are different editions of the same book. Whether two records describe the same real-world entity depends on the exact match of the ISBN. The same book

published by another publisher or with a different binding (i.e., paperback or hardback) has a different ISBN and therefore does not describe the same real-world entity. The density from *title* is also very high because no missing values exist in all data sources. In general, for all sources, the densities of the attributes are quite high and similar, except for three deviations. First of all, the attribute *binding* contains many missing values in all sources except in *wor*. Secondly, as mentioned before *pages* only contains missing values in *bx*. In other sources, *pages* has a very high density. Finally, the attribute *pubdate* has close to 100% density in all data sources except in *ban*. Here the half of the records have missing values for the attribute *pubdate*.

The value overlap is for most of the attributes very high. This can be explained by the fact that the range of possible values for attributes like *pages*, *publisher*, *pubdate*, and especially for *binding* is rather limited. Also, the crawling strategy for the data source *wor* (as explained in subsection 6.2.1) has somewhat limited the variety of attribute values, since books by the same author have been crawled.

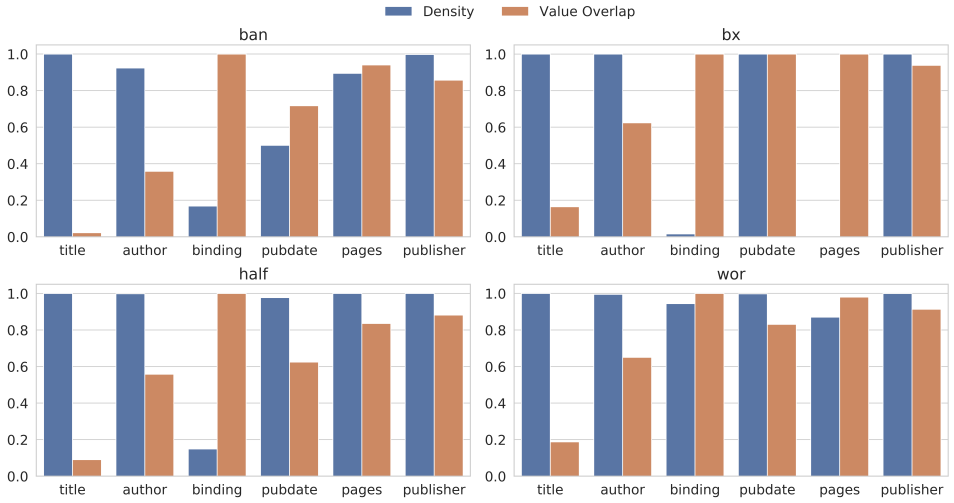


Figure 6.1: Topic Books - Attribute Density and Value Overlap

In Figure 6.2 the distribution of the length of the three string attributes *title*, *author*, and *publisher* is visualized on the left side. Here one can see that there are differences in the *title* attribute across the four data sources. The attribute length from the other two attributes is rather similar. The attribute *pubdate* which is of data type *date* (i.e., like a string) is shown in the middle of Figure 6.2. Obviously, the values of *pubdate* in the source *bx* only contain the year as the length is always exactly four characters without any variation. In contrast, the source *ban* provides

many different formats for the date as the length differs a lot. The other two sources seem to provide the same format for all of their records.

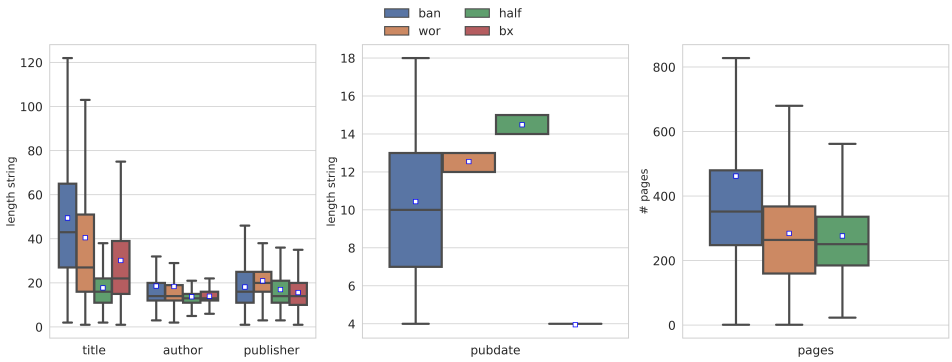


Figure 6.2: Topic Books - Attribute Length

On the right side of Figure 6.2 the distribution of the *pages* attribute is shown. For *wor* and *half* the distribution of the number of *pages* is quite similar in contrast to *ban*. It should be noted that some outliers are present in the data sources but are excluded from the plots because they are not included in the final candidate sets and would interfere with the plotting here.

### Candidate set-level

In Table 6.2, the profiling results on the candidate set-level are shown. Among the relevant attributes, the top relevant attributes are highlighted in bold. It is important to note that the attribute *pubdate* is considered a top relevant attribute for every candidate set. For the candidate sets ban_bx, ban_half, and bx_half, only *pubdate* is considered a top relevant attribute. Therefore, they most probably also have a lower share of corner cases (denoted as CC in Table 6.2). Recap that the share of corner cases is calculated based on the top relevant attributes (explained in subsection 5.6.2). In contrast, for ban_wor additionally *pages* is considered a top relevant attribute. Based on the high value overlap from the attribute *pages* (see Figure 6.1) this is somewhat strange. However, when looking at Figure 6.3 one can see that *pubdate* contains some outliers with low similarity scores for matches and probably *pages* helps discriminating among those. For the candidate set bx_wor besides *pubdate* also *publisher* is considered as top relevant attribute. Wor_half is the only candidate set which has three top relevant attributes. The three candidate sets ban_wor, bx_wor, and wor_half all have a share of corner cases above 50%.

In Figure 6.3 it is also easy to see that each candidate set contains some very

| Set | Relevant Attributes | % all | Profiling Dimension D | SC | SZ | CC |
|---|---|---|---|---|---|---|
| ban_bx | **pubdate (date)**, author (str), publisher (str), title (str) | 0.67 | 0.64 | 4 | 1398 | 0.34 |
| ban_half | **pubdate (date)**, publisher (str), pages (num), title (str) | 0.67 | 0.81 | 4 | 1772 | 0.13 |
| ban_wor | **pubdate (date), pages (num)**, author (str), publisher (str), title (str) | 0.83 | 0.82 | 5 | 1940 | 0.52 |
| bx_half | **pubdate (date)**, author (str), publisher (str), title (str) | 0.67 | 0.67 | 4 | 1100 | 0.25 |
| bx_wor | **pubdate (date), publisher (str)**, author (str), title (str) | 0.67 | 0.67 | 4 | 18,540 | 0.61 |
| wor_half | **pubdate (date), publisher (str), pages (num)**, author (str), title (str) | 0.83 | 0.84 | 5 | 2740 | 0.63 |

Table 6.2: Topic Books - Candidate Set Profiling Results

severe cases. No clear distinction can be made among the attributes that would normally be considered important for distinguishing between match and non-match. In all candidate records, some record pairs do match even though their similarity in the title or author is very low. In addition, many negative record pairs have high similarity values for the attribute *author* and even for *title*.

However, the latter case can be explained by the fact that two records with the same title, author, and publisher do not necessarily refer to the same real-world entity, as they may differ in their binding or refer to a different edition and therefore have a different ISBN. Nevertheless, in this case, at least the *pubdate* and the *binding* attribute if provided, would differ. Note that hardbacks are published two years before paperbacks. Thus, the *binding* attribute has a very high value overlap and, more importantly, for the candidate sets considered here, many missing values (see Figure 6.4 as opposed to *pubdate*). That is probably the reason why *binding* is not considered a relevant attribute for all candidate records, but *pubdate* is.

In Figure 6.4 the densities for each individual attribute and candidate set are visualized by means of a heat map. Here one can see that the attribute *pages*, as already mentioned, is not present in the data source *bx* and therefore all candidate sets containing *bx* have a density of zero. Apart from that, the densities are quite similar across all candidate sets. Also note that the data source *ban* has a total

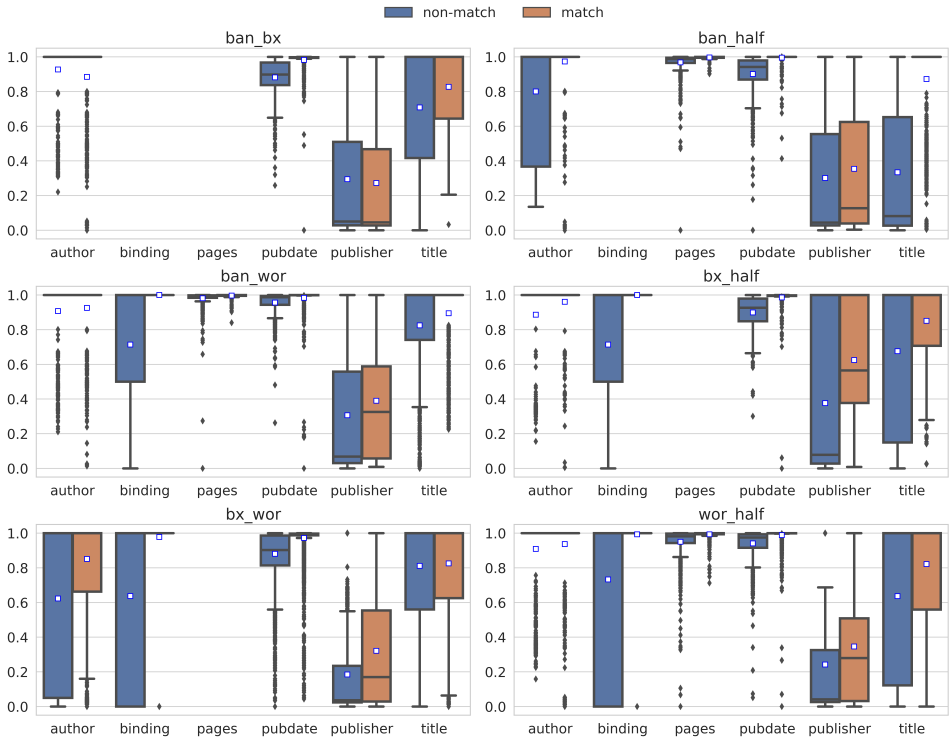


Figure 6.3: Topic Books - Aggregated Sim Scores per Attribute separated by Label

density for *pages* of about $50\%$ (see Figure 6.1), but the candidate sets containing *ban* still have a high density for *pages*. Obviously, the records that provide a value for *pages* were selected primarily.

### Source-target combination-level

In Table 6.3, the domain relatedness for each source-target combination of the topic books is provided. As explained in subsection 5.6.3, an MCC below $0.2$ indicates that the two domains are more closely related because the learned logistic regression model cannot distinguish very well from which domain a particular record pair originates. The MCCs below $0.2$ are highlighted in bold in Table 6.3.

The reason for calculating domain relatedness with two different sets of features, namely *all* and *dense*, is mainly because the source BX only contains missing values for the attribute *pages*. Therefore, the domain relatedness score is close to $1$ for the combinations where the data source BX occurs only once (i.e., ban_bx - ban_half, ban_bx - ban_wor, bx_wor - wor_half, bx_wor - ban_wor, ban_half -

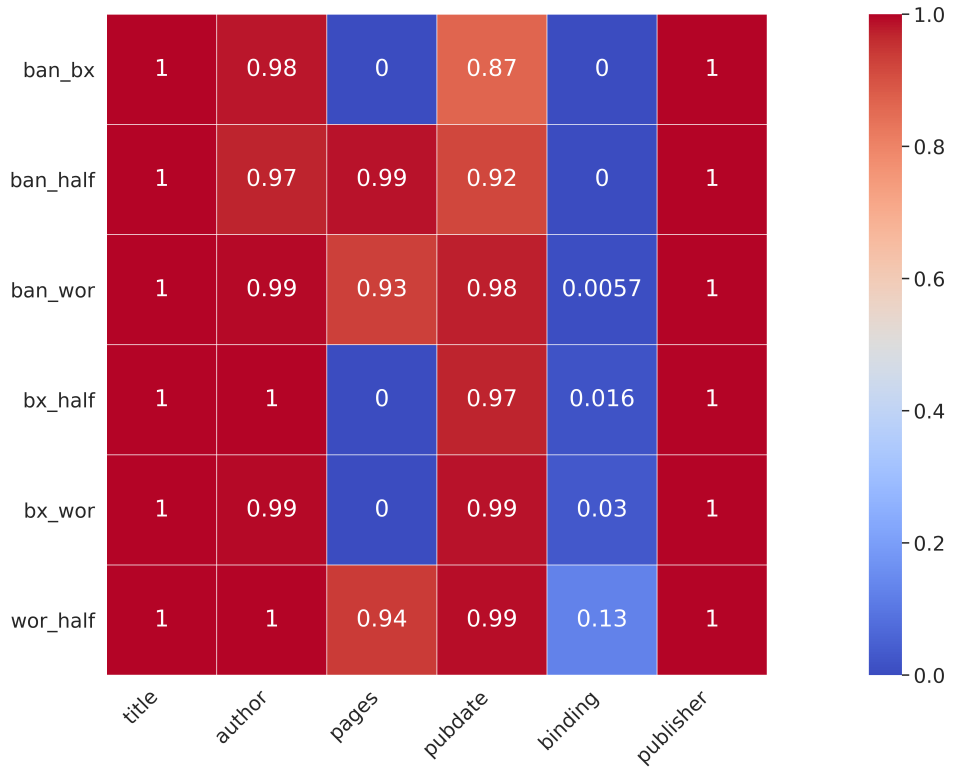


Figure 6.4: Topic Books - Heat map of Attribute Density in all Candidate Sets

bx_half, wor_half - bx_half) and all features are used. However, if the attribute *pages* is excluded (denoted as *dense*) then the MCC is significantly lower. The two feature sets are also used for the initial NTL experiments (explained in subsection 5.4.1).

Overall more combinations have an MCC above $0.2$ than below, and therefore it can be assumed that they differ in there marginal distributions, which can be problematic for the naive transfer.

| Combination | MCC per Feature Set | |
|---|---|---|
| | all | dense |
| ban_bx - bx_wor | **0.118** | **0.098** |
| ban_bx - ban_half | 0.969 | 0.417 |
| ban_bx - bx_half | 0.309 | 0.299 |
| ban_bx - ban_wor | 0.925 | 0.493 |
| bx_wor - wor_half | 0.965 | 0.435 |
| bx_wor - bx_half | 0.232 | 0.25 |
| bx_wor - ban_wor | 0.962 | 0.347 |
| ban_half - wor_half | 0.227 | **0.159** |
| ban_half - bx_half | 0.994 | 0.314 |
| ban_half - ban_wor | 0.272 | 0.268 |
| wor_half - bx_half | 0.903 | 0.434 |
| wor_half - ban_wor | 0.216 | **0.190** |

Table 6.3: Topic Books - Domain Relatedness for each Source-Target Combination

## 6.3 Topic: Kitchen Product Offers (kitchen)

In this section, the data acquisition of the topic kitchen is briefly explained before the profiling results are presented. The data sources with their respective size and the aligned attributes are listed in Table 6.1.

### 6.3.1 Data Acquisition

The product entities from the topic kitchen product offers originate from the Large-Scale Product Matching corpus Version 2.0[1], which contains structured product offer entities from different e-commerce shops. The corpus was created by the Data and Web Science Group of the University of Mannheim. To create the corpus, they used semantically annotated product offers found on the internet, which came with some product ids. Based on these product ids, product clusters were created where the product offers assigned to one product cluster refer to the same real-world product entity in the best case.

For this thesis, data about product offers from three data sources, culinarydepotinc (cdi), katom (katom), and rewonline (rewo), were selected from the corpus as they contained a sufficient amount of matching entities among each other. The three e-shops sell restaurant equipment. Therefore, the product offers are mainly

[1] http://webdatacommons.org/largescaleproductcorpus/v2/index.html

about kitchen-related products. The common attributes, other than *title* and *brand*, among the data sources, are derived from the specification tables of the product offers and contain many missing values. As explained in section 5.3, the *description* attribute, which is also included in the Large-Scale Product Matching corpus, was excluded from the final candidate sets, due to too much overlap among the matching entities from the different data sources.

An additional preprocessing step required to make the matching task more difficult (i.e., a requirement for the experiments) was the introduction of noise in the attribute *title* (as explained in section 5.3). That was necessary because it was discovered that the title contained a product ID that allowed almost all matches to be identified perfectly. From the specification table attributes, only redundant words were removed. Apart from that, no further preprocessing was conducted.

### 6.3.2 Profiling Results

In this subsection, the profiling results on the data source-level, candidate set-level, and source-target combination-level for the semantic topic kitchen are provided. To get an overview of the data sources, the size of the data sources, and the schema's attributes with the data types, please refer to Table 6.1.

#### Data source-level

The *attribute density* and *value overlap* for each attribute and data source are visualized in Figure 6.5. The *title* attribute has a value overlap of $0\%$ but a density of $100\%$ in all data sources. In contrast, *brand*, which also has a very high density, also has a very high value overlap. That makes sense because not many brands make kitchen-related products and equipment for restaurants. The attributes that come from the specification table all have lower density and higher value overlap. Apart from this, density and value overlap are more or less similar for all three sources. Only the *height* attribute has a different density between the data sources.

In Figure 6.6 the distribution of the length of the attribute values is visualized for each attribute. Here one can see that there are not so many significant differences between the three data sources. Only for some attributes from the specification table such as *height*, *category* and *base* at least one source has values with completely different characteristics.

#### Candidate set-level

In Table 6.4, the profiling results are displayed at the level of the candidate set. Among the relevant attributes, the top relevant attributes are again highlighted in

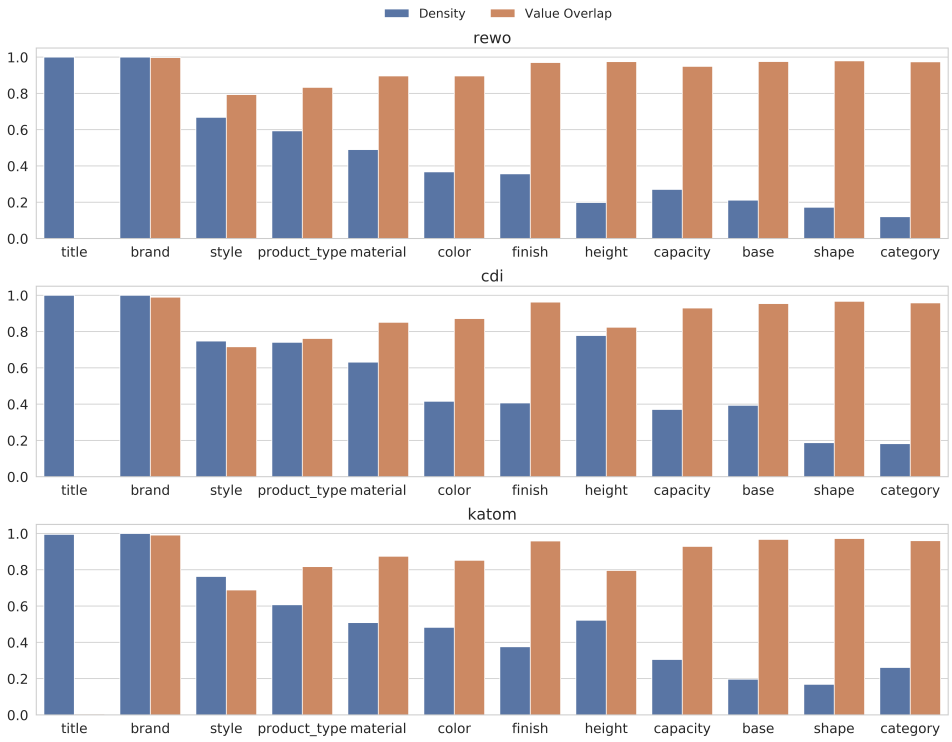


Figure 6.5: Topic Kitchen - Attribute Density and Value Overlap

bold. It is easy to see that only katom_rewo and rewo_cdi share two attributes among their most important relevant attributes. In contrast, katom_cdi and rewo_cdi share only one attribute among the top relevant attributes, and katom_rewo and katom_cdi do not share a single one. However, the schema complexity with nine and ten is quite high across all candidate sets, and the relevant attributes are more or less the same. Only *brand* does not play a role in katom_cdi, but it does for the other two candidate sets. The two attributes *shape* and *category* are not considered relevant attributes in all of them. That is most likely due to the two attributes' low density, as can be seen in Figure 6.8.

The sizes of the candidate sets also differ. Katom_cdi is the smallest with only 1344 record pairs and katom_rewo is the largest with 6457 instances. Rewo_cdi contains a total of 5328 record pairs. Note that matches and mismatches are balanced across all three candidate sets, so the number of both classes is not shown explicitly. The total density of attributes is also not very high for all three candidate sets. However, the most critical information seen in Table 6.4 is the very high proportion of corner cases in all three candidate sets. In katom_rewo, 93% of

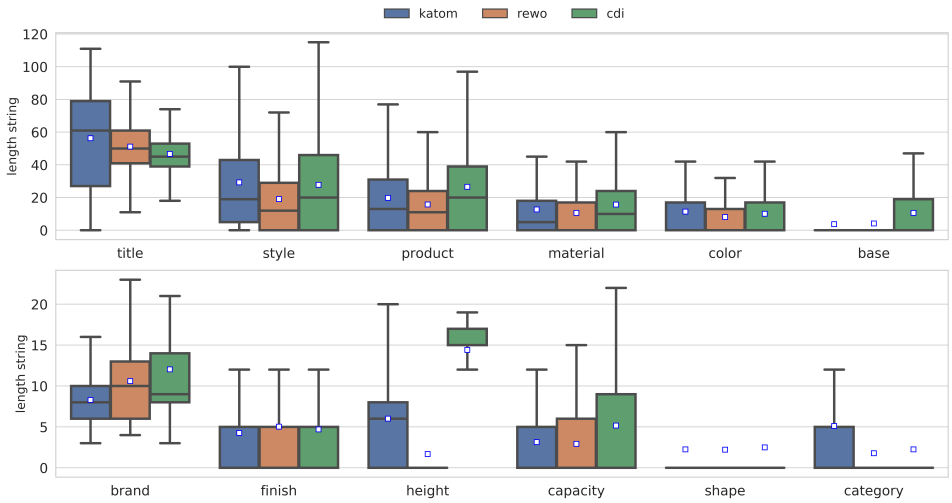


Figure 6.6: Topic Kitchen - Attribute Length

the record pairs are considered corner cases. Therefore, all three matching tasks seem very difficult, which can have a strong influence on the naive transfer performance. For these three candidate sets, there are also differences in the top relevant attributes between katom_cdi and katom_rewo.

If one compares the different distributions of the aggregated similarity values associated with an attribute in Figure 6.7, one can see that katom_rewo has completely different characteristics than the other two, since for each attribute the features have on average a higher similarity value for non-matches than for matches. That is very contrary to the expectation for similarity values. Katom_cdi and rewo_cdi also differ from each other. Nevertheless, unlike katom_rewo, all attribute values in matches have a higher similarity on average than in non-matches, which is usually expected.

### Source-target combination-level

In Table 6.5, the domain relatedness for each source-target combination of the topic kitchen is provided. Since only three candidate sets are available, only three different source-target combinations can be created. Overall all combinations have an MCC above $0.82$. Therefore, it can be assumed that they differ in there marginal distributions, which can be problematic for the naive transfer. Note that the domain relatedness measure does not consider differences in the conditional distributions as shown in Figure 6.7 as the labels are not included (as explained in subsection 5.6.3).

| Set | Relevant Attributes | | Profiling Dimension | | | |
|---|---|---|---|---|---|---|
| | | % all | D | SC | SZ | CC |
| katom_cdi | **height (str), base (str), product_type (str)**, finish (str), capacity (str), material (str), color (str), style (long str), title (long str) | 0.75 | 0.62 | 9 | 1344 | 0.74 |
| katom_rewo | **style (long str), title (long str)**, brand (str), finish (str), capacity (str), material (str), color (str), height (str), base (str), product_type (str) | 0.83 | 0.54 | 10 | 6457 | 0.93 |
| rewo_cdi | **style (long str), title (long str), product_type (str)**, brand (str), finish (str), capacity (str), material (str), color (str), height (str), base (str) | 0.83 | 0.46 | 10 | 5328 | 0.84 |

Table 6.4: Topic Kitchen - Candidate Set Profiling Results

| Combination | MCC per Feature Set | |
|---|---|---|
| | all | dense |
| katom_cdi - rewo_cdi | 0.823 | 0.822 |
| katom_cdi - katom_rewo | 0.829 | 0.835 |
| rewo_cdi - katom_rewo | 0.886 | 0.887 |

Table 6.5: Topic Kitchen - Domain Relatedness for each Source-Target Combination

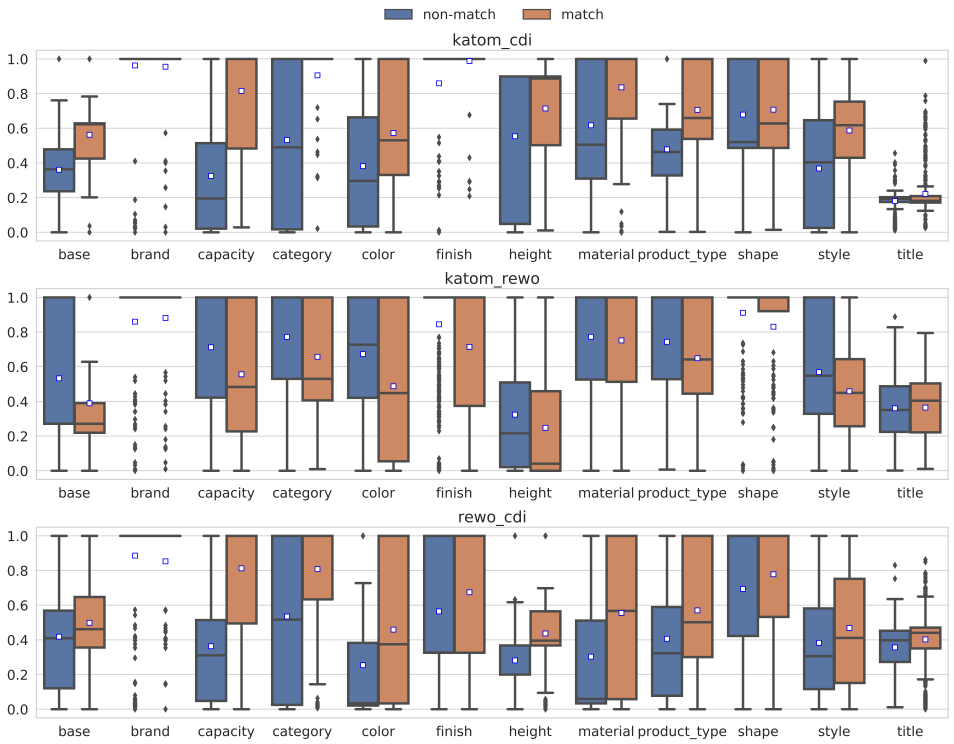


Figure 6.7: Topic Kitchen - Aggregated Sim Scores per Attribute separated by Label

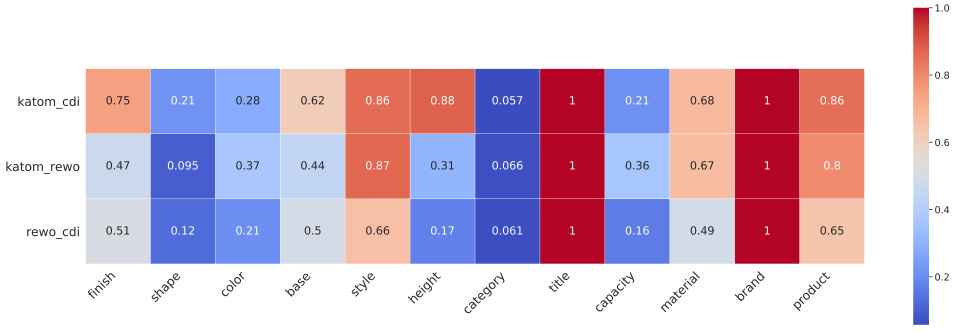


Figure 6.8: Topic Kitchen - Heat map of Attribute Density in all Candidate Sets

## 6.4 Topic: Authors

In this section, the data gathering of the topic authors is briefly explained before the profiling results are presented. The data sources with their respective size and the aligned attributes are listed in Table 6.1.

### 6.4.1 Data Acquisition

The data for the topic authors were used in another work [62] and the test and training sets are publicly available[2]. Therefore, no crawling, preprocessing, or creation of candidate sets was necessary, since the candidate sets were already available. Only the features used in this paper (explained in section 5.2) needed to be created for each record pair.

The four data sources are DBpedia, DNB, VIAF, and Wikidata. DBpedia is a knowledge base (KB) that provides structured information about data from Wikipedia [5]. DNB, which stands for Deutsche Nationalbibliothek, is the central archive library in Germany. It collects, documents, and archives all works concerning Germany or published in German [2]. VIAF, which stands for Virtual International Authority File, allows access to records from many different libraries worldwide. The DNB is also involved in the VIAF project [43]. Wikidata is a free KB where anyone can edit and access the structured data. Most of the data stored in the wiki come from pages like Wikipedia, wikivoyage, and their like [74].

Even though the data for the topic authors stem from four different sources, only three candidate sets are available. According to Primpeli et al. [62] the candidate sets were created using the *owl:samesas* links from DBpedia [62]. Therefore, all three candidate sets contain DBpedia records, and the final candidate sets are dbp_dnb, dbp_viaf, and dbp_wiki.

The only preprocessing performed on the four data sources was the renaming of the common attributes to *birthdate*, *deathdate*, *gender*, *name*, *work*. Additionally, for the source Wikidata, characters from left and right that were not part of the date were removed from the *birthdate* and *deathdate* attributes. In this way, it was ensured that the attribute is handled as an attribute of the type *date*.

### 6.4.2 Profiling Results

In this subsection, the profiling results on the data source-level, candidate set-level, and source-target combination-level for the semantic topic authors are provided.

[2] https://github.com/aprimpeli/UnsupervisedBootAL/tree/master/datasets/author

### Data source-level

The *attribute density* and *value overlap* for each attribute and data source are visualized in Figure 6.9. The attribute *name* has almost no value overlap in all data sources. However, DBPedia is the only source that has absolutely no value overlap for the *name* attribute. The other three have a value overlap close to zero. There are a few duplicate names, but overall, *name* seems to be an attribute that is very important to distinguish between matches and non-matches. In addition, the density of *name* for each data source is $100\%$, except for VIAF, where there are some missing values for *name*. In general, the attribute densities are quite high and similar for all sources, except for two differences. First, the *work* attribute has many missing values in all sources except VIAF. Second, the *deathdate* attribute has a density of only about $50\%$ in all data sources. However, this can also mean that the data sources contain many authors that are still alive, and thus the *deathdate* contains a missing value.

The value overlap for the *gender* attribute is close to $100\%$, which is an expected value. The value overlap is not as high for the other attributes, which have no density close to zero. In particular, VIAF has a small value overlap for the attribute *birthdate*, which could mean that authors from different centuries are included.

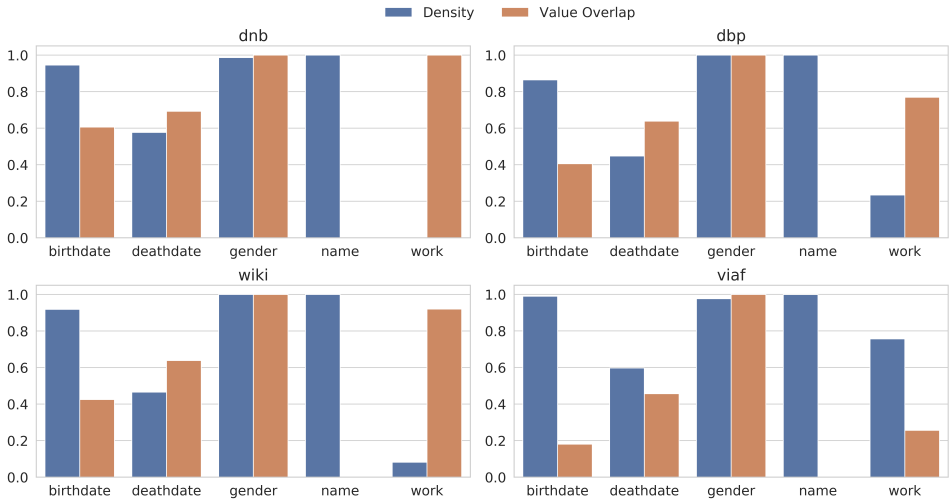


Figure 6.9: Topic Authors - Attribute Density and Value Overlap

In Figure 6.10 the distribution of the length of the values from the attributes is visualized. For the *gender* attribute, there are most likely more male authors in the data sources. At least for Wikidata, DBPedia, and VIAF, the mean value (i.e.,

the tiny blue square) is close to a length of four. Longer words such as *female* or *transgender* therefore seem to be less common. With DNB, there is some noise. It turns out that DNB contains the value *notKnown*, which should indicate a missing value. However, it has been left as it is, so as not to clean up the data.

The *name* attribute has more or less a similar characteristic in all four data sources. For the *birthdate* and *deathdate* attributes, the format appears to be similar when comparing the two attributes for each source. However, there are differences between Wikidata and other sources. Wikidata contains values in the format *1950-01-01 00:00:00* and therefore has longer values. However, this is handled in the same way as values like *1950-01-01*, which can be found in the other sources. The *work* attribute has a lot of missing values (see Figure 6.9) and is not included in the DNB source at all. However, if provided, it contains, on average, very long strings.

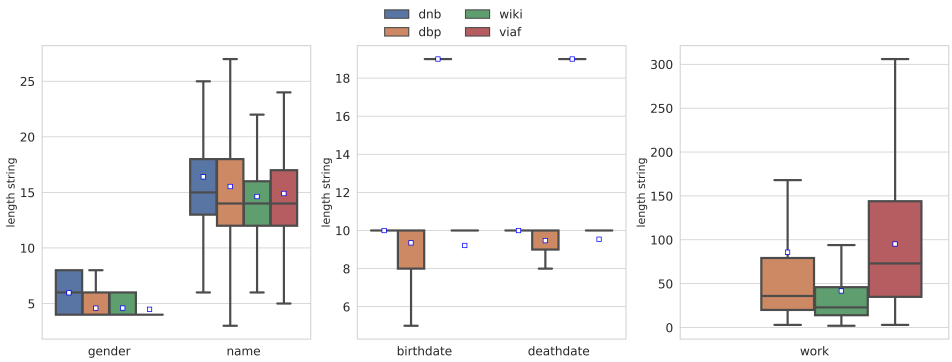


Figure 6.10: Topic Authors - Attribute Length

### Candidate set-level

In Table 6.6, the profiling results are displayed at the candidate set level. As expected, the attribute *name* is considered a top relevant attribute in all three candidate sets due to the fact that it has a very high density and almost no value overlap (see Figure 6.9). For dbp_dnb and dbp_viaf, *birthdate* is also considered a top relevant attribute.

All three candidate sets have a schema complexity of three attributes, which means $60\%$ of all attributes. The attributes *gender* and *work* do not matter. Also, for the candidate set dbp_viaf, the attribute *work* is not considered a relevant attribute, although the density is not as low as in the other two candidate sets (see Figure 6.12).

Apart from that, the candidate sets are all quite large, with more than $15{,}000$ record pairs. Dbp_wiki is the largest with $24{,}035$ instances. It is important to

| Set | Relevant Attributes | % all | Profiling Dimension D | SC | SZ | CC |
|---|---|---|---|---|---|---|
| dbp_dnb | **birthdate (date), name (str)**, deathdate (date) | 0.6 | 0.69 | 3 | 15,809 | 0.34 |
| dbp_viaf | **birthdate (date), name (str)**, deathdate (date) | 0.6 | 0.77 | 3 | 19,144 | 0.27 |
| dbp_wiki | **name (str)**, birthdate (date), deathdate (date) | 0.6 | 0.65 | 3 | 24,035 | 0.08 |

Table 6.6: Topic Authors - Candidate Set Profiling Results

note that the topic author, in contrast to the other two topics, has an unbalanced ratio of matches and non-matches. The proportion of matches is $17\%$ for all three candidate sets for the topic authors. The percentage of corner cases is rather low for all three candidate sets. Dbp_wiki has only $8\%$ corner cases.

When looking at Figure 6.11 it becomes clear why the attribute *name* is considered a highly relevant attribute in all three candidate sets. A clear distinction between matches and non-matches can be seen here. Also, for the *birthdate* attribute, most matches have a similarity value of exactly one, and there are very few outliers that have a similarity value close to one. That is not the case for non-matches. All in all, it seems that the matching tasks of topic authors for supervised learning algorithms are simple.

#### Source-target combination-level

In Table 6.7, the domain relatedness is specified for each source-target combination of topic authors. For the combination of dbp_dnb and dbp_wiki, it makes no difference whether all features or only the dense ones are used, since the attribute densities in both candidate sets are similar (see Figure 6.12). However, for the combinations in which dbp_viaf is involved, it makes a difference. That is because the attribute *work* has a density much higher than zero only in dbp_viaf. Overall, the candidate sets do not appear to be very closely related. Dbp_wiki and dbp_viaf have the lowest MCC for the dense attributes at $0.363$, which is still not close to $0.2$, which would mean that they are related.

Therefore it will be interesting how naive transfer performs. As all candidate sets share the same relevant attributes (see Table 6.6) and also the characteristics of the conditional distribution (see Figure 6.11) are similar among them it still looks promising that a naive transfer can achieve good results.

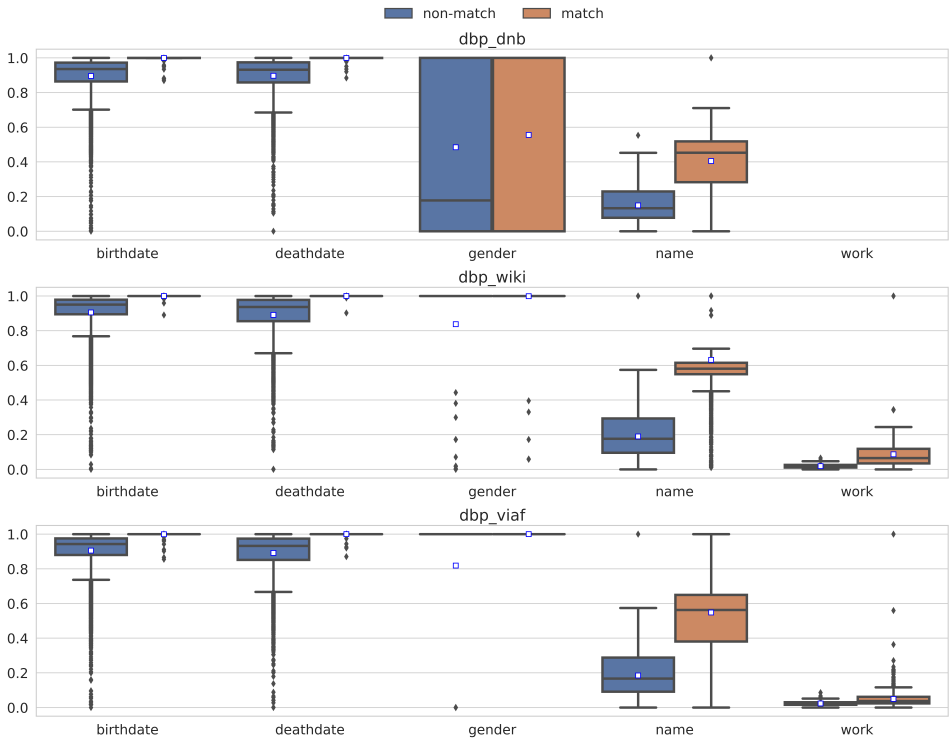


Figure 6.11: Topic Authors - Aggregated Sim Scores per Attribute separated by Label

| **Combination** | **MCC per Feature Set** | |
|---|---|---|
| | all | dense |
| dbp_dnb - dbp_wiki | 0.76 | 0.753 |
| dbp_dnb - dbp_viaf | 0.647 | 0.566 |
| dbp_wiki - dbp_viaf | 0.480 | 0.363 |

Table 6.7: Topic Authors - Domain Relatedness for each Source-Target Combination

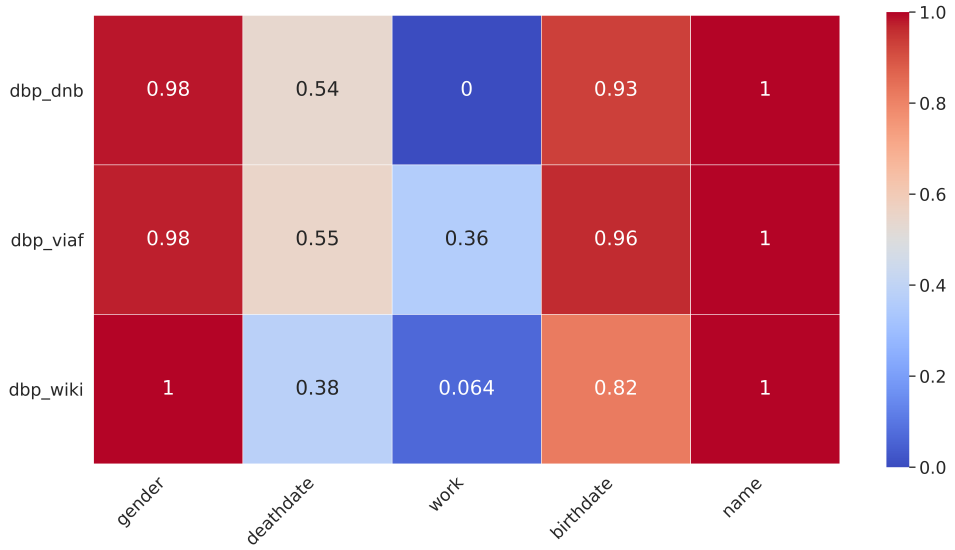


Figure 6.12: Topic Authors - Heat map of Attribute Density in all Candidate Sets

# Chapter 7

# Results

In this chapter, the answers for the research questions of the NTL and ATL experiments for the two semantic topics, books and kitchen, and implications for the final method, which combines TL and ATL for ER, are provided. For that, the results of the experiments are analyzed. The approaches to answer the research questions and an explanation of the result tables are provided before the results are presented. Finally, ATLX's evaluation results on the three semantic topics are provided. In this section, the results are also analyzed in more detail.

## 7.1 NTL Experiments

In this section, the NTL experiment results are linked to each research question for the semantic topics books and kitchen product offers. The approach of how the research questions are answered is explained in the following subsection.

### 7.1.1 Evaluation Method

For each topic, a table with the results of all source-target combinations is presented. With the table, the results of all learning algorithms and baselines for every combination can be seen at once. In Figure 7.1, a section (i.e., not all six experiments and all learning algorithms for the product offers topic are shown) of such a table is shown for explanation purposes. Here each row corresponds to the results of one single experiment and the names of the candidate set acting as source and the candidate set acting as the target are provided. Note that the original data sources involved in either source or target are encoded in the names (f.i. katom_cdi refers to the record pairs of www.katom.com and www.culinarydepotinc.com where the product offers were taken from; more info in chapter 6).

For each different learning algorithm (i.e., called *Estimators* in Figure 7.1), four different results are shown. The first column is TL_avg which provides the naive transfer results. If the number is highlighted in red, a negative transfer happened for that learning algorithm, referring to Baseline 1 (the baselines are explained in subsection 5.4.1). If highlighted in yellow, the learning algorithm achieved the best results among all learning algorithms for this particular source-target combination based on the f1 score.

The other three columns provide an overview of the results of the two remaining Baselines 2 and 3. Tar_max refers to Baseline 2 and is the result of the model trained on the sample with a size of 500 instances from the target training set. Tar_exc indicates how many instances Baseline 2 needed to exceed the naive transfer results. A hyphen indicates that either the naive transfer achieved better results than training on 500 target instances or training on ten target instances has already achieved better results than the naive transfer. For the case that naive transfer is better than training on 500 target instances, TL_avg must also be above TL_max, and the TL_avg results would be highlighted in light green. Tar_sup refers to Baseline 3, the passive learning results, which is the upper bound. For both feature sets, one table each is created.

| | Estimators | logreg | | | | dectree | | | | randforest | | | | xgb | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.650 | 0.694 | **180** | 0.727 | 0.646 | 0.686 | **180** | 0.786 | 0.696 | 0.771 | **140** | 0.822 | 0.675 | 0.765 | **100** | 0.851 |
| rewo_cdi | katom_cdi | 0.427 | 0.798 | - | 0.815 | 0.439 | 0.846 | - | 0.848 | 0.655 | 0.916 | **10** | 0.904 | 0.678 | 0.908 | **20** | 0.917 |
| katom_cdi | katom_rewo | 0.588 | 0.694 | **50** | 0.747 | 0.658 | 0.682 | **180** | 0.768 | 0.672 | 0.759 | **70** | 0.810 | 0.668 | 0.762 | **80** | 0.839 |

Figure 7.1: Section of Naive Transfer Results Table for Explanation

The results are also visualized in plots, to gain more insight into the performance of one single source-target combination. An example plot is shown in Figure 7.2. Here the dotted lines are dedicated to the supervised learning benchmarks and the dashed lines show the target results for different sizes of the training set. The solid lines are the naive transfer results and the line in green with dashes and dots shows the unsupervised matching results.

In order to answer RQ1.1, the number of cells highlighted in yellow is decisive. The best learning algorithm has the most cells highlighted in yellow in Figure 7.1. For RQ1.2, the two tables with results are compared. RQ1.3 cannot fully be explained with the table or plots of the naive transfer. Here one only can see for which combinations a negative transfer happened. The causes of negative transfer are investigated using the profiling dimensions (as described in section 5.6 and the profiling results presented in chapter 6).

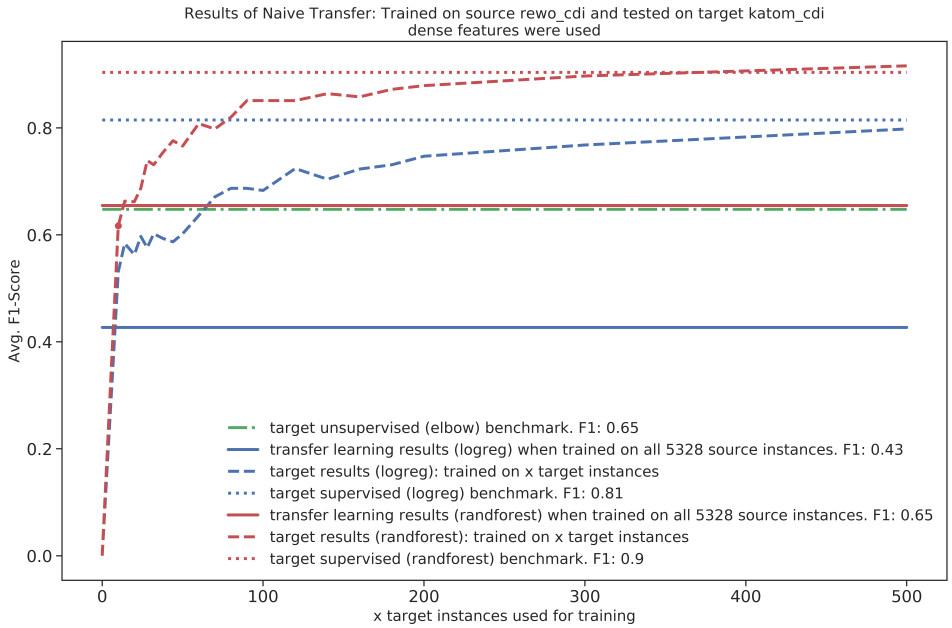


Figure 7.2: Example plot of Naive Transfer Results of two Classifiers for Explanation

### 7.1.2 Research Question 1.1

RQ1.1: Which type of learning algorithm achieves better results for naive transfer?

In the following for each semantic topic, it is analyzed which learning algorithm performs best. The result table with all learning algorithms for the two different feature sets can be found in the appendix section B.2. Due to the size, only sections from the table are shown in this chapter.

#### Topic: Books

For the topic books, the ensemble methods Random Forest and XGBoost exceed the linear models Logistic Regression and Linear SVM as well as a single Decision Tree for the naive transfer results. That was expected since ensemble methods generally achieve better results than single learners.

| | Features | dense | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | logreg | | | | randforest | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | |
| ban_bx | bx_wor | 0.623 | 0.690 | 10 | 0.686 | 0.673 | 0.784 | 14 | 0.818 | 0.703 |
| bx_wor | ban_bx | 0.542 | 0.669 | 10 | 0.655 | 0.758 | 0.834 | 10 | 0.814 | 0.656 |
| ban_bx | ban_half | 0.715 | 0.870 | 10 | 0.889 | 0.951 | 0.960 | 100 | 0.969 | 0.761 |
| ban_half | ban_bx | 0.718 | 0.667 | - | 0.655 | 0.704 | 0.843 | 10 | 0.814 | 0.656 |
| ban_bx | bx_half | 0.580 | 0.734 | 14 | 0.731 | 0.891 | 0.894 | 160 | 0.895 | 0.688 |
| bx_half | ban_bx | 0.580 | 0.672 | 24 | 0.655 | 0.768 | 0.845 | 14 | 0.814 | 0.656 |
| ban_bx | ban_wor | 0.599 | 0.645 | 10 | 0.682 | 0.802 | 0.823 | 140 | 0.830 | 0.679 |
| ban_wor | ban_bx | 0.504 | 0.695 | - | 0.655 | 0.476 | 0.841 | - | 0.814 | 0.656 |
| bx_wor | wor_half | 0.691 | 0.726 | 100 | 0.797 | 0.694 | 0.838 | - | 0.864 | 0.699 |
| wor_half | bx_wor | 0.628 | 0.683 | 24 | 0.686 | 0.533 | 0.790 | - | 0.818 | 0.703 |
| bx_wor | bx_half | 0.755 | 0.731 | - | 0.731 | 0.785 | 0.894 | 10 | 0.895 | 0.688 |
| bx_half | bx_wor | 0.589 | 0.691 | - | 0.686 | 0.740 | 0.791 | 24 | 0.818 | 0.703 |
| bx_wor | ban_wor | 0.656 | 0.636 | - | 0.682 | 0.737 | 0.818 | 20 | 0.830 | 0.679 |
| ban_wor | bx_wor | 0.502 | 0.689 | - | 0.686 | 0.461 | 0.792 | - | 0.818 | 0.703 |
| ban_half | wor_half | 0.756 | 0.751 | - | 0.797 | 0.852 | 0.856 | 300 | 0.864 | 0.699 |
| wor_half | ban_half | 0.837 | 0.869 | 70 | 0.889 | 0.896 | 0.964 | 10 | 0.969 | 0.761 |
| ban_half | bx_half | 0.749 | 0.731 | - | 0.731 | 0.801 | 0.892 | - | 0.895 | 0.688 |
| bx_half | ban_half | 0.648 | 0.869 | - | 0.889 | 0.718 | 0.960 | - | 0.969 | 0.761 |
| ban_half | ban_wor | 0.682 | 0.670 | - | 0.682 | 0.800 | 0.837 | 32 | 0.830 | 0.679 |
| ban_wor | ban_half | 0.658 | 0.869 | - | 0.889 | 0.911 | 0.962 | 14 | 0.969 | 0.761 |
| wor_half | bx_half | 0.759 | 0.736 | - | 0.731 | 0.623 | 0.893 | - | 0.895 | 0.688 |
| bx_half | wor_half | 0.675 | 0.725 | 90 | 0.797 | 0.697 | 0.837 | - | 0.864 | 0.699 |
| wor_half | ban_wor | 0.687 | 0.659 | - | 0.682 | 0.814 | 0.835 | 80 | 0.830 | 0.679 |
| ban_wor | wor_half | 0.676 | 0.747 | 50 | 0.797 | 0.796 | 0.860 | 44 | 0.864 | 0.699 |

Figure 7.3: Topic Books - NTL results of Logistic Regression and Random Forest compared

In Figure 7.3, the results of the Random Forest and the Logistic Regression model are shown. The yellow highlighted cells in the column TL_avg indicate which learning algorithm achieved better results for naive transfer. Logistic Regression performed better for only two combinations. However, when comparing the passive learning results with the naive transfer results, one can see that the Logistic Regression model for six out of 24 combinations even exceeds passive learning with naive transfer (i.e., column Tar_sup highlighted in light green). For Random Forest, only for one combination, naive transfer achieves similar results than passive learning. In Figure 7.3, one can also see that Logistic Regression has

many more cells highlighted in red, which indicate that the unsupervised matching baseline (results in column unsuper_res) did perform better. For Random Forest, eight out of 24 combinations suffer from the negative transfer (i.e., worse than the unsupervised matching baseline) compared to 16 combinations for Logistic Regression.

To summarize, ensemble methods achieve a higher f1 score for naive transfer than all other learning algorithms. However, compared to their passive learning baseline, linear models perform better.

The whole table with the results of all learning algorithms can be found in appendix (all feature: Figure B.2, dense feature: Figure B.3).

### Topic: Kitchen Product Offers

For the topic kitchen product offers, the results are similar to the results of the topic books. Again the ensemble methods achieve better results than single learners. In Figure 7.4 Random Forest is compared to SVM (with linear kernel). Only for one combination out of six SVM performs better than Random Forest. All the linear models achieve comparable results for naive transfer as SVM. Even though the results of Random Forest for the naive transfer are significantly worse than its passive learning benchmark, they are better than unsupervised matching except for one combination. Besides that, as can be seen in Figure 7.4 only for one combination, namely transferring from rewo_cdi to katom_cdi training on ten target instances already achieves better results than naive transfer. For the other combinations, more target instances are required.

| | Features | dense | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | randforest | | | | svm | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.696 | 0.771 | **140** | 0.822 | 0.654 | 0.702 | **160** | 0.727 | **0.652** |
| rewo_cdi | katom_cdi | 0.655 | 0.916 | **10** | 0.904 | 0.378 | 0.841 | - | 0.853 | **0.648** |
| katom_cdi | katom_rewo | 0.672 | 0.759 | **70** | 0.810 | 0.604 | 0.712 | **50** | 0.743 | **0.196** |
| katom_rewo | katom_cdi | 0.706 | 0.913 | **24** | 0.904 | 0.597 | 0.839 | **20** | 0.853 | **0.648** |
| rewo_cdi | katom_rewo | 0.642 | 0.764 | **50** | 0.810 | 0.584 | 0.713 | **32** | 0.743 | **0.196** |
| katom_rewo | rewo_cdi | 0.546 | 0.766 | **14** | 0.822 | 0.660 | 0.705 | **160** | 0.727 | **0.652** |

Figure 7.4: Topic Kitchen - NTL results of SVM and Random Forest compared

To summarize, again, ensemble methods achieve a higher f1 score for naive transfer compared to all other learning algorithms. This time Random Forest achieves better naive transfer results than XGBoost.

The whole table with the results of all learning algorithms can be found in appendix (all feature: Figure B.4, dense feature: Figure B.5).

### 7.1.3 Research Question 1.2

RQ1.2: Do substantial differences in the density of the features across the source and target domain influence the performance of naive transfer?

For each semantic topic, it is analyzed if the results reveal a notable difference when either using all features or excluding features that have a high density in the source but not in the target or vice versa.

#### Topic: Books

In chapter 6, it is mentioned that the data source BX from the topic books does not provide the attribute *pages*. Therefore, it was added as an empty column so that the schemata are aligned across all sources. However, if the source domain record pairs have a high density for the *pages* attribute, it is questionable if naive transfer works very well if the target domain only contains missing values. Because of this, the NTL experiments were conducted on two different feature sets. The first set denoted as *all* contains all features. The second set denoted as *dense* only contains the features that are associated with the attributes that share more or less the same densities across the source and target domain. The densities of the attributes for each data source is shown in Figure 6.4.

In Figure 7.5 the results with *all* and *dense* features for all source-target combinations that contain the data source BX are shown. The first six rows show the combinations when the source BX can be found in the target domain. It is easy to spot that excluding the non-dense features strongly influences the naive transfer results. For two combinations, this leads to an increase in the f1 score from more than $0.34$ (shown in the column $\Delta TL\_avg$). Even though most of them still suffer from negative transfer, the performance is significantly better. The results also indicate that it does not matter the other way around. That was expected because the learning algorithm ignores the features that only contain missing values when learning a classification model.

#### Topic: Kitchen Product Offers

The topic of kitchen product offers does not include attributes with such an extreme difference in density across the data sources. It was still tested if excluding attributes with less than $3\%$ density affects the naive transfer performance. As can be seen in Figure 7.6 no significant changes in the results can be found.

| | Features | all | | | | dense | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | randforest | | | | randforest | | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res | Δ TL_avg |
| Source | Target | | | | | | | | | | |
| ban_half | ban_bx | 0.547 | 0.842 | - | 0.814 | 0.704 | 0.843 | 10 | 0.814 | 0.656 | 0.157 |
| ban_wor | ban_bx | 0.134 | 0.844 | - | 0.814 | 0.476 | 0.841 | - | 0.814 | 0.656 | 0.342 |
| wor_half | bx_wor | 0.391 | 0.788 | - | 0.818 | 0.533 | 0.790 | - | 0.818 | 0.703 | 0.142 |
| ban_wor | bx_wor | 0.105 | 0.788 | - | 0.818 | 0.461 | 0.792 | - | 0.818 | 0.703 | 0.356 |
| ban_half | bx_half | 0.590 | 0.896 | - | 0.895 | 0.801 | 0.892 | - | 0.895 | 0.688 | 0.211 |
| wor_half | bx_half | 0.391 | 0.891 | - | 0.895 | 0.623 | 0.893 | - | 0.895 | 0.688 | 0.232 |
| ban_bx | ban_half | 0.939 | 0.962 | 38 | 0.969 | 0.951 | 0.960 | 100 | 0.969 | 0.761 | 0.012 |
| bx_wor | wor_half | 0.701 | 0.860 | - | 0.864 | 0.694 | 0.838 | - | 0.864 | 0.699 | -0.007 |
| ban_bx | ban_wor | 0.802 | 0.836 | 50 | 0.830 | 0.802 | 0.823 | 140 | 0.830 | 0.679 | 0.0 |
| bx_half | ban_half | 0.715 | 0.960 | - | 0.969 | 0.718 | 0.960 | - | 0.969 | 0.761 | 0.003 |
| bx_wor | ban_wor | 0.724 | 0.837 | 10 | 0.830 | 0.737 | 0.818 | 20 | 0.830 | 0.679 | 0.013 |
| bx_half | wor_half | 0.698 | 0.865 | 10 | 0.864 | 0.697 | 0.837 | - | 0.864 | 0.699 | -0.001 |

Figure 7.5: Topic Books - NTL results of Random Forest using all Features or only dense Features

### 7.1.4 Research Question 1.3

RQ1.3: For which combinations does naive transfer perform worse than unsupervised matching and what are potential causes for this?

In this subsection, the naive transfer results are analyzed in more detail. For that, the profiling results of each source-target combination are examined to understand potential reasons for the negative transfer.

#### Topic: Books

The negative transfer, which, in this thesis, is defined as naive transfer results below the unsupervised matching benchmark, differs across the learning algorithms. The Random Forest and the XGBoost classifiers are the two best performing one for the topic books. Therefore, in further analysis, more emphasis is put on these two ensemble methods. Besides that, the results of the *dense* features are considered, as this has shown to lead to better results. For Random Forest eight combinations lead to negative transfer and for XGBoost five out of the 24 combinations, as can be seen in Figure 7.7. In Figure 7.7 the column $\Delta TL\_avg - Unsup$ shows the difference of the naive transfer results to the unsupervised benchmark results (column $unsuper\_res$). The worst result of naive transfer happens when training on ban_wor and apply on bx_wor. Here, Random Forest results are significantly

| | Features | all | | | | dense | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | randforest | | | | randforest | | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res | Δ TL_avg |
| Source | Target | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.697 | 0.775 | 120 | 0.822 | 0.696 | 0.771 | 140 | 0.822 | 0.652 | -0.001 |
| rewo_cdi | katom_cdi | 0.641 | 0.920 | 10 | 0.904 | 0.655 | 0.916 | 10 | 0.904 | 0.648 | 0.014 |
| katom_cdi | katom_rewo | 0.677 | 0.758 | 90 | 0.810 | 0.672 | 0.759 | 70 | 0.810 | 0.196 | -0.005 |
| katom_rewo | katom_cdi | 0.695 | 0.920 | 28 | 0.904 | 0.706 | 0.913 | 24 | 0.904 | 0.648 | 0.011 |
| rewo_cdi | katom_rewo | 0.639 | 0.757 | 38 | 0.810 | 0.642 | 0.764 | 50 | 0.810 | 0.196 | 0.003 |
| katom_rewo | rewo_cdi | 0.551 | 0.773 | 14 | 0.822 | 0.546 | 0.766 | 14 | 0.822 | 0.652 | -0.005 |

Figure 7.6: Topic Kitchen - NTL results of Random Forest using all Features or only dense Features

worse than unsupervised matching with around 24 percentage points less than unsupervised matching. The results of XGBoost and all other learning algorithms are all worse than unsupervised matching for this source-target combination (see Figure B.3 for all results). For the candidate set profiling results from ban_wor and bx_wor, three observations can be made.

First of all, differences in the top relevant attributes exist. Both candidate sets share *pubdate* as top relevant attribute. However, ban_wor additionally considers *pages* as top relevant attribute (see Table 6.2), which in the candidate set bx_wor is not even considered as BX only contains missing values for *pages*. Bx_wor, on the other hand, considers *publisher* as a top relevant attribute. Even though the two candidate sets differ in their top relevant attributes, when looking at the relevant attribute (which includes top relevant attributes), both share the same attributes except *pages*. Therefore, their SC is five for ban_wor and four for bx_wor.

A further difference that can be spotted in Table 6.2 is the difference in the average density of all attributes and the difference in SZ. Bx_wor is the most comprehensive candidate set of the topic books with 18,540 instances opposed to 1940 instances in ban_wor. Both have a CC higher than 50%. However, a difference that most probably has a higher effect on naive transfer performance is the difference in the distributions of the features (i.e., the marginal distributions). In Figure 6.3, the boxplots of the aggregated features per attribute separated per label indicate great differences. That is the case for all attributes between ban_wor and bx_wor. It is also reflected in the domain relatedness of the two candidate sets. Here an MCC of 0.347 indicates that the Logistic Regression model learned found differences in both candidate sets' data, as the MCC is much better than random guessing (i.e., an MCC close to zero indicates random guessing).

It is reasonable to assume that especially a difference between the top relevant attribute *pubdate* in both domains, has the highest influence on the negative

| | Features | dense | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | randforest | | | | xgb | | | | |
| | Results | TL_avg | Δ TL_avg - Unsup | Tar_exc | Tar_sup | TL_avg | Δ TL_avg - Unsup | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | |
| ban_bx | bx_wor | 0.673 | -0.03 | 14 | 0.818 | 0.683 | -0.02 | 10 | 0.856 | 0.703 |
| bx_wor | ban_bx | 0.758 | 0.102 | 10 | 0.814 | 0.759 | 0.103 | 24 | 0.872 | 0.656 |
| ban_bx | ban_half | 0.951 | 0.19 | 100 | 0.969 | 0.924 | 0.163 | 14 | 0.960 | 0.761 |
| ban_half | ban_bx | 0.704 | 0.048 | 10 | 0.814 | 0.733 | 0.077 | 10 | 0.872 | 0.656 |
| ban_bx | bx_half | 0.891 | 0.203 | 160 | 0.895 | 0.860 | 0.172 | 10 | 0.894 | 0.688 |
| bx_half | ban_bx | 0.768 | 0.112 | 14 | 0.814 | 0.796 | 0.14 | 60 | 0.872 | 0.656 |
| ban_bx | ban_wor | 0.802 | 0.123 | 140 | 0.830 | 0.781 | 0.102 | 70 | 0.846 | 0.679 |
| ban_wor | ban_bx | 0.476 | -0.18 | - | 0.814 | 0.521 | -0.135 | - | 0.872 | 0.656 |
| bx_wor | wor_half | 0.694 | -0.005 | - | 0.864 | 0.714 | 0.015 | 10 | 0.896 | 0.699 |
| wor_half | bx_wor | 0.533 | -0.17 | - | 0.818 | 0.555 | -0.148 | - | 0.856 | 0.703 |
| bx_wor | bx_half | 0.785 | 0.097 | 10 | 0.895 | 0.803 | 0.115 | 10 | 0.894 | 0.688 |
| bx_half | bx_wor | 0.740 | 0.037 | 24 | 0.818 | 0.739 | 0.036 | 44 | 0.856 | 0.703 |
| bx_wor | ban_wor | 0.737 | 0.058 | 20 | 0.830 | 0.751 | 0.072 | 10 | 0.846 | 0.679 |
| ban_wor | bx_wor | 0.461 | -0.242 | - | 0.818 | 0.510 | -0.193 | - | 0.856 | 0.703 |
| ban_half | wor_half | 0.852 | 0.153 | 300 | 0.864 | 0.847 | 0.148 | 200 | 0.896 | 0.699 |
| wor_half | ban_half | 0.896 | 0.135 | 10 | 0.969 | 0.908 | 0.147 | 14 | 0.960 | 0.761 |
| ban_half | bx_half | 0.801 | 0.113 | - | 0.895 | 0.806 | 0.118 | 10 | 0.894 | 0.688 |
| bx_half | ban_half | 0.718 | -0.043 | - | 0.969 | 0.870 | 0.109 | 10 | 0.960 | 0.761 |
| ban_half | ban_wor | 0.800 | 0.121 | 32 | 0.830 | 0.782 | 0.103 | 24 | 0.846 | 0.679 |
| ban_wor | ban_half | 0.911 | 0.15 | 14 | 0.969 | 0.904 | 0.143 | 10 | 0.960 | 0.761 |
| wor_half | bx_half | 0.623 | -0.065 | - | 0.895 | 0.604 | -0.084 | - | 0.894 | 0.688 |
| bx_half | wor_half | 0.697 | -0.002 | - | 0.864 | 0.796 | 0.097 | 100 | 0.896 | 0.699 |
| wor_half | ban_wor | 0.814 | 0.135 | 80 | 0.830 | 0.825 | 0.146 | 300 | 0.846 | 0.679 |
| ban_wor | wor_half | 0.796 | 0.097 | 44 | 0.864 | 0.804 | 0.105 | 70 | 0.896 | 0.699 |

Figure 7.7: Topic Books - NTL results of Random Forest and XGBoost compared to Unsupervised Matching

transfer results of naive transfer. In Figure 6.3 one can spot much more outliers with low similarity scores (also for matches) in the *pubdate* attribute for bx_wor than for ban_wor. To make the difference of the *pubdate* attribute more clear, in Figure 7.8, the empirical cumulative distribution function (ECDF) of the feature *pubdate_days_diff_sim*, is displayed. Here the difference between the two domains can be seen easily as the two ECDFs differ a lot. Note that for the remaining two features of the attribute *pubdate*, namely *pubdate_months_diff_sim* and *pubdate_years_diff_sim*, it follows the exactly same pattern (i.e., the features are of course highly correlated with each other).

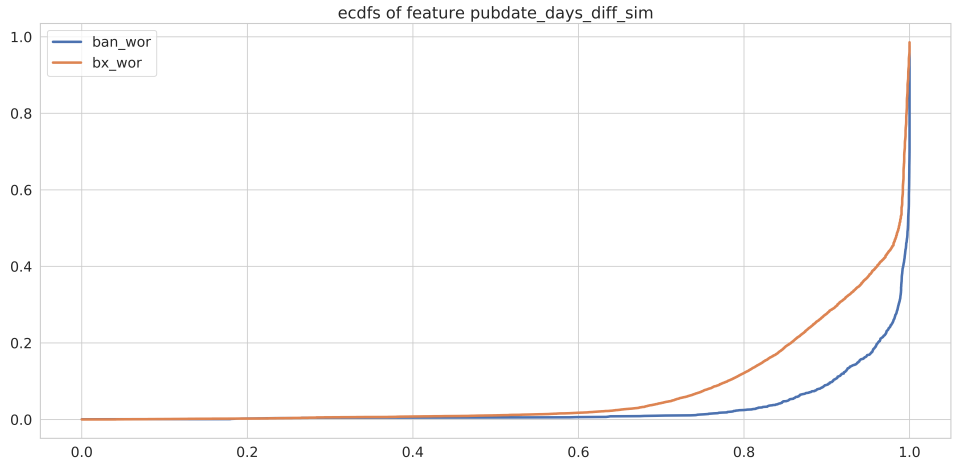


Figure 7.8: Topic Books - ECDF of the Feature *pubdate_days_diff_sim* for ban_wor and bx_wor

Nevertheless, a question that still arises is why the transfer works better the other way around. When training on bx_wor and apply on ban_wor the naive transfer works significantly better (see Figure 7.7). The naive transfer results are still not very good when comparing it with the other baselines, but they are better than unsupervised matching in this direction. This phenomenon can often be seen in the results. A potential explanation for the better results of naive transfer from bx_wor to ban_wor than the opposite direction could be that bx_wor has much more labeled instances with $18{,}540$ instances opposed to $1940$ instances in ban_wor.

Two more source-target combinations lead to naive transfer results significantly worse than unsupervised matching (i.e., more than $10$ percentage points below). One of them is the transfer from wor_half to bx_wor and the other one is ban_wor to ban_bx (see Figure 7.7). For the former, again, the candidate set bx_wor is the target domain. The differences in the profiling dimensions between both candidate sets are similar to those between ban_wor and bx_wor. Both candidate sets share *pubdate* and *publisher* as top relevant attribute but wor_half additionally has *pages* as top relevant attribute (see Table 6.2). Again this is excluded in the naive transfer on the *dense* features due to BX. Additionally, both candidate sets have a high CC with around $60\%$.

For wor_half and bx_wor again differences between the attributes can be seen in Figure 6.3. This time the attributes *author*, *title*, and *pubdate* show significant differences. However, the attribute *publisher*, which is considered a top relevant attribute in both domains, shows the same characteristics in both candidate sets. The estimated domain relatedness between both domains is with an MCC of $0.435$

again quite low, as the higher the MCC, the lower the estimated domain relatedness.

Besides that, once more for these two candidate sets naive transfer from bx_wor to wor_half works better than in the other direction, though still slightly worse than unsupervised matching for Random Forest (see Figure 7.7).

For the naive transfer from ban_wor to ban_bx, which is also significantly worse than unsupervised matching in the target domain ban_bx, also differences in the common top relevant attribute *pubdate* can be found (see Figure 7.9 or Figure 6.3). That could again be a possible cause for negative transfer. All in all, the low estimated domain relatedness between these two candidate sets, an MCC of $0.493$, indicates significant differences in both domains' marginal distributions.

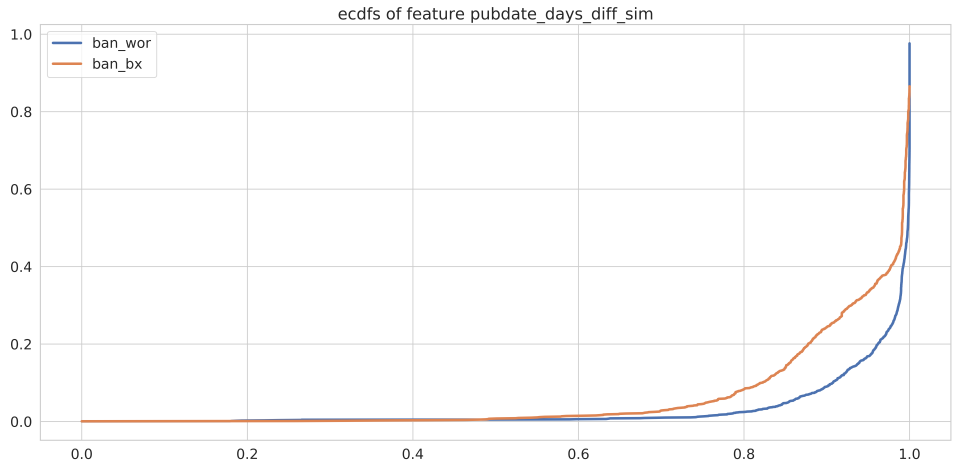


Figure 7.9: Topic Books - ECDF of the Feature *pubdate_days_diff_sim* for ban_wor and ban_bx

However, it does not explain why the naive transfer from ban_bx to ban_wor (i.e., the opposite direction) does work quite well with a difference in f1 score of more than $0.3$ (see Figure 7.7). Additionally, for this direction of transfer around $140$ target instances are required for the Random Forest classifier to achieve similar results than naive transfer (see column Tar_exc, which is Baseline 2, in Figure 7.7), and the naive transfer comes close to the supervised learning benchmark (i.e., column Tar_sup). Thus, just one look at the profiling results of the entire candidate set of ban_bx might be a bit short-sighted to find causes for the negative transfer. It could be that in the test set from ban_bx, more corner cases are included that challenge the classifier learned on ban_wor. In fact, when profiling the ban_bx test set, a CC of $0.64$ compared to $0.34$ in the entire candidate set is the result, partly because *publisher* is also considered a highly relevant attribute in the ban_bx test

set (see Figure B.1).

As can be seen in Figure 6.3 for the whole candidate sets the mean of the aggregated features from *publisher* are different for ban_bx and ban_wor. While for ban_wor the mean value of the positive cases for the *publisher* attribute similarities is higher than for the non-matches, for ban_bx the opposite is true. That could be a potential cause for the negative transfer when training a classifier on ban_wor and test on the test set from ban_bx. It also explains the different results when transferring in the opposite direction.

For the topic books, there are four more source-target combinations for Random Forest and two more for XGBoost where unsupervised matching performs better than naive transfer. For Random Forest wor_half to bx_half, bx_half to wor_half, bx_half to ban_half, and ban_bx to bx_wor have slightly worse results for naive transfer than unsupervised matching in the corresponding target domain. For XGBoost only wor_half to bx_half and ban_bx to bx_wor suffer from negative transfer (see Figure 7.7). All of them, except ban_bx and bx_wor, have an MCC higher than $0.3$ (see Table 6.3), which indicates a lower estimated domain relatedness and they also have differences in their relevant attributes (see Table 6.2). In contrast, ban_bx and bx_wor have an MCC of $0.098$, which indicates that there are no significant differences in the marginal distributions and, thus, the domains are rather related to each other. Also, their relevant attributes match. However, *publisher* is a highly relevant attribute in bx_wor and here, as mentioned above, the instances in ban_bx have on average higher similarity values for non-matches than for matches. The opposite, however, is the case for bx_wor.

Apart from ban_bx and bx_wor, a higher domain relatedness often indicates lower chances of a negative transfer. The source-target combinations ban_half and wor_half and wor_half and ban_wor both have an MCC below $0.2$ for the dense features, which indicates higher domain relatedness (see Table 6.3). These combinations also show positive results in both directions for the tree-based ensemble methods (see Figure 7.7).

#### Topic: Kitchen Product Offers

For the topic of kitchen product offers, when looking at the naive transfer results for the best performing classifier (i.e., Random Forest), only one source-target combination, katom_rewo to rewo_cdi, has significantly worse results than the unsupervised baseline (see Figure 7.10).

| | Features | dense | | | | |
|---|---|---|---|---|---|---|
| | Estimators | randforest | | | | |
| | Results | TL_avg | Δ TL_avg - Unsup | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | |
| katom_cdi | rewo_cdi | 0.696 | **0.044** | **140** | 0.822 | **0.652** |
| rewo_cdi | katom_cdi | 0.655 | **0.007** | **10** | 0.904 | **0.648** |
| katom_cdi | katom_rewo | 0.672 | **0.476** | **70** | 0.810 | **0.196** |
| katom_rewo | katom_cdi | 0.706 | **0.058** | **24** | 0.904 | **0.648** |
| rewo_cdi | katom_rewo | 0.642 | **0.446** | **50** | 0.810 | **0.196** |
| katom_rewo | rewo_cdi | 0.546 | **-0.106** | **14** | 0.822 | **0.652** |

Figure 7.10: Topic Kitchen - NTL results of Random Forest compared to Unsupervised Matching

The problem with katom_rewo can be seen in Figure 6.7. For almost all attributes, the aggregated similarity scores are higher for the non-matches than for the matches. That is also the reason for the poor unsupervised matching results of katom_rewo with an f1 score of only 0.196 (see column unsuper_res when katom_rewo is the target in Figure 7.10). For the top relevant attributes of katom_rewo, *style* and *title*, this is also reflected in a very high CC of 0.93 (see Table 6.4).

Therefore, even though rewo_cdi and katom_rewo share two top relevant attributes, a negative transfer was expected due to the significant differences in the conditional distributions, amplified by the fact that differences exist for all relevant attributes that are the same in both candidate sets (see Table 6.4). Additionally, katom_rewo and rewo_cdi also have the lowest domain relatedness of all three source-target combinations. Nonetheless, a low estimated domain relatedness is the case for all of them (see Table 6.5). Note that the conditional distributions are not considered but only the marginal distributions in the domain relatedness estimation. Besides that, all candidate sets have a high CC. Katom_cdi has the lowest CC with 0.74. An even worse negative transfer between katom_rewo and rewo_cdi would have been expected if the CC of rewo_cdi were much lower.

Compared to the unsupervised baseline, the naive transfer results for the remaining source-target combinations are not bad. Besides that, when looking at how many target instances are approximately needed to achieve similar results than naive transfer (shown in column Tar_exc in Figure 7.10), for half of the source-target combinations 50 or more target instances are required. Nevertheless, the naive transfer results are far below the supervised learning results when learning on the target domain directly (column Tar_sup in Figure 7.10). However, that was

expected. First of all, there is a low domain relatedness between all candidate sets (Table 6.5). Secondly, they all have a large portion of corner cases, and there are differences in the top relevant attributes of katom_cdi and rewo_cdi (see Table 6.4).

### 7.1.5 Summary and Implications

The NTL experiments have shown that ensemble methods achieve better results than all single learners for naive transfer when comparing the f1 score from naive transfer among all learning algorithms. The ensemble methods, therefore, more often also perform better than unsupervised matching. However, the ensemble methods less often come close to their supervised learning benchmark. That does not mean ensemble methods work better for NTL in general, but only that they more often achieve better results than unsupervised matching.

The NTL experiments also indicate that substantial differences in the density of the features across the source and target domain influence the performance of naive transfer. If the source domain contains one attribute with only little missing values (i.e., high attribute density), which, however, has only missing values in the target domain, excluding the features leads to better results. Therefore, in the ATL experiments, features with almost only missing values are excluded (denoted as *using only dense features*).

Causes for negative transfer are manifold. It is well-known from the literature that the causes can be traced back to differences in the marginal and conditional distributions between source and target domain. That is also reflected by the NTL experiments conducted in this work.

Based on the profiling dimensions, the NTL experiments have shown that differences in the top relevant attributes between source and target domain can have a negative effect.

In the problem setting considered in this thesis, it is assumed that no labeled data is available in the target domain. Therefore, only differences in the marginal distributions between both domains can be estimated, which is done with the domain relatedness measure introduced in subsection 5.6.3. In this setting, it needs to be assumed that the conditional distributions are the same between the source and target domain within one semantic topic. Data profiling has shown that this assumption is sometimes violated. Especially for the semantic topic kitchen significant differences exist (see Figure 6.7). However, the results have shown that naive transfer can still compete with unsupervised matching.

Nevertheless, naively incorporating knowledge from the source domain can lead to worse results compared to unsupervised matching. Therefore, techniques that mitigate the risk of negative transfer have to be considered. Still, NTL has the potential to help with the cold-start problem in AL.

The NTL results have also shown that the naive transfer results are often far from the supervised learning in the target benchmark, at least for the best-performing learning algorithms. Therefore, AL seems appropriate because acquiring labeled target domain data could close the gap between the naive transfer results and supervised results.

In the following ATL experiments, only Random Forest is considered as the AL model based on the excellent naive transfer performance in both semantic topics.

## 7.2 ATL Experiments

Most of the result tables can be found in section B.3. The research questions are only answered based on the difference of f1 score between ATL and the baselines. In the evaluation of the final method, a more precise comparison will be conducted. The evaluation method will be explained in the next subsection.

### 7.2.1 Evaluation Method

For the ATL experiments, a similar table like the one for the naive transfer results is created. That again provides an overview of all source-target combinations' performance at once compared to the baselines. As can be seen in Figure 7.11 for each source-target combination, the results for the committee-based query strategy (i.e., lr_lsvc_rf_dt) is shown next to the results of the random sampling baseline. The results are provided for the 2nd, 10th, 20th, 30th, 50th, and the final iteration with 100 target instances queried. For each iteration, the results of the ATL experiment and the AL baseline are provided. In the final column of each query strategy, the passive learning benchmark is provided. If the ATL results are highlighted in orange, then the random sampling results for ATL exceeded those of the committee-based query strategy for that iteration. If the AL results are highlighted in red, they exceeded the ATL results with the number of target instances queried. In total, three tables per topic are analyzed. That is one table with the results of no domain adaptation and one table for each domain adaptation technique.

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | random | | | | | | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | | |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | | |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.692 | 0.000 | 0.692 | 0.519 | 0.681 | 0.534 | 0.694 | 0.562 | 0.699 | 0.623 | 0.717 | 0.692 | **0.822** | 0.689 | 0.000 | 0.689 | 0.461 | 0.687 | 0.601 | 0.694 | 0.584 | 0.697 | 0.593 | 0.707 | 0.682 | **0.822** | | |
| rewo_cdi | katom_cdi | 0.695 | 0.000 | 0.677 | 0.551 | 0.725 | 0.675 | 0.748 | 0.706 | 0.787 | 0.744 | 0.822 | 0.846 | **0.904** | 0.692 | 0.000 | 0.723 | 0.568 | 0.722 | 0.663 | 0.777 | 0.735 | 0.801 | 0.778 | 0.820 | 0.820 | **0.904** | | |

Figure 7.11: Section of ATL Results Table for Explanation

The tables shall provide an overview of all source-target combinations at once;

this is especially important for the book topic with 24 different source-target combinations. Besides that, the tables help to answer RQ2.1 and RQ2.2, as can be seen for which combination the query strategy is superior to random sampling and ATL achieves better results than AL.

In order to answer RQ2.3, an additional table with the results is used. In this table, the different domain adaptation techniques are visualized next to each other (see Figure 7.12 for four different iterations. The column *ATL* contains the ATL results without weighting, *_NN* refers to the ATL results with NN, whereas *_LP* refers to the ATL results with lr_predict_proba used as domain adaptation technique. Here again, different color coding is used. Green is used to visualize the best results per iteration among the three different approaches, and yellow indicates the best results per row, which excludes the passive learning benchmark results of the target domain in column *Tar_sup*.

| | Estimator | rf | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | | | |
| | Iteration | 2nd | | | | 20th | | | | 50th | | | | 100th | | | | all |
| | Results | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.692 | 0.683 | 0.678 | 0.000 | 0.681 | 0.698 | 0.684 | 0.534 | 0.699 | 0.700 | 0.695 | 0.623 | 0.717 | 0.727 | 0.709 | 0.692 | **0.822** |
| rewo_cdi | katom_cdi | 0.695 | 0.690 | 0.750 | 0.000 | 0.725 | 0.758 | 0.694 | 0.675 | 0.787 | 0.827 | 0.754 | 0.744 | 0.822 | 0.853 | 0.826 | 0.846 | **0.904** |
| katom_cdi | katom_rewo | 0.675 | 0.674 | 0.664 | 0.000 | 0.695 | 0.704 | 0.679 | 0.567 | 0.713 | 0.717 | 0.700 | 0.655 | 0.718 | 0.737 | 0.716 | 0.697 | **0.81** |

Figure 7.12: Section of ATL Results Table DA Comparison for Explanation

The results are also plotted to compare the domain adaptation techniques for a single source-target combination with more detail. This is especially useful to see whether there is an increase in performance once more target instances are queried or if the source data is dominating the learning model throughout the complete AL process. An example plot is shown in Figure 7.13. This plot shows the ATL results with NN, the AL baseline, and the passive learning benchmark as a dotted line. By representing the standard deviation in the plots using error bars, the variability of the results can be compared.

If interesting patterns were identified using the two different tables, the plots were used to analyze further.

### 7.2.2 Research Question 2.1

RQ2.1: Does the query strategy perform better than random sampling?

In this subsection, the results are analyzed to answer whether incorporating the labeled source domain data into the labeled set helps the query strategy select the

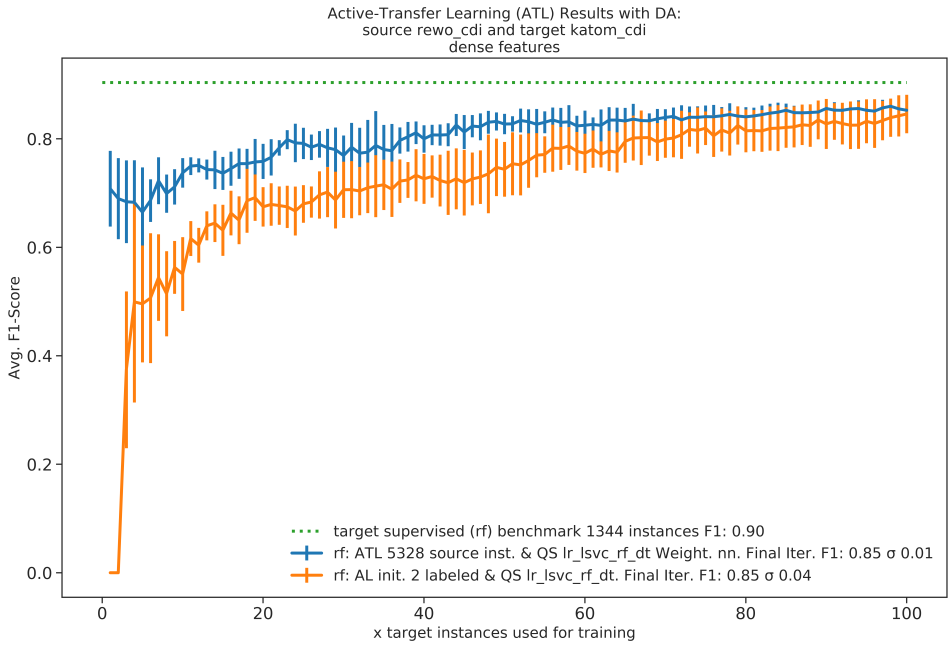


Figure 7.13: ATL Results Plot One Source-Target Combination With DA and Errorbars for Explanation

most informative record pairs. If the results of random sampling exceed the results with the query strategy, it can be assumed that the labeled source domain data added to the labeled set misleads the query strategy.

### Topic: Books

The result tables are added to the appendix due to the size (see section B.3). Without domain adaptation, for 16 out of 24 combinations in the final iteration (i.e., 100 target instances queried) the f1 score of random sampling exceed the f1 score of the query strategy or is less than $0.01$ worse (see ATL results highlighted in orange in Figure B.7). That is also the case for 18 combinations when NN is applied as the domain adaptation technique (see Figure B.8). LP does lead to better performance than random sampling, as only 12 combinations are affected (see Figure B.9).

### Topic: Kitchen Product Offers

For the topic kitchen, ATL results are better or similar to random sampling in the final iteration when no domain adaptation is applied. Only for two out of six

source-target combinations, the ATL results are worse than random sampling (see highlighted in orange Figure B.10). For the two domain adaptation techniques, the results of ATL are slightly worse when compared to random sampling (see Figure B.11 and Figure B.12).

### 7.2.3 Research Question 2.2

RQ2.2: How does this simple active learning setting with transfer learning perform compared to the active learning baseline?

The AL results highlighted in red in the result tables in section B.3 indicate that the f1 score of the AL baseline exceeds the f1 score of ATL in this iteration. If AL performs better, including the whole source domain data into the labeled set may hinder the active learning model from adapting to the target domain data.

#### Topic: Books

For the topic books for more than half of the source-target combinations, AL achieves better results in the final iteration. This is also the case when using the domain adaptation techniques (see Figure B.8 and Figure B.9).

#### Topic: Kitchen Product Offers

ATL performs better for the topic kitchen than the AL baseline for at least half of the source-target combinations.

### 7.2.4 Research Question 2.3

RQ2.3: Leads Unsupervised Domain Adaptation to better performance for source-target combinations that suffer from negative transfer?

The research question shall answer whether weighting source domain instances that are more similar to the target domain more heavily can increase the performance of the source-target combinations where naive transfer leads to poor results.

#### Topic: Books

In Figure 7.14 the results of ATL without domain adaptation (denoted as *ATL*), with nearest-neighbor based weighting (denoted as *_NN*), and with Logistic Regression based weighting (denoted as *_LP*) are visualized next to each other. Only the source-target combinations, which suffered from negative transfer in the NTL

experiments, are shown. In Figure B.6 the results of all source-target combinations are provided. For the combination ban_bx as the source domain and bx_wor as tar-

| | Estimator | rf | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | | | |
| | Iteration | 2nd | | | | 20th | | | | 50th | | | | 100th | | | | all |
| | Results | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.675 | 0.692 | 0.674 | 0.000 | 0.699 | 0.716 | 0.712 | 0.752 | 0.751 | 0.753 | 0.756 | 0.740 | 0.760 | 0.764 | 0.762 | 0.746 | **0.818** |
| ban_wor | ban_bx | 0.488 | 0.462 | 0.473 | 0.000 | 0.504 | 0.542 | 0.516 | 0.652 | 0.565 | 0.597 | 0.575 | 0.791 | 0.619 | 0.639 | 0.632 | 0.830 | **0.814** |
| bx_wor | wor_half | 0.698 | 0.701 | 0.692 | 0.000 | 0.709 | 0.711 | 0.701 | 0.823 | 0.723 | 0.735 | 0.725 | 0.830 | 0.748 | 0.765 | 0.758 | 0.807 | **0.864** |
| wor_half | bx_wor | 0.531 | 0.548 | 0.537 | 0.000 | 0.567 | 0.610 | 0.571 | 0.752 | 0.613 | 0.647 | 0.626 | 0.740 | 0.662 | 0.685 | 0.671 | 0.746 | **0.818** |
| ban_wor | bx_wor | 0.456 | 0.454 | 0.455 | 0.000 | 0.498 | 0.510 | 0.486 | 0.752 | 0.547 | 0.539 | 0.531 | 0.740 | 0.587 | 0.589 | 0.581 | 0.746 | **0.818** |
| bx_half | ban_half | 0.747 | 0.841 | 0.861 | 0.000 | 0.883 | 0.908 | 0.899 | 0.925 | 0.907 | 0.922 | 0.917 | 0.947 | 0.930 | 0.937 | 0.940 | 0.953 | **0.969** |
| wor_half | bx_half | 0.624 | 0.648 | 0.625 | 0.000 | 0.740 | 0.777 | 0.730 | 0.838 | 0.790 | 0.806 | 0.799 | 0.853 | 0.821 | 0.827 | 0.838 | 0.869 | **0.895** |
| bx_half | wor_half | 0.725 | 0.764 | 0.737 | 0.000 | 0.752 | 0.781 | 0.764 | 0.823 | 0.778 | 0.792 | 0.781 | 0.830 | 0.794 | 0.801 | 0.789 | 0.807 | **0.864** |

Figure 7.14: Topic Books - ATL results of Domain Adaptation Techniques compared (only subset)

get domain, the domain adaptation techniques do not make much of a difference. That was expected because the estimated domain relatedness is rather high between both domains (see Table 6.3). For ban_wor to bx_wor, the domain adaptation techniques do not increase the performance significantly even though the two domains are less related to each other than ban_bx and bx_wor. However, apart from these two cases, the domain adaptation techniques, especially NN, increase the f1 score for the remaining ones, which suffered from negative transfer in the NTL experiments. Nevertheless, except ban_bx to bx_wor, they all achieve worse results than the AL baseline. Some of them even achieve significantly worse results with an f1 score of more than $15\%$ percentage points below the one from the AL baseline.

### Topic: Kitchen Product Offers

In Figure 7.15, the results of all ATL experiments of the topic kitchen are shown. Recap that all candidate sets are rather unrelated to each other in the topic kitchen (see Table 6.5). Therefore, the domain adaptation techniques, especially NN, help to increase the performance. Apart from the combination katom_rewo as the source domain and rewo_cdi as the target domain, all combinations achieve better results than the AL baseline. However, they are far away from the passive learning benchmark.

| | Estimator | rf | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | | | | |
| | Iteration | 2nd | | | | 20th | | | | 50th | | | | 100th | | | | all |
| | Results | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.692 | 0.683 | 0.678 | 0.000 | 0.681 | 0.698 | 0.684 | 0.534 | 0.699 | 0.700 | 0.695 | 0.623 | 0.717 | 0.727 | 0.709 | 0.692 | **0.822** |
| rewo_cdi | katom_cdi | 0.695 | 0.690 | 0.750 | 0.000 | 0.725 | 0.758 | 0.694 | 0.675 | 0.787 | 0.827 | 0.754 | 0.744 | 0.822 | 0.853 | 0.826 | 0.846 | **0.904** |
| katom_cdi | katom_rewo | 0.675 | 0.674 | 0.664 | 0.000 | 0.695 | 0.704 | 0.679 | 0.567 | 0.713 | 0.717 | 0.700 | 0.655 | 0.718 | 0.737 | 0.716 | 0.697 | **0.81** |
| katom_rewo | katom_cdi | 0.693 | 0.678 | 0.668 | 0.000 | 0.727 | 0.731 | 0.748 | 0.675 | 0.769 | 0.812 | 0.810 | 0.744 | 0.833 | 0.840 | 0.852 | 0.846 | **0.904** |
| rewo_cdi | katom_rewo | 0.642 | 0.643 | 0.644 | 0.000 | 0.667 | 0.670 | 0.666 | 0.567 | 0.689 | 0.682 | 0.688 | 0.655 | 0.716 | 0.722 | 0.711 | 0.697 | **0.81** |
| katom_rewo | rewo_cdi | 0.555 | 0.560 | 0.539 | 0.000 | 0.573 | 0.578 | 0.569 | 0.534 | 0.608 | 0.620 | 0.612 | 0.623 | 0.647 | 0.667 | 0.656 | 0.692 | **0.822** |

Figure 7.15: Topic Kitchen - ATL results Domain Adaptation Techniques compared

### 7.2.5 Summary and Implications

The ATL experiments have shown that merely adding all labeled source domain data to the labeled set often leads to worse results than random sampling concerning the f1 score. Also, domain adaptation techniques do not contribute to better results compared to random sampling.

Apart from this, the AL model often leads to a worse f1 score than the AL baseline f1 score, even if less than 50 target instances are queried. Although domain adaptation techniques can slightly increase the f1 score, the AL baseline, which does not contain source-domain knowledge, still gives better results for the source-target combinations that suffer from negative transfer.

The results, even if only the f1 scores are compared to the baselines, indicate that the mere inclusion of all labeled source domain data in the labeled set not only misleads the query strategy committee but also prevents the active learning model itself from adapting to the target domain. A possible cause is a significant imbalance between the source and target domain data in the labeled set. Note that if there are more than a thousand labeled source domain instances in the labeled set, the labeled target domain data acquired up to a maximum quota of only 100 instances has little impact on the learned model. In addition, if the source domain data is not representative of the target domain data, it cannot be expected that this ATL setting will produce better results than AL.

Therefore, the implications are that the active learning model of the final ATL method must give more influence to the labeled source domain data at the beginning if there is still too little labeled target domain data. However, the labeled source domain data's influence must decrease again if more labeled target domain data has been acquired. Besides, the query strategy of the final ATL method should also be allowed to focus more on the target domain data. Here the ATL experiments

have shown that weighting the labeled source domain data through the domain adaptation techniques is insufficient. Thus not training the committee members on any labeled source domain data could be more beneficial.

## 7.3 ATLX

Before the evaluation results of the final method compared to its baselines are provided, the evaluation method is explained in detail.

### 7.3.1 Evaluation Method

The same approach, as for the two initial experiments, is used to compare the results of ATLX with the two baselines. That means for each source-target combination, the f1 scores of a few iterations and the baseline results are displayed in a result table. If the results of ATLX are highlighted in orange, Baseline 1 achieved better results for that iteration (i.e., either better or less than a difference of $1\%$). If the results of Baseline 2 are highlighted in red, they exceed the results of ATLX or are less than $1\%$ worse. If the ATLX results are highlighted in yellow, they exceed the results of Baseline 3. Additionally, to analyze the method more precisely, the following model parameters and performance indicators are measured. Some of them provide insight into how the active learning model evolves from right after bootstrapping until the convergence point.

*Correctness of bootstrapping sample*. In ATLX, the labeled set is bootstrapped with one positive and one negative record pair from the target domain (the size of the labeled set after bootstrapping is provided in column *ls* in Figure 7.16). As the labels are predicted based on a classification model learned on the labeled source domain data, it is not guaranteed that the labels are correct. For each class, the portion of wrongly predicted labels is measured ($n\_+$ for positive class and $n\_-$ for negative class in Figure 7.16). A value of zero means the predicted labels were correct, and one means the predicted labels were incorrect. Noise in the bootstrapping sample could be a potential cause of worse performance than Baseline 2.

*Share of corrected labels*. The portion of queried target instances where the initial classification model, which was learned only on the source domain data, predicted the wrong label (denoted as *cor* in Figure 7.16). A high value means that the query strategy successfully selects instances that provide new knowledge which helps the active learning model adapt to the target domain.

*Difference of f1 score at start and end.* It quantifies the performance increase achieved with the target instances queried ($\Delta\_out\_f1$ in Figure 7.16). It is the difference between column *f1_out_0* (f1 score at start) and *f1_out_-1* (f1 score at end).

*Difference of f1 score final iteration to Baselines.* The difference of f1 score between ATLX and Baseline 1 (i.e., Random Sampling, denoted as $\Delta\_R\_f1$ in Figure 7.16), Baseline 2 (i.e., AL Unsupervised Bootstrap; $\Delta\_AL\_f1$), and Baseline 3 (i.e., Passive Learning; $\Delta\_pas\_f1$). The results in column $\Delta\_R\_f1$ and $\Delta\_AL\_f1$ are highlighted in green if ATLX exceeds Baseline 1 and 2 with more than $0.01$, respectively. If the results in column $\Delta\_pas\_f1$ are highlighted in magenta, ATLX exceeds the passive learning benachmark with only 100 target instances queried.

*Standard deviation of f1 score at final iteration.* It quantifies the variability of the results. Each experiment is conducted with five runs to account for the randomness introduced when selecting one instance from the record pairs with the committee's highest disagreement. The lower the standard deviation, the more robust is the final active learning model (denoted as *sig_out_-1* in Figure 7.16). The standard deviation is also provided for the results of the last iteration for Baseline 1 and 2.

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source | Target | DA | | | | | | | | | | | | | |
| dbp_dnb | dbp_wiki | - | 2 | 0 | 0 | 0.320 | 0.942 | 0.990 | 0.048 | 0.000 | 0.043 | -0.004 | 0.001 | 0.002 | 0.041 |
| | | nn | 2 | 0 | 0 | 0.260 | 0.955 | 0.990 | 0.035 | 0.000 | 0.026 | -0.004 | 0.001 | 0.002 | 0.024 |
| | | lp | 2 | 0 | 0 | 0.290 | 0.962 | 0.991 | 0.029 | 0.001 | 0.023 | -0.003 | 0.001 | 0.002 | 0.011 |

Figure 7.16: ATLX Section of Table with Performance Indicators for Explanation

*Confidence of classification model at start and end.* This measure quantifies how confident the AL model is with its predictions on the test set right after bootstrapping and after the final iteration (denoted as *c_0* for start and *c_-1* for final iteration in Figure 7.17). The absolute difference of the predicted probabilities from the positive class to the threshold of $0.5$ is measured and averaged. If the value is close to zero, the model is somewhat less confident with its predictions as they are (on average!) close to the decision boundary. However, if the value is close to $0.5$ it means that the model is more confident with its predictions.

*Average Depth of Trees at start and end.* This measure provides the trees' average depth in the Random Forest at the first iteration and the final iteration (*adt_0* for start and *adt_-1* for end in Figure 7.17). It shows how the Random Forest evolves.

*Importance of attributes at start and end.* Feature importance is measured right after bootstrapping and after the convergence point. The attributes that are related to the features are listed ordered based on their importance (*ordered_attr_import* in Figure 7.17).

| | | | adt_0 | adt_-1 | c_0 | c_-1 | ordered_attr_import_0 | ordered_attr_import_-1 |
|---|---|---|---|---|---|---|---|---|
| Source | Target | DA | | | | | | |
| ban_bx | bx_wor | - | 24.300 | 7.670 | 0.341 | 0.225 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | nn | 23.700 | 7.205 | 0.344 | 0.248 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | lp | 23.700 | 7.981 | 0.351 | 0.226 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |

Figure 7.17: ATLX Section of Table with Change of AL Model for Explanation

### 7.3.2 Evaluation Results

In the following, ATLX's evaluation results for each semantic topic are presented. The method and its baselines are applied to all possible source-target combinations, the same as for the two initial experiments. As the bootstrapping sample size, the most conservative approach is used (i.e., the minimal amount of two instances, one positive and one negative).

#### Topic: Books

In Figure 7.18, the performance indicators (as explained in section 5.5) of the ATLX method, with a bootstrap sample size of one positive and one negative instance and no domain adaptation technique applied, are listed. Compared to Baseline 1 (i.e., Random Sampling; denoted as R in Figure 7.18) ATLX achieves worse results in the last iteration for only four out of 24 combinations (highlighted in red in column $\Delta\_R\_f1$ in Figure 7.18). These combinations all have bx_wor as candidate set, either as source or target, in common. The profiling of bx_wor has shown that it is by far the largest candidate set within the topic books and has a high share of corner cases (see Table 6.2). However, for the estimated domain relatedness of the combinations where Baseline 1 outperforms ATLX, there is no clear trend that a lower domain relatedness between source and target could be a potential cause of random sampling performing better (see Table 6.3 for the domain relatedness estimations). For these combinations the bootstrapping sample also does not contain any false positives or false negatives (see $n\_+$ and $n\_-$ in Figure 7.18).

Nevertheless, in Figure 7.19, the results of these four source-target combinations when domain adaptation is applied are listed. Apart from the combination of ban_bx as source and bx_wor as the target, the three remaining combinations show

that applying LP (see section 4.3.1) as domain adaptation technique can significantly increase the performance. For ban_bx and bx_wor LP does not increase the performance, but NN (see section 4.3.1).

Even though for these four combinations ATLX performs worse than random sampling when no domain adaptation is applied, for half of the experiments ATLX without domain adaptation outperforms random sampling with a higher f1 score in the last iteration of at least 0.01 (highlighted in green in $\Delta\_R\_f1$ in Figure 7.18). For the remaining eight combinations, the averaged results of ATLX in the final iteration are the same or only slightly better than Baseline 1. The variability (standard deviation denoted as *sig_...* in Figure 7.18) of the results of the last iteration for these combinations are comparable between ATLX and Baseline 1.

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source | Target | DA | | | | | | | | | | | | | |
| ban_bx | bx_wor | - | 2 | 0 | 0 | 0.410 | 0.674 | 0.752 | 0.078 | -0.009 | -0.020 | -0.066 | 0.030 | 0.011 | 0.008 |
| bx_wor | ban_bx | - | 2 | 0 | 0 | 0.450 | 0.730 | 0.837 | 0.107 | 0.039 | 0.009 | 0.023 | 0.017 | 0.034 | 0.008 |
| ban_bx | ban_half | - | 2 | 0 | 0 | 0.310 | 0.948 | 0.959 | 0.011 | 0.003 | 0.020 | -0.010 | 0.003 | 0.007 | 0.009 |
| ban_half | ban_bx | - | 2 | 0 | 0 | 0.420 | 0.670 | 0.833 | 0.163 | 0.035 | 0.002 | 0.019 | 0.015 | 0.034 | 0.017 |
| ban_bx | bx_half | - | 2 | 0 | 0 | 0.220 | 0.867 | 0.896 | 0.029 | 0.020 | 0.021 | 0.001 | 0.004 | 0.008 | 0.022 |
| bx_half | ban_bx | - | 2 | 0 | 1 | 0.330 | 0.728 | 0.830 | 0.102 | 0.032 | 0.033 | 0.016 | 0.009 | 0.034 | 0.025 |
| ban_bx | ban_wor | - | 2 | 0 | 0 | 0.410 | 0.797 | 0.799 | 0.002 | 0.022 | 0.001 | -0.031 | 0.030 | 0.035 | 0.034 |
| ban_wor | ban_bx | - | 2 | 0 | 1 | 0.500 | 0.462 | 0.821 | 0.359 | 0.023 | 0.000 | 0.007 | 0.011 | 0.034 | 0.017 |
| bx_wor | wor_half | - | 2 | 0 | 0 | 0.450 | 0.703 | 0.825 | 0.122 | 0.000 | 0.012 | -0.039 | 0.012 | 0.006 | 0.012 |
| wor_half | bx_wor | - | 2 | 0 | 0 | 0.470 | 0.509 | 0.769 | 0.260 | 0.008 | 0.008 | -0.049 | 0.020 | 0.011 | 0.025 |
| bx_wor | bx_half | - | 2 | 0 | 0 | 0.400 | 0.780 | 0.884 | 0.104 | 0.008 | -0.001 | -0.011 | 0.018 | 0.008 | 0.010 |
| bx_half | bx_wor | - | 2 | 0 | 0 | 0.340 | 0.721 | 0.747 | 0.026 | -0.014 | -0.023 | -0.071 | 0.025 | 0.011 | 0.011 |
| bx_wor | ban_wor | - | 2 | 0 | 0 | 0.450 | 0.747 | 0.797 | 0.050 | 0.020 | -0.003 | -0.033 | 0.020 | 0.035 | 0.016 |
| ban_wor | bx_wor | - | 2 | 0 | 0 | 0.450 | 0.420 | 0.749 | 0.329 | -0.012 | 0.020 | -0.069 | 0.026 | 0.011 | 0.042 |
| ban_half | wor_half | - | 2 | 0 | 0 | 0.270 | 0.842 | 0.839 | -0.003 | 0.014 | 0.038 | -0.025 | 0.009 | 0.006 | 0.030 |
| wor_half | ban_half | - | 2 | 0 | 0 | 0.320 | 0.869 | 0.955 | 0.086 | -0.001 | 0.010 | -0.014 | 0.007 | 0.007 | 0.010 |
| ban_half | bx_half | - | 2 | 0 | 0 | 0.420 | 0.750 | 0.894 | 0.144 | 0.018 | 0.030 | -0.001 | 0.009 | 0.008 | 0.039 |
| bx_half | ban_half | - | 2 | 0 | 0 | 0.440 | 0.736 | 0.963 | 0.227 | 0.007 | 0.005 | -0.006 | 0.004 | 0.007 | 0.003 |
| ban_half | ban_wor | - | 2 | 0 | 0 | 0.350 | 0.796 | 0.825 | 0.029 | 0.048 | 0.011 | -0.005 | 0.008 | 0.035 | 0.011 |
| ban_wor | ban_half | - | 2 | 0 | 0 | 0.440 | 0.881 | 0.961 | 0.080 | 0.005 | 0.014 | -0.008 | 0.003 | 0.007 | 0.015 |
| wor_half | bx_half | - | 2 | 0 | 0 | 0.560 | 0.580 | 0.885 | 0.305 | 0.009 | 0.023 | -0.010 | 0.011 | 0.008 | 0.024 |
| bx_half | wor_half | - | 2 | 0 | 0 | 0.430 | 0.693 | 0.811 | 0.118 | -0.014 | 0.009 | -0.053 | 0.022 | 0.006 | 0.022 |
| wor_half | ban_wor | - | 2 | 0 | 0 | 0.330 | 0.799 | 0.825 | 0.026 | 0.048 | 0.033 | -0.005 | 0.019 | 0.035 | 0.027 |
| ban_wor | wor_half | - | 2 | 0 | 0 | 0.340 | 0.772 | 0.839 | 0.067 | 0.014 | 0.009 | -0.025 | 0.013 | 0.006 | 0.010 |

Figure 7.18: Topic Books - ATLX results without DA

With regards to Baseline 2 (i.e., AL with Unsupervised Bootstrapping, denoted as AL in Figure 7.18) ATLX without any domain adaptation achieves better (i.e., at least 0.01 higher f1 score; highlighted in green) results for 12 out of 24 combina-

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source | Target | DA | | | | | | | | | | | | | |
| ban_bx | bx_wor | - | 2 | 0 | 0 | 0.410 | 0.674 | 0.752 | 0.078 | -0.009 | -0.020 | -0.066 | 0.030 | 0.011 | 0.008 |
| | | nn | 2 | 0 | 0 | 0.490 | 0.689 | 0.774 | 0.085 | 0.013 | 0.015 | -0.044 | 0.014 | 0.011 | 0.020 |
| | | lp | 2 | 0 | 0 | 0.410 | 0.661 | 0.750 | 0.089 | -0.011 | -0.009 | -0.068 | 0.034 | 0.011 | 0.047 |
| bx_wor | bx_half | - | 2 | 0 | 0 | 0.400 | 0.780 | 0.884 | 0.104 | 0.008 | -0.001 | -0.011 | 0.018 | 0.008 | 0.010 |
| | | nn | 2 | 0 | 0 | 0.530 | 0.793 | 0.900 | 0.107 | 0.024 | -0.004 | 0.005 | 0.007 | 0.008 | 0.008 |
| | | lp | 2 | 0 | 0 | 0.460 | 0.781 | 0.900 | 0.119 | 0.024 | 0.025 | 0.005 | 0.010 | 0.008 | 0.024 |
| bx_half | bx_wor | - | 2 | 0 | 0 | 0.340 | 0.721 | 0.747 | 0.026 | -0.014 | -0.023 | -0.071 | 0.025 | 0.011 | 0.011 |
| | | nn | 2 | 0 | 0 | 0.370 | 0.722 | 0.749 | 0.027 | -0.012 | 0.009 | -0.069 | 0.053 | 0.011 | 0.042 |
| | | lp | 2 | 0 | 0 | 0.340 | 0.725 | 0.781 | 0.056 | 0.020 | 0.034 | -0.037 | 0.010 | 0.011 | 0.027 |
| bx_wor | ban_wor | - | 2 | 0 | 0 | 0.450 | 0.747 | 0.797 | 0.050 | 0.020 | -0.003 | -0.033 | 0.020 | 0.035 | 0.016 |
| | | nn | 2 | 1 | 0 | 0.470 | 0.719 | 0.811 | 0.092 | 0.034 | 0.007 | -0.019 | 0.010 | 0.035 | 0.012 |
| | | lp | 2 | 0 | 0 | 0.420 | 0.732 | 0.817 | 0.085 | 0.040 | 0.022 | -0.013 | 0.012 | 0.035 | 0.045 |

Figure 7.19: Topic Books - ATLX DAs compared (Baseline 1 better than ATLX)

tions. For five out of 24 combinations, the results of ATLX in the last iteration are below Baseline 2. For these combinations, the results are also, most of the time, less robust in the last iteration compared to Baseline 2.

For the remaining seven combinations, the results of ATLX are comparable to Baseline 2 (i.e., the f1 score of ATLX is less than $0.01$ above Baseline 2). In Figure B.13 and Figure B.14 the results of all combinations are listed also with the two domain adaptation techniques. There, one can see that either NN or LP does help achieve better results than Baseline 2 for the five combinations where the results of ATLX are below Baseline 2. Still, one can also see that the domain adaptation techniques for other combinations lead to worse results than Baseline 2. Besides that, it is not always the same domain adaptation technique that leads to better results.

For five from the 12 combinations where ATLX is much better than Baseline 2, ATLX achieves better results than Baseline 3 (i.e., passive learning benchmark; highlighted in magenta in Figure 7.18). Surprisingly, this is even the case for two combinations, bx_half and ban_bx as well as ban_wor and ban_bx, which were bootstrapped with a wrongly classified negative record pair (values in column $n\_$- highlighted in red). Most of the remaining combinations can significantly improve performance with only $100$ target instances queried, except for two combinations (marked in red in the $\Delta\_out\_f1$ column) that show an increase of less than $0.01$ or even a decrease in the f1 score from the last iteration and the first iteration. These two combinations, however, both exceed Baseline 1 and 2.

The result tables of how the AL model evolves from start until the last target instance queried with regards to the average depth of trees, the confidence of

predictions, and the importance of the attributes are listed in Figure B.15 and Figure B.16. Here the average depth of trees is very high right after bootstrapping but drops significantly after 100 target instances are queried for all source-target combinations. The confidence in the predicted probabilities is also lower after the last iteration for most of the combinations. No matter if a domain adaptation technique is applied or not for the same source-target combination, the same ordered importance of the attributes is the result after 100 target instances are queried.

**Topic: Kitchen Product Offers**

In Figure 7.20, the performance indicators of the ATLX method applied to the data from the topic kitchen, with a bootstrap sample size of one positive and one negative instance without (denoted as "-" in column *DA*) and with domain adaptation technique applied, are listed. Compared to Baseline 1 (i.e., random sampling), ATLX without DA achieves worse results for only one out of six combinations. This combination has katom_rewo as the target domain and rewo_cdi as the source. The profiling of the topic kitchen's candidate sets has shown that these two candidate sets are only very little related to each other as a Logistic Regression model can almost perfectly differentiate between both domains (see Table 6.5). Furthermore, the conditional distributions are entirely different between both candidate sets. Recap that katom_rewo has, on average, higher similarities for the attribute values of non-matches than for matches, which is not at all typical for ER and also not the case for the remaining candidate sets (see Figure 6.7). Therefore, Baseline 2 also performs significantly worse than ATLX for this combination because the unsupervised matching, which is used to bootstrap AL, is creating a lot of noise for the target domain katom_rewo. Recap that the unsupervised matching technique applied is based on the assumption that everything higher than the threshold is considered a match and everything below a non-match.

Nevertheless, the performance of ATLX without DA for this combination is also not very satisfying as the f1 score even decreases slightly with the 100 target instances added to the labeled set. Though, for this combination, LP as DA does lead to significantly better results. However, if katom_rewo is used as the source domain and rewo_cdi as the target domain, this is not the case. For this transfer direction, ATLX without DA works better and even exceeds Baseline 1 and 2, whereas both DAs lead to worse results than the baselines. When looking at the results of the combinations which involve katom_rewo, one can see that sometimes DA can help to increase the performance compared to Baseline 1 or Baseline 2 but not always. The reason for that is that the unsupervised DAs, LP and NN, only assign more weight on source domain instances that are more similar to the target domain without looking at the labels because they are unknown. Therefore,

the substantial differences in the conditional distributions of katom_rewo to the other domains can potentially cause even more damage if more weight is given to similar instances (based on the marginal distribution) even though the actual labels of these record pairs are different across the domains. For the remaining two combinations, which are also very little related to each other based on the marginal distributions (see Table 6.5), LP or NN as DA improves or at least does not decrease the performance of ATLX without DA.

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | | | | | | | | |
| **katom_cdi** | **rewo_cdi** | **-** | 2 | 0 | 0 | 0.390 | 0.630 | 0.691 | 0.061 | 0.042 | 0.033 | -0.131 | 0.019 | 0.062 | 0.066 |
| | | **nn** | 2 | 0 | 0 | 0.360 | 0.623 | 0.677 | 0.054 | 0.028 | -0.002 | -0.145 | 0.035 | 0.062 | 0.018 |
| | | **lp** | 2 | 0 | 0 | 0.440 | 0.660 | 0.690 | 0.030 | 0.041 | 0.035 | -0.132 | 0.020 | 0.062 | 0.038 |
| **rewo_cdi** | **katom_cdi** | **-** | 2 | 0 | 0 | 0.460 | 0.607 | 0.802 | 0.195 | -0.003 | 0.047 | -0.102 | 0.057 | 0.031 | 0.044 |
| | | **nn** | 2 | 0 | 0 | 0.450 | 0.573 | 0.840 | 0.267 | 0.035 | 0.056 | -0.064 | 0.038 | 0.031 | 0.020 |
| | | **lp** | 2 | 1 | 0 | 0.420 | 0.703 | 0.813 | 0.110 | 0.008 | 0.049 | -0.091 | 0.028 | 0.031 | 0.087 |
| **katom_cdi** | **katom_rewo** | **-** | 2 | 0 | 0 | 0.520 | 0.637 | 0.652 | 0.015 | 0.084 | 0.061 | -0.158 | 0.081 | 0.082 | 0.080 |
| | | **nn** | 2 | 0 | 0 | 0.460 | 0.655 | 0.632 | -0.023 | 0.064 | 0.008 | -0.178 | 0.046 | 0.082 | 0.050 |
| | | **lp** | 2 | 0 | 0 | 0.480 | 0.623 | 0.650 | 0.027 | 0.082 | -0.040 | -0.160 | 0.072 | 0.082 | 0.027 |
| **katom_rewo** | **katom_cdi** | **-** | 2 | 0 | 0 | 0.480 | 0.741 | 0.823 | 0.082 | 0.018 | 0.017 | -0.081 | 0.025 | 0.031 | 0.030 |
| | | **nn** | 2 | 0 | 0 | 0.500 | 0.638 | 0.793 | 0.155 | -0.012 | 0.004 | -0.111 | 0.057 | 0.031 | 0.074 |
| | | **lp** | 2 | 0 | 0 | 0.500 | 0.672 | 0.831 | 0.159 | 0.026 | 0.074 | -0.073 | 0.037 | 0.031 | 0.063 |
| **rewo_cdi** | **katom_rewo** | **-** | 2 | 0 | 0 | 0.520 | 0.623 | 0.616 | -0.007 | 0.048 | -0.027 | -0.194 | 0.066 | 0.082 | 0.052 |
| | | **nn** | 2 | 0 | 1 | 0.460 | 0.627 | 0.631 | 0.004 | 0.063 | -0.013 | -0.179 | 0.076 | 0.082 | 0.066 |
| | | **lp** | 2 | 0 | 0 | 0.540 | 0.637 | 0.681 | 0.044 | 0.113 | 0.006 | -0.129 | 0.037 | 0.082 | 0.017 |
| **katom_rewo** | **rewo_cdi** | **-** | 2 | 0 | 1 | 0.490 | 0.490 | 0.657 | 0.167 | 0.008 | 0.032 | -0.165 | 0.043 | 0.062 | 0.048 |
| | | **nn** | 2 | 0 | 1 | 0.500 | 0.506 | 0.612 | 0.106 | -0.037 | -0.033 | -0.210 | 0.126 | 0.062 | 0.056 |
| | | **lp** | 2 | 0 | 1 | 0.560 | 0.480 | 0.628 | 0.148 | -0.021 | -0.030 | -0.194 | 0.030 | 0.062 | 0.032 |

Figure 7.20: Topic Kitchen - ATLX results with and without DA

With regards to Baseline 2 (i.e., AL with Unsupervised Bootstrapping) ATLX without any domain adaptation achieves slightly worse results for only one combination, rewo_cdi as the source domain and katom_cdi as the target domain. For the remaining five combinations the results of ATLX are significantly better than Baseline 2 with regards to f1 score of the last iteration as well as variability of the performance (see *sig_out* and *sig_AL* in Figure 7.20).

However, with only 100 target instances, ATLX never manages to achieve similar results than Baseline 3 (i.e., passive learning benchmark). Nevertheless, it was expected for this rather complicated topic because the NTL experiments have shown that for half of the combinations, more than 50 target instances are roughly needed even to achieve better results than naive transfer (see Figure 7.10). Therefore, it cannot be expected that only 100 target instances are sufficient to reach the passive learning benchmark, which, as shown in the NTL experiments, is always

significantly better than the naive transfer results Figure 7.10. However, the share of corrected labels within the quota (i.e., the portion of the 100 target instances where the predictions of the source model were incorrect; denoted as *cor*) shows that the query strategy can select many instances that help the active learning model to adapt to the target domain.

The result tables of how the AL model evolves from start until the last target instance queried with regards to the average depth of trees, the confidence of predictions, and the importance of the attributes are listed in Figure B.18. Here, same as for the topic books, the average depth of trees is very high right after bootstrapping but drops significantly after 100 target instances are queried for all source-target combinations. The confidence in the AL model's predicted probabilities is already quite low lower when only learned on the source domain data. After expanding the forest with more trees that are only learned on the target instances, the predicted probabilities are even closer to the decision boundary. For the same source-target combination, differences in the ordered importance of the attributes after 100 target instances are queried can be found between applying a DA technique and not.

**Topic: Authors**

In Figure 7.21, the performance indicators of the ATLX method applied to the data from the topic authors, with a bootstrap sample size of one positive and one negative instance without and with domain adaptation technique applied, are listed. Compared to Baseline 1 (i.e., random sampling), ATLX without or with DA achieves significantly better results for all combinations. The profiling of the topic author's candidate sets has shown that the estimated domain relatedness is rather low among all of them, with dbp_dnb and dbp_wiki being the less related to each other and dbp_wiki and dbp_viaf being most related to each other (see Table 6.7). However, the conditional distributions are very similar among all three candidate sets, or at least there is no such a big difference to see in Figure 6.11 as it is the case for katom_rewo in the topic kitchen. In Figure 7.21 in column *f1_out_0* one can see the f1 score of the active learning model right after bootstrapping (i.e., iteration zero). For all combinations, no matter how high the estimated domain relatedness between source and target is, the naive transfer works very well. That is also reflected in the fact that all bootstrapping samples do not contain any *false positive* or *false negative* and the share of corrected labels (i.e., column *cor*) is rather low (if compared to the results of the other two semantic topics). However, one can also see that for the less related candidate sets, dbp_dnb and dbp_wiki, DA (especially LP) improves the f1 score right after bootstrapping. Nevertheless, the performance with DA or without in the last iteration is comparable, only for dbp_wiki as the source domain and dbp_dnb as the target domain, the standard deviation of the f1

score of the final iteration is lower if DA is applied.

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | | | | | | | | |
| **dbp_dnb** | **dbp_wiki** | **-** | 2 | 0 | 0 | 0.320 | 0.942 | 0.990 | 0.048 | 0.000 | 0.043 | -0.004 | 0.001 | 0.002 | 0.041 |
| | | **nn** | 2 | 0 | 0 | 0.260 | 0.955 | 0.990 | 0.035 | 0.000 | 0.026 | -0.004 | 0.001 | 0.002 | 0.024 |
| | | **lp** | 2 | 0 | 0 | 0.290 | 0.962 | 0.991 | 0.029 | 0.001 | 0.023 | -0.003 | 0.001 | 0.002 | 0.011 |
| **dbp_wiki** | **dbp_dnb** | **-** | 2 | 0 | 0 | 0.340 | 0.855 | 0.964 | 0.109 | 0.016 | 0.035 | -0.012 | 0.011 | 0.013 | 0.013 |
| | | **nn** | 2 | 0 | 0 | 0.350 | 0.863 | 0.967 | 0.104 | 0.019 | 0.038 | -0.009 | 0.004 | 0.013 | 0.011 |
| | | **lp** | 2 | 0 | 0 | 0.330 | 0.880 | 0.967 | 0.087 | 0.019 | 0.052 | -0.009 | 0.002 | 0.013 | 0.016 |
| **dbp_dnb** | **dbp_viaf** | **-** | 2 | 0 | 0 | 0.170 | 0.984 | 0.985 | 0.001 | 0.006 | 0.052 | -0.006 | 0.001 | 0.002 | 0.040 |
| | | **nn** | 2 | 0 | 0 | 0.200 | 0.986 | 0.985 | -0.001 | 0.006 | 0.033 | -0.006 | 0.002 | 0.002 | 0.008 |
| | | **lp** | 2 | 0 | 0 | 0.170 | 0.983 | 0.982 | -0.001 | 0.003 | 0.022 | -0.009 | 0.003 | 0.002 | 0.004 |
| **dbp_viaf** | **dbp_dnb** | **-** | 2 | 0 | 0 | 0.180 | 0.941 | 0.971 | 0.030 | 0.023 | 0.048 | -0.005 | 0.003 | 0.013 | 0.023 |
| | | **nn** | 2 | 0 | 0 | 0.140 | 0.926 | 0.969 | 0.043 | 0.021 | 0.030 | -0.007 | 0.002 | 0.013 | 0.011 |
| | | **lp** | 2 | 0 | 0 | 0.230 | 0.952 | 0.969 | 0.017 | 0.021 | 0.069 | -0.007 | 0.002 | 0.013 | 0.032 |
| **dbp_wiki** | **dbp_viaf** | **-** | 2 | 0 | 0 | 0.090 | 0.984 | 0.984 | 0.000 | 0.005 | 0.031 | -0.007 | 0.003 | 0.002 | 0.011 |
| | | **nn** | 2 | 0 | 0 | 0.130 | 0.984 | 0.985 | 0.001 | 0.006 | 0.029 | -0.006 | 0.002 | 0.002 | 0.006 |
| | | **lp** | 2 | 0 | 0 | 0.140 | 0.986 | 0.982 | -0.004 | 0.003 | 0.029 | -0.009 | 0.001 | 0.002 | 0.010 |
| **dbp_viaf** | **dbp_wiki** | **-** | 2 | 0 | 0 | 0.080 | 0.992 | 0.991 | -0.001 | 0.001 | 0.011 | -0.003 | 0.001 | 0.002 | 0.002 |
| | | **nn** | 2 | 0 | 0 | 0.120 | 0.991 | 0.991 | 0.000 | 0.001 | 0.026 | -0.003 | 0.001 | 0.002 | 0.016 |
| | | **lp** | 2 | 0 | 0 | 0.110 | 0.992 | 0.992 | 0.000 | 0.002 | 0.019 | -0.002 | 0.001 | 0.002 | 0.008 |

Figure 7.21: Topic Authors - ATLX results with and without DA

The result tables of how the AL model evolves from start until the last target instance queried with regards to the average depth of trees, the confidence of predictions, and the importance of the attributes are listed in Figure B.19. Here, similar to the remaining topics, the average depth of trees is very high right after bootstrapping but drops significantly after 100 target instances are queried for all source-target combinations. The confidence in the AL model's predicted probabilities is very high when only learned on the source domain data and only drops slightly after 100 target instances are queried. That was expected as all candidate sets of the topic authors are rather straightforward. For the same source-target combination, no differences in the ordered importance of the attributes after 100 target instances are queried can be found between applying a DA technique and not. For some combinations, the attribute importance does not change between after bootstrapping and final iteration.

With regards to Baseline 2 (i.e., AL with Unsupervised Bootstrapping), ATLX with or without DA achieves slightly better or similar results in the last iteration for four out of six combinations. For the remaining two, the results are significantly better than Baseline 2. The superior performance of ATLX compared to its baselines can be seen in Figure 7.22, where the f1 score per iteration and the standard

deviation is plotted. Here, one can see that ATLX performs significantly better than Baseline 2 right from the beginning and that the query strategy selects valuable instances from the pool of unlabeled target instances as the performance from ATLX with random sampling (i.e., "ATLX: no_weighting: random") drops significantly for all combinations after a few target instances are queried. On the other hand, ATLX without DA can increase performance or at least keep the performance after bootstrapping (which is already close to the passive learning benchmark) for all combinations.

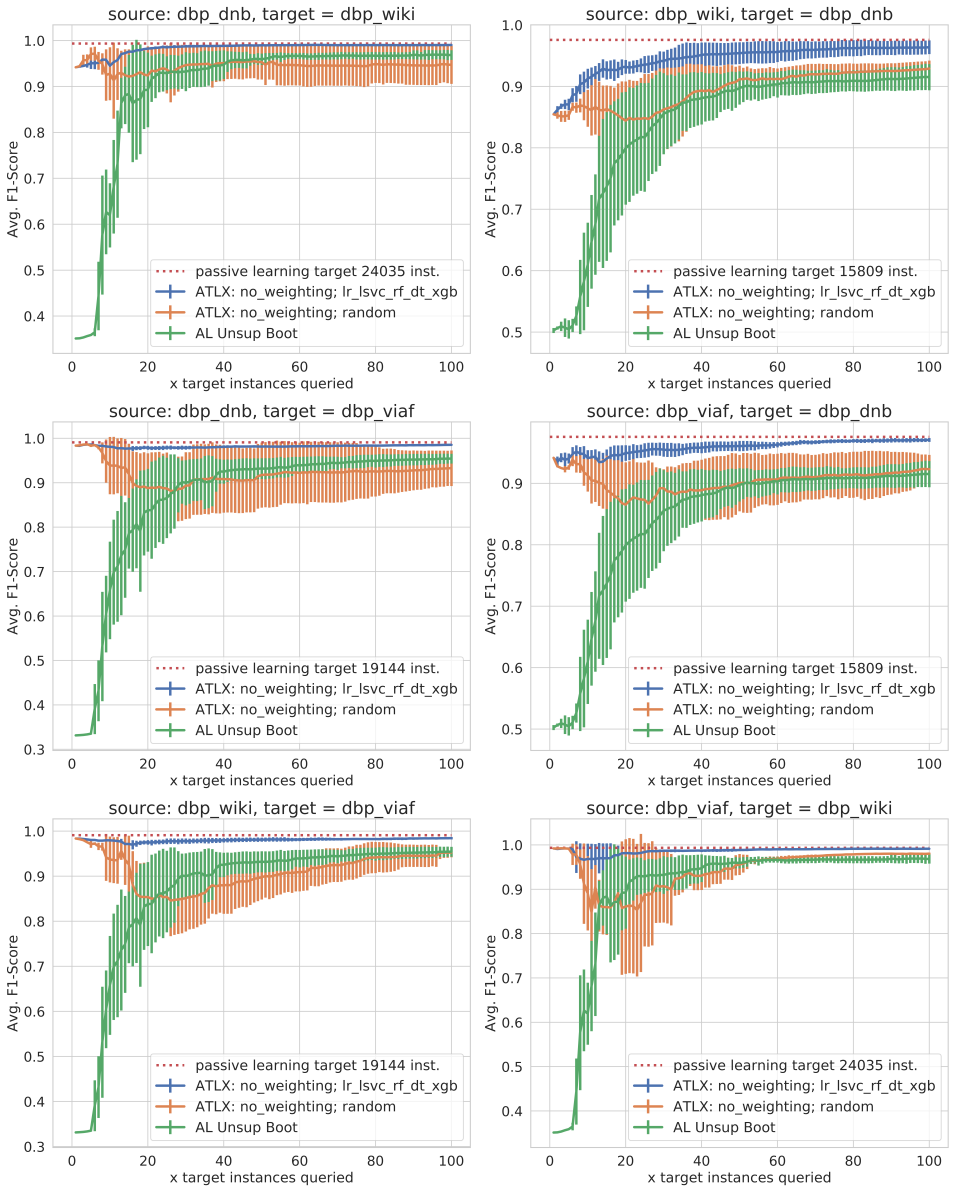


Figure 7.22: Topic Authors - ATLX without DA and Baselines (errorbars)

# Chapter 8

# Discussion and Future Work

The proposed method for ER, ATLX, which combines TL and AL, has shown to successfully mitigate the risks identified in the naive transfer experiments and the initial ATL experiments where simply all labeled source domain data gets included in the labeled set. Besides that, ATLX without DA manages to exceed its baseline for all combinations of the semantic topic authors. It also leads to much better performance with regards to quality and stability right from the beginning of the AL process. In the best case, the unsupervised bootstrapped AL baseline's f1 score is outperformed by $0.023$, and random sampling's f1 score is outperformed by $0.052$.

Furthermore, the f1 score of all combinations in the last iteration comes very close to the passive learning benchmark, with the best case being only $0.003$ below passive learning's f1 score. Applying DA does not lead to different results than ATLX without DA in the last iteration. Only for the combination dbp_dnb and dbp_wiki, where the estimated domain relatedness indicated a lower relatedness, DA leads to better results right after bootstrapping. Here LP leads to a better f1 score than NN. Due to the lower portion of corner cases in the candidate sets of the topic authors (maximum has dbp_dnb with $0.34$) and the same relevant attributes among the candidate sets, this topic can be considered the less difficult one among the three semantic topics. Dbp_dnb and dbp_viaf from the topic authors have been used in another work [62].

For the remaining two semantic topics, which contain candidate sets with a very high amount of corner cases (topic kitchen: katom_rewo with $0.93$, topic books: wor_half with $0.63$) ATLX without DA still achieves better results than its baselines for most of the combinations of both topics and for the topic books it even exceeds the passive learning benchmark's f1 score in five out of 24 combinations with only 100 target instances queried. In the best case, ATLX without DA

outperforms the unsupervised bootstrapped AL baseline's f1 score by $0.048$ and the random sampling's f1 score by $0.038$ for the topic books. For the topic kitchen, the unsupervised bootstrapped AL baseline's f1 score is outperformed by ATLX without DA by $0.084$, in the best case, and random sampling's f1 score by $0.061$

For the few combinations (i.e., topic books: four and topic kitchen: one) where the unsupervised bootstrapped AL baseline's f1 score is slightly above the f1 score of ATLX without DA in the last iteration, applying one of the DA techniques leads to better results than the baseline. However, it is impossible to determine upfront which of the DA technique leads to better results as not always both increase the performance. The only observation is that there are instances in the source domain that help the classification model in the target domain more and should be assigned a higher weight.

Further research needs to be conducted to find more sophisticated ways to identify these instances as the unsupervised DA techniques do not consider the conditional distributions in both domains. However, a potential cause of worse performance when applying unsupervised DA techniques like LP and NN could be that more weight is assigned on instances that are not representative of the conditional distributions in the target domain. Nevertheless, as in the AL process labels are acquired from the target domain, this knowledge could be used to identify source domain instances that are also representative of the conditional distributions. Thus, a different setting where labeled source domain data is not only incorporated before the AL process but also during (i.e., after some target instances are queried), for instance, similar to the work of Persello and Bruzzone [58], needs to be researched in future work.

Because nothing is known about the conditional distributions, the estimated domain relatedness that only looks at the marginal distributions between source and target domains has shown not to be a perfect indicator of whether a transfer of knowledge between both domains will work well. Besides that, the estimated domain relatedness implications could get unreliable if too many features are considered as it gets easier for the logistic regression model to find patterns that help discriminate between both domains. However, these differences in the distributions may not play a role in the naive transfer as these features are not relevant for the classification model. Therefore, good transfer results are still possible.

One possibility that needs to be analyzed in future research would be to estimate the domain relatedness only for the top relevant attributes from the source domain. In the scenario considered in this thesis, labeled data is only available for the source domain (before AL starts). For this reason, the top relevant attributes from the target domain cannot be known. However, if substantial differences in the marginal distributions of the top relevant attributes from the source can be found in both domains, it should indicate a higher chance for negative transfer. This aspect

and the fact that a naive transfer sometimes works in one direction quite well but not the other way around needs to be analyzed in future work.

Another possibility for future work is to analyze if the bootstrapping sample size in ATLX of two record pairs can be increased. Here initial tests with a sample size of five positive and five negative record pairs have shown that it can lead to better performance with fewer target instances queried. However, the risk of adding noise to the label set increases. Additionally, finding metrics other than the ones presented in this work to measure how ATLX evolves from right after bootstrapping until the convergence point could help to analyze ATLX's performance better. The metrics, average depth of trees, the confidence of predictions, and order importance of attributes did not reveal significant patterns as to why ATLX works very well and why it does not.

Furthermore, measuring the disagreement among the committee members in ATLX using another disagreement measure than vote entropy is an alternative that could be analyzed in future work. However, initial experiments with measuring the disagreement using KL divergence on the predicted probabilities instead of the predicted labels have shown to not lead to better results. Still, this needs to be researched more precisely.

Additionally, ATLX should be validated on more semantic topics and compared to other state-of-the-art ER methods. It would be especially interesting to see how ATLX performs compared to the method proposed by Kasai et al. [32], which is the only method that combines AL and TL for ER, besides ATLX, but with the means of deep learning.

# Chapter 9

# Conclusion

In this thesis, the problem of transferring knowledge from a source domain to resolve entities in a target domain within the same semantic topic was researched. More precisely, it was examined whether the transferred knowledge can help AL in the target domain. For that, two initial experiments concerning the naive transfer (i.e., learning on the labeled source domain data and apply on the target domain) and incorporating labeled source domain data in the labeled set of an AL setting have been conducted. The experiments aimed to gain insight into what pitfalls a final method combining TL and AL needs to tackle before the final method was conceptualized.

The NTL experiments have shown that a naive transfer within a semantic topic can work quite well compared to unsupervised matching. Here ensemble methods have shown to achieve better results. However, it is not guaranteed that naive transfer does not lead to worse results than unsupervised matching. The profiling dimensions have shown that differences in the top related attributes between the source and target domains and a high number of corner cases can be a potential cause of negative transfer.

Still, when considering the problem setting of this thesis where the labels are only known in the source domain, these profiling dimensions cannot be used to determine upfront if a transfer will work well. Nevertheless, the estimated domain relatedness, based on differences in the feature space of the two domains, suggests that the more related the two domains are, the higher the chance that the naive transfer will work quite well.

Combining AL and TL (ATL) by merely incorporating all labeled source domain instances in the labeled set of AL has shown to not only hinder the active learning model from adapting to the target domain but also to mislead the query strategy to select valuable instances from the pool of unlabeled target instances.

Here the DA techniques in the form of a stronger weighting of source instances that are more similar to the target instances have proven unable to reduce these risks on their own.

The final method proposed in this thesis, which combines AL and TL and is called ATLX, does consider the risks identified from the NTL and ATL experiments. First of all, in ATLX, the influence of the source domain data on the active learning model is lowered so that a potential negative transfer has less influence on the performance. Therefore, it can adapt to the target domain, with only 100 target instances queried. That is achieved - following a similar approach as [62] - by training an initial random forest with ten trees only on the labeled data of the source domain and extending the random forest with each iteration by two additional trees, which are trained only on the target instances queried so far.

Second, the committee members are not trained on any source domain data but on the labeled set, which gets bootstrapped with one positive and one negative record pair from the target domain (i.e., the bootstrapping sample). Before the start of AL, a random forest with 100 estimators is trained only on the source domain data to make predictions on the unlabeled pool of target instances. Here DA techniques can be used if the two domains are less related to each other. The two record pairs with the highest predicted probabilities are added to the labeled set as the bootstrapping sample. Additionally, each prediction counts for one additional vote besides the votes of the committee members.

ATLX has shown that it can successfully solve the cold start problem, a common problem with AL. It has also been shown that when labeled data is available from another domain of the same semantic topic, it is preferable to bootstrapping AL with unsupervised matching. The evaluation of ATLX has shown that it outperforms its baselines in most source-target combinations for the two difficult semantic topics books and kitchen product offers. For the topic authors, ATLX has shown to outperform its baseline in terms of the quality and robustness of the model learned for all source-target combinations at any time.

# Bibliography


[1] 8.1. datetime — Basic date and time types — Python 3.3.7 documentation.

[2] Die Deutsche Nationalbibliothek im Porträt. Library Catalog: www.dnb.de.

[3] Arvind Arasu, Michaela Götz, and Raghav Kaushik. On active learning of record matching packages. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, SIGMOD '10, pages 783–794, Indianapolis, Indiana, USA, June 2010. Association for Computing Machinery.

[4] Les E. Atlas, David A. Cohn, and Richard E. Ladner. Training Connectionist Networks with Queries and Selective Sampling. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 566–573. Morgan-Kaufmann, 1990.

[5] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for a Web of Open Data. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, volume 4825, pages 722–735. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. Series Title: Lecture Notes in Computer Science.

[6] B Settles. Active learning. *Synthesis lectures on artificial intelligence and machine learning*, 6(1):1, 2012.

[7] Pierre Baldi, Søren Brunak, Yves Chauvin, Claus A. F. Andersen, and Henrik Nielsen. Assessing the accuracy of prediction algorithms for classification:

an overview. *Bioinformatics*, 16(5):412–424, May 2000. Publisher: Oxford Academic.

[8] Kedar Bellare, Suresh Iyengar, Aditya G. Parameswaran, and Vibhor Rastogi. Active Sampling for Entity Matching. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1131–1139, New York, NY, USA, 2012. ACM. event-place: Beijing, China.

[9] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning - ICML '07*, pages 81–88, Corvalis, Oregon, 2007. ACM Press.

[10] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996.

[11] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001.

[12] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. event-place: San Francisco, California, USA.

[13] Xiao Chen, Yinlong Xu, David Broneske, Gabriel Campero Durand, Roman Zoun, and Gunter Saake. Heterogeneous Committee-Based Active Learning for Entity Resolution (HeALER). In Tatjana Welzer, Johann Eder, Vili Podgorelec, and Aida Kamišalić Latifić, editors, *Advances in Databases and Information Systems*, volume 11695, pages 69–85. Springer International Publishing, Cham, 2019. Series Title: Lecture Notes in Computer Science.

[14] Peter Christen. A Comparison of Personal Name Matching: Techniques and Practical Issues. In *Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06)*, pages 290–294, Hong Kong, China, 2006. IEEE.

[15] Peter Christen. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Data-Centric Systems and Applications. Springer-Verlag, Berlin Heidelberg, 2012.

[16] Peter Christen. A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. *IEEE Transactions on Knowledge and Data Engineering*, 24(9):1537–1555, September 2012.

[17] Victor Christen, Peter Christen, and Erhard Rahm. Informativeness-Based Active Learning for Entity Resolution. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 125–141. Springer, 2019.

[18] William W. Cohen and Jacob Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 475–480, Edmonton, Alberta, Canada, July 2002. Association for Computing Machinery.

[19] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, May 1994.

[20] Sanjib Das, AnHai Doan, Paul Suganthan G. C., Chaitanya Gokhale, Pradap Konda, Yash Govind, and Derek Paulsen. *The Magellan Data Repository*. University of Wisconsin-Madison.

[21] H. Daume III and D. Marcu. Domain Adaptation for Statistical Classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, June 2006. arXiv: 1109.6341.

[22] Oscar Day and Taghi M. Khoshgoftaar. A survey on heterogeneous transfer learning. *Journal of Big Data*, 4(1):29, September 2017.

[23] Junio de Freitas, Gisele L. Pappa, Altigran S. da Silva, Marcos A. Gonc¸alves, Edleno Moura, Adriano Veloso, Alberto H. F. Laender, and Moisés G. de Carvalho. Active Learning Genetic programming for record deduplication. In *IEEE Congress on Evolutionary Computation*, pages 1–8, July 2010. ISSN: 1089-778X, 1941-0026.

[24] AnHai Doan. *Principles of data integration*. Amsterdam Heidelberg ua: Elsevier, Morgan Kaufmann, Amsterdam Heidelberg [u.a., 2012.

[25] Xin Luna Dong and Divesh Srivastava. Big Data Integration. *2015*, page 200.

[26] Ivan P. Fellegi and Alan B. Sunter. A Theory for Record Linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, December 1969. Publisher: Taylor & Francis.

[27] Jeffrey Fisher, Peter Christen, and Qing Wang. Active Learning Based Entity Resolution Using Markov Logic. In James Bailey, Latifur Khan, Takashi

Washio, Gill Dobbie, Joshua Zhexue Huang, and Ruili Wang, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 338–349, Cham, 2016. Springer International Publishing.

[28] franciscojmartin. A Simple Machine Learning Method to Detect Covariate Shift, January 2014. Library Catalog: blog.bigml.com.

[29] Jerome H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. Publisher: Institute of Mathematical Statistics.

[30] Robert Isele and Christian Bizer. Active learning of expressive linkage rules using genetic programming. *Journal of Web Semantics*, 23:2–15, December 2013.

[31] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, editors. *An introduction to statistical learning: with applications in R*. Number 103 in Springer texts in statistics. Springer, New York, 2013. OCLC: ocn828488009.

[32] Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource Deep Entity Resolution with Transfer and Active Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5851–5861, Florence, Italy, 2019. Association for Computational Linguistics.

[33] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[34] Pradap Konda, Jeff Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, Vijay Raghavendra, Sanjib Das, Paul Suganthan G. C., AnHai Doan, Adel Ardalan, Jeffrey R. Ballard, Han Li, Fatemah Panahi, and Haojun Zhang. Magellan: toward building entity matching management systems. *Proceedings of the VLDB Endowment*, 9(12):1197–1208, August 2016.

[35] Wouter M. Kouw and Marco Loog. A review of domain adaptation without target labels. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[36] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. Publisher: Institute of Mathematical Statistics.

[37] Hanna Köpcke and Erhard Rahm. Training Selection for Tuning Entity Matching. page 11, 2008.

[38] Hanna Köpcke, Andreas Thor, and Erhard Rahm. Evaluation of entity resolution approaches on real-world match problems. *Proceedings of the VLDB Endowment*, 3(1-2):484–493, September 2010.

[39] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. Publisher: Nature Publishing Group.

[40] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, pages 3–12, Dublin, Ireland, August 1994. Springer-Verlag.

[41] Xin Li, Paul Morie, and Dan Roth. Robust Reading: Identification and Tracing of Ambiguous Names. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 17–24, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics.

[42] Xin Li, Paul Morie, and Dan Roth. Semantic Integration in Text: From Ambiguous Names to Identifiable Entities. *AI Magazine*, 26(1):45–45, March 2005. Number: 1.

[43] Martha Fallahay Loesch. VIAF (The Virtual International Authority File) – http://viaf.org. *Technical Services Quarterly*, 28(2):255–256, February 2011. Publisher: Routledge _eprint: https://doi.org/10.1080/07317131.2011.546304.

[44] Marco Loog. Nearest neighbor-based importance weighting. In *2012 IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6, September 2012. ISSN: 2378-928X.

[45] Andrew McCallum, Kamal Nigam, and Lyle H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, pages 169–178, Boston, Massachusetts, United States, 2000. ACM Press.

[46] David Menestrina, Steven Euijong Whang, and Hector Garcia-Molina. Evaluating entity resolution results. *Proceedings of the VLDB Endowment*, 3(1-2):208–219, September 2010.

[47] Kevin P Murphy. *Machine learning : a probabilistic perspective*. ProQuest Ebook Central. Cambridge, Massachusetts London, England, 2012.

[48] Felix Naumann and Melanie Herschel. An Introduction to Duplicate Detection. *Synthesis Lectures on Data Management*, 2(1):1–87, January 2010.

[49] Sahand Negahban, Benjamin I. P. Rubinstein, and Jim Gemmell. Scaling Multiple-Source Entity Resolution using Statistically Efficient Transfer Learning. *arXiv:1208.1860 [cs]*, August 2012. arXiv: 1208.1860.

[50] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic Linkage of Vital Records: Computers can be used to extract "follow-up" statistics of families from files of routine records. *Science*, 130(3381):954–959, October 1959. Publisher: American Association for the Advancement of Science Section: Articles.

[51] Axel-Cyrille Ngonga Ngomo, Jens Lehmann, Soren Auer, and Konrad Hoffner. RAVEN – Active Learning of Link Specifications. In *Proceedings of the 6th International Workshop on Ontology Matching (OM-2011)*, page 12, 2011.

[52] Axel-Cyrille Ngonga Ngomo, Jens Lehmann, and Mofeed Hassan. Towards Transfer Learning of Link Specifications. In *2013 IEEE Seventh International Conference on Semantic Computing*, pages 202–205, September 2013.

[53] Axel-Cyrille Ngonga Ngomo, Klaus Lyko, and Victor Christen. COALA – Correlation-Aware Active Learning of Link Specifications. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Philipp Cimiano, Oscar Corcho, Valentina Presutti, Laura Hollink, and Sebastian Rudolph, editors, *The Semantic Web: Semantics and Big Data*, volume 7882, pages 442–456. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. Series Title: Lecture Notes in Computer Science.

[54] Axel-Cyrille Ngonga Ngomo and Klaus Lyko. EAGLE: Efficient Active Learning of Link Specifications Using Genetic Programming. In Elena Simperl, Philipp Cimiano, Axel Polleres, Oscar Corcho, and Valentina Presutti, editors, *The Semantic Web: Research and Applications*, Lecture Notes in Computer Science, pages 149–163, Berlin, Heidelberg, 2012. Springer.

[55] Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

[56] George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. A Survey of Blocking and Filtering Techniques for Entity Resolution. 1(1):36, May 2019.

[57] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[58] Claudio Persello and Lorenzo Bruzzone. Active Learning for Domain Adaptation in the Supervised Classification of Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, 50(11):4468–4483, November 2012.

[59] Martin F Porter. *An algorithm for suffix stripping*. 1980.

[60] Anna Primpeli and Christian Bizer. Robust Active Learning of Expressive Linkage Rules. In *Proceedings of the 9th International Conference on Web Intelligence, Mining and Semantics - WIMS2019*, pages 1–7, Seoul, Republic of Korea, 2019. ACM Press.

[61] Anna Primpeli and Christian Bizer. Profiling Entity Matching Benchmark Tasks. In *[under review]*, page 9, 2020.

[62] Anna Primpeli, Christian Bizer, and Margret Keuper. Unsupervised Bootstrapping of Active Learning for Entity Resolution. page 16, 2020.

[63] Kun Qian, Lucian Popa, and Prithviraj Sen. Active Learning for Large-Scale Entity Resolution. page 11, November 2017.

[64] Piyush Rai, Avishek Saha, Hal Daumé, and Suresh Venkatasubramanian. Domain Adaptation meets Active Learning. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 27–32, Los Angeles, California, June 2010. Association for Computational Linguistics.

[65] Pradeep Ravikumar and William Cohen. A Hierarchical Graphical Model for Record Linkage. *arXiv:1207.4180 [cs, stat]*, July 2012. arXiv: 1207.4180.

[66] Michael T. Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G. Dietterich. To transfer or not to transfer. In *In NIPS'05 Workshop, Inductive Transfer: 10 Years Later*, 2005.

[67] G. van Rossum. Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, May 1995.

[68] Sunita Sarawagi and Anuradha Bhamidipaty. Interactive deduplication using active learning. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 269–278, Edmonton, Alberta, Canada, July 2002. Association for Computing Machinery.

[69] Giovanni Seni and John F. Elder. Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 2(1):1–126, January 2010.

[70] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, October 2000.

[71] Sheila Tejada, Craig A Knoblock, and Steven Minton. Learning object identification rules for information integration. *Information Systems*, 26(8):607–633, December 2001.

[72] Saravanan Thirumuruganathan, Shameem A. Puthiya Parambath, Mourad Ouzzani, Nan Tang, and Shafiq Joty. Reuse and Adaptation for Entity Resolution through Transfer Learning. *arXiv:1809.11084 [cs, stat]*, September 2018. arXiv: 1809.11084.

[73] C. J. van Rijsbergen. Information Retrieval, 1979.

[74] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, September 2014.

[75] Qing Wang, Dinusha Vatsalan, and Peter Christen. Efficient Interactive Training Selection for Large-scale Entity Resolution. page 13, 2015.

[76] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 3(1):9, May 2016.

[77] Yao-Yuan Yang, Shao-Chuan Lee, Yu-An Chung, Tung-En Wu, Si-An Chen, and Hsuan-Tien Lin. libact: Pool-based Active Learning in Python. Technical report, National Taiwan University, October 2017.

[78] Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *In International Conference on Machine Learning ICML'04*, pages 903–910, 2004.

[79] Chen Zhao and Yeye He. Auto-EM: End-to-end Fuzzy Entity-Matching using Pre-trained Deep Models and Transfer Learning. In *The World Wide Web Conference*, WWW '19, pages 2413–2424, San Francisco, CA, USA, May 2019. Association for Computing Machinery.

[80] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A Comprehensive Survey on Transfer Learning. *arXiv:1911.02685 [cs, stat]*, November 2019. arXiv: 1911.02685.

[81] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web - WWW '05*, page 22, Chiba, Japan, 2005. ACM Press.

# Appendix A

# Parameter Setting

Everything is also available at GitHub[1] and on the memory stick attached to this thesis.

## A.1 Filtering

| Combo | Filtering Attribute | Similarity Measure | Threshold | PC (%) | RR (%) | #Corr |
|---|---|---|---|---|---|---|
| Ban_bx | Title | Jaccard (on 3-grams) | 0.3 | 91.85 | 99.99 | 204,456 |
| Ban_half | Author | Jaccard (on 3-grams) | 0.3 | 99.21 | 99.82 | 49,764 |
| Ban_wor | First two authors | Jaccard (on 3-grams) | 0.3 | 98.45 | 99.8 | 843,590 |
| Bx_half | Author | Jaccard (on 3-grams) | 0.3 | 99.27 | 99.83 | 700,881 |
| Bx_wor | Title NP NSW | Jaccard (on 3-grams) | 0.5 | 91.65 | 99.98 | 1,334,010 |
| Wor_half | Author | Jaccard (on 3-grams) | 0.3 | 97.3 | 99.81 | 146,851 |

Figure A.1: Topic Books - Filtering Setting. NP = no parentheses, NSW = English stop words removed

| Combo | Filtering Attribute | Similarity Measure | Threshold | #Corr |
|---|---|---|---|---|
| Rewo_cdi | All Attributes concatenated | Jaccard (on 3-grams) | 0.35 | 833,042 |
| Katom_cdi | All Attributes concatenated | Jaccard (on 3-grams) | 0.3 | 563,710 |
| Katom_rewo | All Attributes concatenated | Jaccard (on 3-grams) | 0.5 | 607,956 |

Figure A.2: Topic Kitchen - Filtering Setting

[1] https://github.com/JayKay0104/ma-atl-for-er

## A.2 Parameter Setting of Learning Algorithms

| | Parameter Setting |
|---|---|
| **logreg** | {'C': 1.0, 'class_weight': None, 'dual': False, 'fit_intercept': True, 'intercept_scaling': 1, 'l1_ratio': None, 'max_iter': 1000, 'multi_class': 'auto', 'n_jobs': None, 'penalty': 'l2', 'random_state': 42, 'solver': 'liblinear', 'tol': 0.0001, 'verbose': 0, 'warm_start': False} |
| **dectree** | {'ccp_alpha': 0.0, 'class_weight': None, 'criterion': 'gini', 'max_depth': None, 'max_features': None, 'max_leaf_nodes': None, 'min_impurity_decrease': 0.0, 'min_impurity_split': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'presort': 'deprecated', 'random_state': 42, 'splitter': 'best'} |
| **randforest** | {'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': None, 'criterion': 'gini', 'max_depth': None, 'max_features': 'auto', 'max_leaf_nodes': None, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_impurity_split': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 100, 'n_jobs': None, 'oob_score': False, 'random_state': 42, 'verbose': 0, 'warm_start': False} |
| **xgb** | {'objective': 'binary:logistic', 'base_score': None, 'booster': None, 'colsample_bylevel': None, 'colsample_bynode': None, 'colsample_bytree': None, 'gamma': None, 'gpu_id': None, 'importance_type': 'gain', 'interaction_constraints': None, 'learning_rate': None, 'max_delta_step': None, 'max_depth': None, 'min_child_weight': None, 'missing': nan, 'monotone_constraints': None, 'n_estimators': 100, 'n_jobs': None, 'num_parallel_tree': None, 'random_state': 42, 'reg_alpha': None, 'reg_lambda': None, 'scale_pos_weight': None, 'subsample': None, 'tree_method': None, 'validate_parameters': False, 'verbosity': None} |
| **svm** | {'C': 1.0, 'class_weight': None, 'dual': False, 'fit_intercept': True, 'intercept_scaling': 1, 'loss': 'squared_hinge', 'max_iter': 1000, 'multi_class': 'ovr', 'penalty': 'l2', 'random_state': 42, 'tol': 0.0001, 'verbose': 0} |
| **logregcv** | {'Cs': 10, 'class_weight': None, 'cv': 5, 'dual': False, 'fit_intercept': True, 'intercept_scaling': 1.0, 'l1_ratios': None, 'max_iter': 1000, 'multi_class': 'auto', 'n_jobs': None, 'penalty': 'l2', 'random_state': 42, 'refit': True, 'scoring': None, 'solver': 'liblinear', 'tol': 0.0001, 'verbose': 0} |

Figure A.3: Parameter Setting of Learning Algorithms used

# Appendix B

# Further Profiling and Experimental Results

## B.1 Further Profiling Information

### B.1.1 Topic: Books

| | Combination: ban_bx Test Set |
|---|---|
| **matching_relevant_features** | [pubdate_months_diff_sim, pubdate_days_diff_sim, pubdate_years_diff_sim, publisher_lev_sim, title_containment_sim, publisher_jac_q3_sim, title_jac_q3_sim, author_lev_sim, title_jac_an_sim, title_rel_jac_an_sim, title_lev_sim, author_jac_q3_sim, publisher_rel_jac_an_sim, publisher_jac_an_sim, author_rel_jac_an_sim, publisher_containment_sim, author_jac_an_sim, author_exact_sim] |
| **matching_relevant_attributes_datatypes** | {str, date} |
| **matching_relevant_attributes** | {author, title, pubdate, publisher} |
| **matching_relevant_attributes_count** | 4 |
| **matching_relevant_attributes_density** | 0.98 |
| **top_matching_relevant_features_count** | 4 |
| **F1_xval_max** | 0.8 |
| **F1_xval_top_matching_relevant_features** | 0.79 |
| **top_matching_relevant_features** | [pubdate_months_diff_sim, pubdate_days_diff_sim, pubdate_years_diff_sim, publisher_lev_sim] |
| **top_relevant_attributes** | {pubdate, publisher} |
| **top_relevant_attributes_datatypes** | {str, date} |
| **top_relevant_attributes_count** | 2 |
| **top_relevant_attributes_density** | 0.9 |
| **avg_length_tokens_top_relevant_attributes** | 3 |
| **avg_length_words_top_relevant_attributes** | 1 |
| **corner_cases_top_matching_relevant_features** | 0.64 |
| **avg_uniqueness_top_matching_relevant_features** | 0.85 |

Figure B.1: Profiling Results for the test set of ban_bx

## B.2 NTL Results

| | Features | all | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | logreg | | | | dectree | | | | randforest | | | | xgb | | | | svm | | | | logregcv | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.632 | 0.709 | 24 | 0.686 | 0.636 | 0.744 | - | 0.794 | 0.674 | 0.787 | 10 | 0.818 | 0.683 | 0.788 | - | 0.856 | 0.621 | 0.716 | 20 | 0.678 | 0.611 | 0.721 | 14 | 0.688 | 0.703 |
| bx_wor | ban_bx | 0.533 | 0.705 | 10 | 0.655 | 0.668 | 0.798 | 10 | 0.794 | 0.755 | 0.845 | 14 | 0.814 | 0.777 | 0.849 | 20 | 0.872 | 0.535 | 0.669 | 10 | 0.640 | 0.539 | 0.650 | 10 | 0.703 | 0.656 |
| ban_bx | ban_half | 0.822 | 0.875 | 38 | 0.889 | 0.898 | 0.942 | 10 | 0.945 | 0.939 | 0.962 | 38 | 0.969 | 0.924 | 0.954 | 20 | 0.960 | 0.830 | 0.899 | 28 | 0.897 | 0.834 | 0.929 | 28 | 0.914 | 0.761 |
| ban_half | ban_bx | 0.726 | 0.687 | - | 0.655 | 0.395 | 0.801 | - | 0.794 | 0.547 | 0.842 | - | 0.814 | 0.332 | 0.846 | - | 0.872 | 0.000 | 0.670 | - | 0.640 | 0.000 | 0.664 | - | 0.703 | 0.656 |
| ban_bx | bx_half | 0.585 | 0.731 | 10 | 0.731 | 0.873 | 0.865 | - | 0.866 | 0.894 | 0.896 | 300 | 0.895 | 0.860 | 0.890 | 14 | 0.894 | 0.555 | 0.751 | - | 0.756 | 0.601 | 0.750 | 10 | 0.766 | 0.688 |
| bx_half | ban_bx | 0.577 | 0.688 | 10 | 0.655 | 0.731 | 0.793 | 10 | 0.794 | 0.766 | 0.836 | 14 | 0.814 | 0.796 | 0.851 | 38 | 0.872 | 0.569 | 0.671 | 10 | 0.640 | 0.577 | 0.661 | 14 | 0.703 | 0.656 |
| ban_bx | ban_wor | 0.694 | 0.660 | - | 0.682 | 0.766 | 0.796 | 120 | 0.788 | 0.802 | 0.836 | 50 | 0.830 | 0.781 | 0.830 | 32 | 0.846 | 0.690 | 0.674 | - | 0.709 | 0.695 | 0.667 | - | 0.708 | 0.679 |
| ban_wor | ban_bx | 0.000 | 0.659 | - | 0.655 | 0.198 | 0.795 | - | 0.794 | 0.134 | 0.844 | - | 0.814 | 0.107 | 0.855 | - | 0.872 | 0.034 | 0.652 | - | 0.640 | 0.034 | 0.651 | - | 0.703 | 0.656 |
| bx_wor | wor_half | 0.684 | 0.754 | 60 | 0.797 | 0.706 | 0.809 | - | 0.841 | 0.701 | 0.860 | - | 0.864 | 0.698 | 0.864 | - | 0.896 | 0.689 | 0.769 | 44 | 0.801 | 0.685 | 0.780 | 28 | 0.802 | 0.699 |
| wor_half | bx_wor | 0.269 | 0.702 | - | 0.686 | 0.432 | 0.741 | - | 0.794 | 0.391 | 0.788 | - | 0.818 | 0.382 | 0.792 | - | 0.856 | 0.270 | 0.707 | - | 0.678 | 0.317 | 0.724 | - | 0.688 | 0.703 |
| bx_wor | bx_half | 0.746 | 0.731 | - | 0.731 | 0.767 | 0.867 | - | 0.866 | 0.795 | 0.892 | 10 | 0.895 | 0.795 | 0.888 | 10 | 0.894 | 0.757 | 0.753 | - | 0.756 | 0.752 | 0.756 | 300 | 0.766 | 0.688 |
| bx_half | bx_wor | 0.619 | 0.686 | 10 | 0.686 | 0.701 | 0.737 | 60 | 0.794 | 0.740 | 0.787 | 38 | 0.818 | 0.739 | 0.789 | 50 | 0.856 | 0.615 | 0.716 | 10 | 0.678 | 0.625 | 0.719 | 10 | 0.688 | 0.703 |
| bx_wor | ban_wor | 0.675 | 0.676 | 300 | 0.682 | 0.691 | 0.794 | 10 | 0.788 | 0.724 | 0.837 | 10 | 0.830 | 0.760 | 0.835 | 20 | 0.846 | 0.677 | 0.665 | - | 0.709 | 0.675 | 0.668 | - | 0.708 | 0.679 |
| ban_wor | bx_wor | 0.025 | 0.694 | - | 0.686 | 0.177 | 0.739 | - | 0.794 | 0.105 | 0.788 | - | 0.818 | 0.131 | 0.785 | - | 0.856 | 0.057 | 0.709 | - | 0.678 | 0.026 | 0.716 | - | 0.688 | 0.703 |
| ban_half | wor_half | 0.747 | 0.765 | 200 | 0.797 | 0.811 | 0.808 | - | 0.841 | 0.849 | 0.856 | 300 | 0.864 | 0.847 | 0.858 | 200 | 0.896 | 0.742 | 0.776 | 80 | 0.801 | 0.760 | 0.775 | 80 | 0.802 | 0.699 |
| wor_half | ban_half | 0.896 | 0.871 | - | 0.889 | 0.796 | 0.948 | - | 0.945 | 0.906 | 0.961 | 14 | 0.969 | 0.901 | 0.953 | 14 | 0.960 | 0.912 | 0.902 | - | 0.897 | 0.918 | 0.932 | 120 | 0.914 | 0.761 |
| ban_half | bx_half | 0.780 | 0.726 | - | 0.731 | 0.212 | 0.871 | - | 0.866 | 0.590 | 0.896 | - | 0.895 | 0.286 | 0.888 | - | 0.894 | 0.032 | 0.750 | - | 0.756 | 0.032 | 0.750 | - | 0.766 | 0.688 |
| bx_half | ban_half | 0.756 | 0.874 | - | 0.889 | 0.544 | 0.944 | - | 0.945 | 0.715 | 0.960 | - | 0.969 | 0.870 | 0.953 | 10 | 0.960 | 0.759 | 0.905 | 10 | 0.897 | 0.783 | 0.921 | 20 | 0.914 | 0.761 |
| ban_half | ban_wor | 0.684 | 0.660 | - | 0.682 | 0.740 | 0.790 | 38 | 0.788 | 0.799 | 0.842 | 44 | 0.830 | 0.782 | 0.830 | 38 | 0.846 | 0.694 | 0.673 | - | 0.709 | 0.712 | 0.672 | - | 0.708 | 0.679 |
| ban_wor | ban_half | 0.653 | 0.874 | - | 0.889 | 0.823 | 0.947 | - | 0.945 | 0.910 | 0.959 | 10 | 0.969 | 0.904 | 0.951 | 14 | 0.960 | 0.646 | 0.900 | - | 0.897 | 0.664 | 0.915 | - | 0.914 | 0.761 |
| wor_half | bx_half | 0.313 | 0.728 | - | 0.731 | 0.523 | 0.870 | - | 0.866 | 0.391 | 0.891 | - | 0.895 | 0.410 | 0.888 | - | 0.894 | 0.329 | 0.749 | - | 0.756 | 0.342 | 0.761 | - | 0.766 | 0.688 |
| bx_half | wor_half | 0.718 | 0.764 | 140 | 0.797 | 0.600 | 0.807 | - | 0.841 | 0.698 | 0.865 | 10 | 0.864 | 0.796 | 0.858 | 70 | 0.896 | 0.721 | 0.763 | 60 | 0.801 | 0.728 | 0.779 | 60 | 0.802 | 0.699 |
| wor_half | ban_wor | 0.714 | 0.663 | - | 0.682 | 0.761 | 0.794 | 44 | 0.788 | 0.805 | 0.834 | 50 | 0.830 | 0.834 | 0.835 | 300 | 0.846 | 0.714 | 0.676 | - | 0.709 | 0.712 | 0.670 | - | 0.708 | 0.679 |
| ban_wor | wor_half | 0.672 | 0.746 | 50 | 0.797 | 0.675 | 0.806 | 10 | 0.841 | 0.796 | 0.854 | 38 | 0.864 | 0.804 | 0.860 | 60 | 0.896 | 0.679 | 0.780 | 32 | 0.801 | 0.673 | 0.788 | 38 | 0.802 | 0.699 |

Figure B.2: Topic Books: NTL Results All Feature

| | Features | dense | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | logreg | | | | dectree | | | | randforest | | | | xgb | | | | svm | | | | logregcv | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.623 | 0.690 | 10 | 0.686 | 0.640 | 0.739 | 14 | 0.794 | 0.673 | 0.784 | 14 | 0.818 | 0.683 | 0.794 | 10 | 0.856 | 0.615 | 0.697 | 20 | 0.678 | 0.606 | 0.718 | 10 | 0.688 | 0.703 |
| bx_wor | ban_bx | 0.542 | 0.669 | 10 | 0.655 | 0.658 | 0.787 | - | 0.794 | 0.758 | 0.834 | 10 | 0.814 | 0.759 | 0.848 | 24 | 0.872 | 0.542 | 0.630 | 14 | 0.640 | 0.548 | 0.649 | 20 | 0.703 | 0.656 |
| ban_bx | ban_half | 0.715 | 0.870 | 10 | 0.889 | 0.903 | 0.952 | 14 | 0.945 | 0.951 | 0.960 | 100 | 0.969 | 0.924 | 0.956 | 14 | 0.960 | 0.678 | 0.900 | 10 | 0.897 | 0.655 | 0.921 | - | 0.914 | 0.761 |
| ban_half | ban_bx | 0.718 | 0.667 | - | 0.655 | 0.652 | 0.797 | - | 0.794 | 0.704 | 0.843 | 10 | 0.814 | 0.733 | 0.845 | 10 | 0.872 | 0.731 | 0.677 | - | 0.640 | 0.761 | 0.647 | - | 0.703 | 0.656 |
| ban_bx | bx_half | 0.580 | 0.734 | 14 | 0.731 | 0.873 | 0.866 | - | 0.866 | 0.891 | 0.894 | 160 | 0.895 | 0.860 | 0.890 | 10 | 0.894 | 0.552 | 0.756 | - | 0.756 | 0.596 | 0.756 | - | 0.766 | 0.688 |
| bx_half | ban_bx | 0.580 | 0.672 | 24 | 0.655 | 0.701 | 0.794 | - | 0.794 | 0.768 | 0.845 | 14 | 0.814 | 0.796 | 0.845 | 60 | 0.872 | 0.578 | 0.669 | 28 | 0.640 | 0.577 | 0.654 | 10 | 0.703 | 0.656 |
| ban_bx | ban_wor | 0.599 | 0.645 | 10 | 0.682 | 0.768 | 0.772 | 200 | 0.788 | 0.802 | 0.823 | 140 | 0.830 | 0.781 | 0.826 | 70 | 0.846 | 0.567 | 0.650 | 10 | 0.709 | 0.527 | 0.643 | - | 0.708 | 0.679 |
| ban_wor | ban_bx | 0.504 | 0.695 | - | 0.655 | 0.595 | 0.797 | - | 0.794 | 0.476 | 0.841 | - | 0.814 | 0.521 | 0.850 | - | 0.872 | 0.557 | 0.656 | 10 | 0.640 | 0.568 | 0.660 | 14 | 0.703 | 0.656 |
| bx_wor | wor_half | 0.691 | 0.726 | 100 | 0.797 | 0.704 | 0.800 | 10 | 0.841 | 0.694 | 0.838 | - | 0.864 | 0.714 | 0.844 | 10 | 0.896 | 0.683 | 0.772 | 50 | 0.801 | 0.692 | 0.762 | 38 | 0.802 | 0.699 |
| wor_half | bx_wor | 0.628 | 0.683 | 24 | 0.686 | 0.561 | 0.740 | - | 0.794 | 0.533 | 0.790 | - | 0.818 | 0.555 | 0.788 | - | 0.856 | 0.637 | 0.712 | 28 | 0.678 | 0.663 | 0.725 | 24 | 0.688 | 0.703 |
| bx_wor | bx_half | 0.755 | 0.731 | - | 0.731 | 0.760 | 0.870 | - | 0.866 | 0.785 | 0.894 | 10 | 0.895 | 0.803 | 0.895 | 10 | 0.894 | 0.753 | 0.754 | 300 | 0.756 | 0.754 | 0.758 | 180 | 0.766 | 0.688 |
| bx_half | bx_wor | 0.589 | 0.691 | - | 0.686 | 0.692 | 0.741 | 10 | 0.794 | 0.740 | 0.791 | 24 | 0.818 | 0.739 | 0.792 | 44 | 0.856 | 0.611 | 0.714 | 24 | 0.678 | 0.623 | 0.711 | 10 | 0.688 | 0.703 |
| bx_wor | ban_wor | 0.656 | 0.636 | - | 0.682 | 0.660 | 0.776 | - | 0.788 | 0.737 | 0.818 | 20 | 0.830 | 0.751 | 0.822 | 10 | 0.846 | 0.657 | 0.648 | - | 0.709 | 0.656 | 0.645 | - | 0.708 | 0.679 |
| ban_wor | bx_wor | 0.502 | 0.689 | - | 0.686 | 0.522 | 0.738 | - | 0.794 | 0.461 | 0.792 | - | 0.818 | 0.510 | 0.789 | - | 0.856 | 0.555 | 0.722 | 10 | 0.678 | 0.560 | 0.715 | - | 0.688 | 0.703 |
| ban_half | wor_half | 0.756 | 0.751 | - | 0.797 | 0.828 | 0.811 | - | 0.841 | 0.852 | 0.856 | 300 | 0.864 | 0.847 | 0.863 | 200 | 0.896 | 0.758 | 0.776 | 100 | 0.801 | 0.783 | 0.770 | - | 0.802 | 0.699 |
| wor_half | ban_half | 0.837 | 0.869 | 70 | 0.889 | 0.861 | 0.945 | - | 0.945 | 0.896 | 0.964 | 10 | 0.969 | 0.908 | 0.951 | 14 | 0.960 | 0.913 | 0.897 | - | 0.897 | 0.901 | 0.923 | 120 | 0.914 | 0.761 |
| ban_half | bx_half | 0.749 | 0.731 | - | 0.731 | 0.804 | 0.874 | 10 | 0.866 | 0.801 | 0.892 | - | 0.895 | 0.806 | 0.893 | 10 | 0.894 | 0.757 | 0.751 | - | 0.756 | 0.773 | 0.758 | - | 0.766 | 0.688 |
| bx_half | ban_half | 0.648 | 0.869 | - | 0.889 | 0.525 | 0.951 | - | 0.945 | 0.718 | 0.960 | - | 0.969 | 0.870 | 0.953 | 10 | 0.960 | 0.648 | 0.901 | - | 0.897 | 0.620 | 0.919 | - | 0.914 | 0.761 |
| ban_half | ban_wor | 0.682 | 0.670 | - | 0.682 | 0.733 | 0.784 | 32 | 0.788 | 0.800 | 0.837 | 32 | 0.830 | 0.782 | 0.833 | 24 | 0.846 | 0.696 | 0.674 | - | 0.709 | 0.716 | 0.676 | - | 0.708 | 0.679 |
| ban_wor | ban_half | 0.658 | 0.869 | - | 0.889 | 0.833 | 0.945 | - | 0.945 | 0.911 | 0.962 | 14 | 0.969 | 0.904 | 0.953 | 10 | 0.960 | 0.646 | 0.902 | - | 0.897 | 0.658 | 0.923 | - | 0.914 | 0.761 |
| wor_half | bx_half | 0.759 | 0.736 | - | 0.731 | 0.647 | 0.863 | - | 0.866 | 0.623 | 0.893 | - | 0.895 | 0.604 | 0.887 | - | 0.894 | 0.736 | 0.750 | 38 | 0.756 | 0.735 | 0.754 | 38 | 0.766 | 0.688 |
| bx_half | wor_half | 0.675 | 0.725 | 90 | 0.797 | 0.628 | 0.807 | - | 0.841 | 0.697 | 0.837 | - | 0.864 | 0.796 | 0.846 | 100 | 0.896 | 0.679 | 0.759 | 44 | 0.801 | 0.658 | 0.772 | 28 | 0.802 | 0.699 |
| wor_half | ban_wor | 0.687 | 0.659 | - | 0.682 | 0.784 | 0.790 | 200 | 0.788 | 0.814 | 0.835 | 80 | 0.830 | 0.825 | 0.841 | 300 | 0.846 | 0.712 | 0.668 | - | 0.709 | 0.715 | 0.673 | - | 0.708 | 0.679 |
| ban_wor | wor_half | 0.676 | 0.747 | 50 | 0.797 | 0.688 | 0.804 | 10 | 0.841 | 0.796 | 0.860 | 44 | 0.864 | 0.804 | 0.870 | 70 | 0.896 | 0.673 | 0.784 | 24 | 0.801 | 0.669 | 0.782 | 24 | 0.802 | 0.699 |

Figure B.3: Topic Books: NTL Results Dense Feature

| | Features | all | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | logreg | | | | dectree | | | | randforest | | | | xgb | | | | svm | | | | logregcv | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.651 | 0.681 | 180 | 0.727 | 0.661 | 0.692 | 300 | 0.786 | 0.697 | 0.775 | 120 | 0.822 | 0.686 | 0.766 | 140 | 0.851 | 0.661 | 0.705 | 140 | 0.727 | 0.661 | 0.689 | 140 | 0.729 | 0.652 |
| rewo_cdi | katom_cdi | 0.448 | 0.801 | - | 0.815 | 0.489 | 0.848 | - | 0.848 | 0.641 | 0.920 | 10 | 0.904 | 0.680 | 0.913 | 24 | 0.917 | 0.411 | 0.847 | - | 0.853 | 0.397 | 0.852 | - | 0.875 | 0.648 |
| katom_cdi | katom_rewo | 0.593 | 0.709 | 50 | 0.747 | 0.648 | 0.690 | 140 | 0.768 | 0.677 | 0.758 | 90 | 0.810 | 0.671 | 0.768 | 90 | 0.839 | 0.613 | 0.709 | 50 | 0.743 | 0.603 | 0.713 | 50 | 0.748 | 0.196 |
| katom_rewo | katom_cdi | 0.641 | 0.806 | 60 | 0.815 | 0.645 | 0.850 | 20 | 0.848 | 0.695 | 0.920 | 28 | 0.904 | 0.704 | 0.900 | 38 | 0.917 | 0.618 | 0.846 | 28 | 0.853 | 0.596 | 0.842 | 28 | 0.875 | 0.648 |
| rewo_cdi | katom_rewo | 0.604 | 0.705 | 60 | 0.747 | 0.543 | 0.692 | 10 | 0.768 | 0.639 | 0.757 | 38 | 0.810 | 0.601 | 0.762 | 38 | 0.839 | 0.595 | 0.721 | 44 | 0.743 | 0.589 | 0.709 | 50 | 0.748 | 0.196 |
| katom_rewo | rewo_cdi | 0.643 | 0.690 | 120 | 0.727 | 0.517 | 0.695 | 10 | 0.786 | 0.551 | 0.773 | 14 | 0.822 | 0.563 | 0.764 | 10 | 0.851 | 0.663 | 0.698 | 160 | 0.727 | 0.669 | 0.703 | 200 | 0.729 | 0.652 |

Figure B.4: Topic Kitchen: NTL Results All Feature

| | Features | dense | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Estimators | logreg | | | | dectree | | | | randforest | | | | xgb | | | | svm | | | | logregcv | | | | |
| | Results | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | TL_avg | Tar_max | Tar_exc | Tar_sup | unsuper_res |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.650 | 0.694 | 180 | 0.727 | 0.646 | 0.686 | 180 | 0.786 | 0.696 | 0.771 | 140 | 0.822 | 0.675 | 0.765 | 100 | 0.851 | 0.654 | 0.702 | 160 | 0.727 | 0.669 | 0.710 | 160 | 0.729 | 0.652 |
| rewo_cdi | katom_cdi | 0.427 | 0.798 | - | 0.815 | 0.439 | 0.846 | - | 0.848 | 0.655 | 0.916 | 10 | 0.904 | 0.678 | 0.908 | 20 | 0.917 | 0.378 | 0.841 | - | 0.853 | 0.373 | 0.848 | - | 0.875 | 0.648 |
| katom_cdi | katom_rewo | 0.588 | 0.694 | 50 | 0.747 | 0.658 | 0.682 | 180 | 0.768 | 0.672 | 0.759 | 70 | 0.810 | 0.668 | 0.762 | 80 | 0.839 | 0.604 | 0.712 | 50 | 0.743 | 0.565 | 0.706 | 28 | 0.748 | 0.196 |
| katom_rewo | katom_cdi | 0.621 | 0.803 | 32 | 0.815 | 0.610 | 0.842 | 14 | 0.848 | 0.706 | 0.913 | 24 | 0.904 | 0.669 | 0.906 | 28 | 0.917 | 0.597 | 0.839 | 20 | 0.853 | 0.592 | 0.847 | 24 | 0.875 | 0.648 |
| rewo_cdi | katom_rewo | 0.601 | 0.691 | 50 | 0.747 | 0.561 | 0.681 | 24 | 0.768 | 0.642 | 0.764 | 50 | 0.810 | 0.607 | 0.767 | 50 | 0.839 | 0.584 | 0.713 | 32 | 0.743 | 0.582 | 0.709 | 44 | 0.748 | 0.196 |
| katom_rewo | rewo_cdi | 0.640 | 0.683 | 140 | 0.727 | 0.505 | 0.703 | 10 | 0.786 | 0.546 | 0.766 | 14 | 0.822 | 0.518 | 0.766 | 10 | 0.851 | 0.660 | 0.705 | 160 | 0.727 | 0.658 | 0.699 | 160 | 0.729 | 0.652 |

Figure B.5: Topic Kitchen: NTL Results Dense Feature

## B.3 ATL Results

| | Estimator | rf | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | | | |
| | Iteration | 2nd | | | | 20th | | | | 50th | | | | 100th | | | | all |
| | Results | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | ATL | _NN | _LP | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.675 | 0.692 | 0.674 | 0.000 | 0.699 | 0.716 | 0.712 | 0.752 | 0.751 | 0.753 | 0.756 | 0.740 | 0.760 | 0.764 | 0.762 | 0.746 | **0.818** |
| bx_wor | ban_bx | 0.753 | 0.754 | 0.747 | 0.000 | 0.747 | 0.753 | 0.753 | 0.652 | 0.756 | 0.761 | 0.757 | 0.791 | 0.768 | 0.770 | 0.776 | 0.830 | **0.814** |
| ban_bx | ban_half | 0.951 | 0.943 | 0.946 | 0.000 | 0.953 | 0.944 | 0.953 | 0.925 | 0.955 | 0.943 | 0.953 | 0.947 | 0.957 | 0.948 | 0.961 | 0.953 | **0.969** |
| ban_half | ban_bx | 0.714 | 0.676 | 0.702 | 0.000 | 0.755 | 0.698 | 0.729 | 0.652 | 0.769 | 0.743 | 0.755 | 0.791 | 0.800 | 0.792 | 0.775 | 0.830 | **0.814** |
| ban_bx | bx_half | 0.891 | 0.888 | 0.892 | 0.000 | 0.889 | 0.890 | 0.892 | 0.838 | 0.888 | 0.881 | 0.890 | 0.853 | 0.885 | 0.889 | 0.893 | 0.869 | **0.895** |
| bx_half | ban_bx | 0.762 | 0.771 | 0.777 | 0.000 | 0.775 | 0.787 | 0.781 | 0.652 | 0.781 | 0.800 | 0.790 | 0.791 | 0.807 | 0.802 | 0.801 | 0.830 | **0.814** |
| ban_bx | ban_wor | 0.800 | 0.797 | 0.802 | 0.000 | 0.800 | 0.795 | 0.797 | 0.774 | 0.800 | 0.797 | 0.798 | 0.780 | 0.803 | 0.800 | 0.817 | 0.805 | **0.83** |
| ban_wor | ban_bx | 0.488 | 0.462 | 0.473 | 0.000 | 0.504 | 0.542 | 0.516 | 0.652 | 0.565 | 0.597 | 0.575 | 0.791 | 0.619 | 0.639 | 0.632 | 0.830 | **0.814** |
| bx_wor | wor_half | 0.698 | 0.701 | 0.692 | 0.000 | 0.709 | 0.711 | 0.701 | 0.823 | 0.723 | 0.735 | 0.725 | 0.830 | 0.748 | 0.765 | 0.758 | 0.807 | **0.864** |
| wor_half | bx_wor | 0.531 | 0.548 | 0.537 | 0.000 | 0.567 | 0.610 | 0.571 | 0.752 | 0.613 | 0.647 | 0.626 | 0.740 | 0.662 | 0.685 | 0.671 | 0.746 | **0.818** |
| bx_wor | bx_half | 0.792 | 0.805 | 0.786 | 0.000 | 0.803 | 0.818 | 0.805 | 0.838 | 0.812 | 0.832 | 0.822 | 0.853 | 0.833 | 0.854 | 0.837 | 0.869 | **0.895** |
| bx_half | bx_wor | 0.742 | 0.740 | 0.739 | 0.000 | 0.744 | 0.748 | 0.745 | 0.752 | 0.745 | 0.748 | 0.749 | 0.740 | 0.745 | 0.755 | 0.754 | 0.746 | **0.818** |
| bx_wor | ban_wor | 0.730 | 0.730 | 0.724 | 0.000 | 0.735 | 0.740 | 0.729 | 0.774 | 0.746 | 0.746 | 0.740 | 0.780 | 0.747 | 0.764 | 0.755 | 0.805 | **0.83** |
| ban_wor | bx_wor | 0.456 | 0.454 | 0.455 | 0.000 | 0.498 | 0.510 | 0.486 | 0.752 | 0.547 | 0.539 | 0.531 | 0.740 | 0.587 | 0.589 | 0.581 | 0.746 | **0.818** |
| ban_half | wor_half | 0.855 | 0.843 | 0.856 | 0.000 | 0.852 | 0.842 | 0.855 | 0.823 | 0.856 | 0.844 | 0.856 | 0.830 | 0.853 | 0.848 | 0.857 | 0.807 | **0.864** |
| wor_half | ban_half | 0.905 | 0.897 | 0.897 | 0.000 | 0.927 | 0.920 | 0.926 | 0.925 | 0.943 | 0.940 | 0.938 | 0.947 | 0.953 | 0.949 | 0.952 | 0.953 | **0.969** |
| ban_half | bx_half | 0.815 | 0.781 | 0.798 | 0.000 | 0.825 | 0.837 | 0.823 | 0.838 | 0.836 | 0.841 | 0.842 | 0.853 | 0.846 | 0.856 | 0.858 | 0.869 | **0.895** |
| bx_half | ban_half | 0.747 | 0.841 | 0.861 | 0.000 | 0.883 | 0.908 | 0.899 | 0.925 | 0.907 | 0.922 | 0.917 | 0.947 | 0.930 | 0.937 | 0.940 | 0.953 | **0.969** |
| ban_half | ban_wor | 0.803 | 0.804 | 0.807 | 0.000 | 0.809 | 0.811 | 0.808 | 0.774 | 0.816 | 0.820 | 0.818 | 0.780 | 0.820 | 0.819 | 0.819 | 0.805 | **0.83** |
| ban_wor | ban_half | 0.913 | 0.899 | 0.900 | 0.000 | 0.909 | 0.903 | 0.903 | 0.925 | 0.915 | 0.912 | 0.911 | 0.947 | 0.919 | 0.921 | 0.919 | 0.953 | **0.969** |
| wor_half | bx_half | 0.624 | 0.648 | 0.625 | 0.000 | 0.740 | 0.777 | 0.730 | 0.838 | 0.790 | 0.806 | 0.799 | 0.853 | 0.821 | 0.827 | 0.838 | 0.869 | **0.895** |
| bx_half | wor_half | 0.725 | 0.764 | 0.737 | 0.000 | 0.752 | 0.781 | 0.764 | 0.823 | 0.778 | 0.792 | 0.781 | 0.830 | 0.794 | 0.801 | 0.789 | 0.807 | **0.864** |
| wor_half | ban_wor | 0.805 | 0.818 | 0.802 | 0.000 | 0.807 | 0.815 | 0.805 | 0.774 | 0.819 | 0.815 | 0.815 | 0.780 | 0.825 | 0.823 | 0.818 | 0.805 | **0.83** |
| ban_wor | wor_half | 0.800 | 0.797 | 0.794 | 0.000 | 0.797 | 0.789 | 0.799 | 0.823 | 0.794 | 0.799 | 0.805 | 0.830 | 0.801 | 0.805 | 0.811 | 0.807 | **0.864** |

Figure B.6: Topic Books - ATL results Domain Adaptation Techniques compared (all combinations)

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | random | | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.675 | 0.000 | 0.684 | 0.639 | 0.699 | 0.752 | 0.738 | 0.747 | 0.751 | 0.740 | 0.760 | 0.746 | **0.818** | 0.678 | 0.000 | 0.698 | 0.608 | 0.741 | 0.666 | 0.752 | 0.710 | 0.757 | 0.724 | 0.762 | 0.755 | **0.818** |
| bx_wor | ban_bx | 0.753 | 0.000 | 0.751 | 0.639 | 0.747 | 0.652 | 0.753 | 0.772 | 0.756 | 0.791 | 0.768 | 0.830 | **0.814** | 0.751 | 0.000 | 0.750 | 0.753 | 0.752 | 0.779 | 0.753 | 0.820 | 0.754 | 0.812 | 0.761 | 0.823 | **0.814** |
| ban_bx | ban_half | 0.951 | 0.000 | 0.949 | 0.749 | 0.953 | 0.925 | 0.954 | 0.944 | 0.955 | 0.947 | 0.957 | 0.953 | **0.969** | 0.950 | 0.000 | 0.946 | 0.855 | 0.944 | 0.929 | 0.944 | 0.946 | 0.949 | 0.947 | 0.945 | 0.954 | **0.969** |
| ban_half | ban_bx | 0.714 | 0.000 | 0.739 | 0.639 | 0.755 | 0.652 | 0.756 | 0.772 | 0.769 | 0.791 | 0.800 | 0.830 | **0.814** | 0.723 | 0.000 | 0.734 | 0.753 | 0.744 | 0.779 | 0.758 | 0.820 | 0.769 | 0.812 | 0.788 | 0.823 | **0.814** |
| ban_bx | bx_half | 0.891 | 0.000 | 0.889 | 0.788 | 0.889 | 0.838 | 0.889 | 0.857 | 0.888 | 0.853 | 0.885 | 0.869 | **0.895** | 0.892 | 0.000 | 0.892 | 0.768 | 0.894 | 0.813 | 0.895 | 0.838 | 0.893 | 0.844 | 0.895 | 0.872 | **0.895** |
| bx_half | ban_bx | 0.762 | 0.000 | 0.769 | 0.639 | 0.775 | 0.652 | 0.780 | 0.772 | 0.781 | 0.791 | 0.807 | 0.830 | **0.814** | 0.770 | 0.000 | 0.783 | 0.753 | 0.785 | 0.779 | 0.788 | 0.820 | 0.796 | 0.812 | 0.799 | 0.823 | **0.814** |
| ban_bx | ban_wor | 0.800 | 0.000 | 0.797 | 0.543 | 0.800 | 0.774 | 0.803 | 0.754 | 0.800 | 0.780 | 0.803 | 0.805 | **0.83** | 0.800 | 0.000 | 0.805 | 0.800 | 0.801 | 0.798 | 0.806 | 0.797 | 0.805 | 0.795 | 0.809 | 0.817 | **0.83** |
| ban_wor | ban_bx | 0.488 | 0.000 | 0.493 | 0.639 | 0.504 | 0.652 | 0.542 | 0.772 | 0.565 | 0.791 | 0.619 | 0.830 | **0.814** | 0.487 | 0.000 | 0.506 | 0.753 | 0.517 | 0.779 | 0.536 | 0.820 | 0.579 | 0.812 | 0.642 | 0.823 | **0.814** |
| bx_wor | wor_half | 0.698 | 0.000 | 0.705 | 0.790 | 0.709 | 0.823 | 0.712 | 0.824 | 0.723 | 0.830 | 0.748 | 0.807 | **0.864** | 0.701 | 0.000 | 0.703 | 0.707 | 0.702 | 0.782 | 0.707 | 0.782 | 0.716 | 0.792 | 0.729 | 0.804 | **0.864** |
| wor_half | bx_wor | 0.531 | 0.000 | 0.546 | 0.639 | 0.567 | 0.752 | 0.586 | 0.747 | 0.613 | 0.740 | 0.662 | 0.746 | **0.818** | 0.537 | 0.000 | 0.559 | 0.608 | 0.583 | 0.666 | 0.591 | 0.710 | 0.616 | 0.724 | 0.651 | 0.755 | **0.818** |
| bx_wor | bx_half | 0.792 | 0.000 | 0.801 | 0.788 | 0.803 | 0.838 | 0.806 | 0.857 | 0.812 | 0.853 | 0.833 | 0.869 | **0.895** | 0.790 | 0.000 | 0.798 | 0.768 | 0.797 | 0.813 | 0.800 | 0.838 | 0.803 | 0.844 | 0.816 | 0.872 | **0.895** |
| bx_half | bx_wor | 0.742 | 0.000 | 0.741 | 0.639 | 0.744 | 0.752 | 0.745 | 0.747 | 0.745 | 0.740 | 0.745 | 0.746 | **0.818** | 0.741 | 0.000 | 0.743 | 0.608 | 0.748 | 0.666 | 0.744 | 0.710 | 0.748 | 0.724 | 0.750 | 0.755 | **0.818** |
| bx_wor | ban_wor | 0.730 | 0.000 | 0.731 | 0.543 | 0.735 | 0.774 | 0.737 | 0.754 | 0.746 | 0.780 | 0.747 | 0.805 | **0.83** | 0.735 | 0.000 | 0.727 | 0.800 | 0.729 | 0.798 | 0.733 | 0.797 | 0.732 | 0.795 | 0.740 | 0.817 | **0.83** |
| ban_wor | bx_wor | 0.456 | 0.000 | 0.471 | 0.639 | 0.498 | 0.752 | 0.518 | 0.747 | 0.547 | 0.740 | 0.587 | 0.746 | **0.818** | 0.460 | 0.000 | 0.470 | 0.608 | 0.483 | 0.666 | 0.489 | 0.710 | 0.512 | 0.724 | 0.550 | 0.755 | **0.818** |
| ban_half | wor_half | 0.855 | 0.000 | 0.853 | 0.790 | 0.852 | 0.823 | 0.853 | 0.824 | 0.856 | 0.830 | 0.853 | 0.807 | **0.864** | 0.851 | 0.000 | 0.850 | 0.707 | 0.852 | 0.782 | 0.850 | 0.782 | 0.851 | 0.792 | 0.854 | 0.804 | **0.864** |
| wor_half | ban_half | 0.905 | 0.000 | 0.915 | 0.749 | 0.927 | 0.925 | 0.934 | 0.944 | 0.943 | 0.947 | 0.953 | 0.953 | **0.969** | 0.896 | 0.000 | 0.904 | 0.855 | 0.905 | 0.929 | 0.910 | 0.946 | 0.926 | 0.947 | 0.941 | 0.954 | **0.969** |
| ban_half | bx_half | 0.815 | 0.000 | 0.832 | 0.788 | 0.825 | 0.838 | 0.830 | 0.857 | 0.836 | 0.853 | 0.846 | 0.869 | **0.895** | 0.812 | 0.000 | 0.798 | 0.768 | 0.809 | 0.813 | 0.829 | 0.838 | 0.835 | 0.844 | 0.855 | 0.872 | **0.895** |
| bx_half | ban_half | 0.747 | 0.000 | 0.873 | 0.749 | 0.883 | 0.925 | 0.896 | 0.944 | 0.907 | 0.947 | 0.930 | 0.953 | **0.969** | 0.782 | 0.000 | 0.878 | 0.855 | 0.887 | 0.929 | 0.902 | 0.946 | 0.916 | 0.947 | 0.925 | 0.954 | **0.969** |
| ban_half | ban_wor | 0.803 | 0.000 | 0.808 | 0.543 | 0.809 | 0.774 | 0.812 | 0.754 | 0.816 | 0.780 | 0.820 | 0.805 | **0.83** | 0.800 | 0.000 | 0.808 | 0.800 | 0.811 | 0.798 | 0.811 | 0.797 | 0.817 | 0.795 | 0.819 | 0.817 | **0.83** |
| ban_wor | ban_half | 0.913 | 0.000 | 0.909 | 0.749 | 0.909 | 0.925 | 0.915 | 0.944 | 0.915 | 0.947 | 0.919 | 0.953 | **0.969** | 0.914 | 0.000 | 0.916 | 0.855 | 0.912 | 0.929 | 0.910 | 0.946 | 0.911 | 0.947 | 0.918 | 0.954 | **0.969** |
| wor_half | bx_half | 0.624 | 0.000 | 0.686 | 0.788 | 0.740 | 0.838 | 0.766 | 0.857 | 0.790 | 0.853 | 0.821 | 0.869 | **0.895** | 0.621 | 0.000 | 0.651 | 0.768 | 0.658 | 0.813 | 0.680 | 0.838 | 0.743 | 0.844 | 0.794 | 0.872 | **0.895** |
| bx_half | wor_half | 0.725 | 0.000 | 0.767 | 0.790 | 0.752 | 0.823 | 0.759 | 0.824 | 0.778 | 0.830 | 0.794 | 0.807 | **0.864** | 0.715 | 0.000 | 0.768 | 0.707 | 0.764 | 0.782 | 0.793 | 0.782 | 0.800 | 0.792 | 0.807 | 0.804 | **0.864** |
| wor_half | ban_wor | 0.805 | 0.000 | 0.807 | 0.543 | 0.807 | 0.774 | 0.813 | 0.754 | 0.819 | 0.780 | 0.825 | 0.805 | **0.83** | 0.808 | 0.000 | 0.809 | 0.800 | 0.805 | 0.798 | 0.806 | 0.797 | 0.815 | 0.795 | 0.822 | 0.817 | **0.83** |
| ban_wor | wor_half | 0.800 | 0.000 | 0.793 | 0.790 | 0.797 | 0.823 | 0.796 | 0.824 | 0.794 | 0.830 | 0.801 | 0.807 | **0.864** | 0.797 | 0.000 | 0.795 | 0.707 | 0.801 | 0.782 | 0.799 | 0.782 | 0.800 | 0.792 | 0.807 | 0.804 | **0.864** |

Figure B.7: Topic Books - ATL results with no Domain Adaptation

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | random | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.692 | 0.000 | 0.700 | 0.639 | 0.716 | 0.752 | 0.727 | 0.747 | 0.753 | 0.740 | 0.764 | 0.746 | **0.818** | 0.705 | 0.000 | 0.720 | 0.608 | 0.743 | 0.666 | 0.751 | 0.710 | 0.756 | 0.724 | 0.760 | 0.755 | **0.818** |
| bx_wor | ban_bx | 0.754 | 0.000 | 0.750 | 0.639 | 0.753 | 0.652 | 0.759 | 0.772 | 0.761 | 0.791 | 0.770 | 0.830 | **0.814** | 0.753 | 0.000 | 0.749 | 0.753 | 0.760 | 0.779 | 0.761 | 0.820 | 0.771 | 0.812 | 0.781 | 0.823 | **0.814** |
| ban_bx | ban_half | 0.943 | 0.000 | 0.940 | 0.749 | 0.944 | 0.925 | 0.945 | 0.944 | 0.943 | 0.947 | 0.948 | 0.953 | **0.969** | 0.944 | 0.000 | 0.949 | 0.855 | 0.948 | 0.929 | 0.948 | 0.946 | 0.948 | 0.947 | 0.948 | 0.954 | **0.969** |
| ban_half | ban_bx | 0.676 | 0.000 | 0.702 | 0.639 | 0.698 | 0.652 | 0.733 | 0.772 | 0.743 | 0.791 | 0.792 | 0.830 | **0.814** | 0.692 | 0.000 | 0.708 | 0.753 | 0.737 | 0.779 | 0.747 | 0.820 | 0.750 | 0.812 | 0.781 | 0.823 | **0.814** |
| ban_bx | bx_half | 0.888 | 0.000 | 0.893 | 0.788 | 0.890 | 0.838 | 0.892 | 0.857 | 0.881 | 0.853 | 0.889 | 0.869 | **0.895** | 0.891 | 0.000 | 0.893 | 0.768 | 0.891 | 0.813 | 0.892 | 0.838 | 0.891 | 0.844 | 0.892 | 0.872 | **0.895** |
| bx_half | ban_bx | 0.771 | 0.000 | 0.767 | 0.639 | 0.787 | 0.652 | 0.783 | 0.772 | 0.800 | 0.791 | 0.802 | 0.830 | **0.814** | 0.760 | 0.000 | 0.774 | 0.753 | 0.786 | 0.779 | 0.789 | 0.820 | 0.784 | 0.812 | 0.803 | 0.823 | **0.814** |
| ban_bx | ban_wor | 0.797 | 0.000 | 0.790 | 0.543 | 0.795 | 0.774 | 0.797 | 0.754 | 0.797 | 0.780 | 0.800 | 0.805 | **0.83** | 0.801 | 0.000 | 0.799 | 0.800 | 0.806 | 0.798 | 0.802 | 0.797 | 0.802 | 0.795 | 0.803 | 0.817 | **0.83** |
| ban_wor | ban_bx | 0.462 | 0.000 | 0.498 | 0.639 | 0.542 | 0.652 | 0.558 | 0.772 | 0.597 | 0.791 | 0.639 | 0.830 | **0.814** | 0.470 | 0.000 | 0.495 | 0.753 | 0.518 | 0.779 | 0.546 | 0.820 | 0.608 | 0.812 | 0.659 | 0.823 | **0.814** |
| bx_wor | wor_half | 0.701 | 0.000 | 0.707 | 0.790 | 0.711 | 0.823 | 0.723 | 0.824 | 0.735 | 0.830 | 0.765 | 0.807 | **0.864** | 0.706 | 0.000 | 0.713 | 0.707 | 0.706 | 0.782 | 0.706 | 0.782 | 0.710 | 0.792 | 0.747 | 0.804 | **0.864** |
| wor_half | bx_wor | 0.548 | 0.000 | 0.590 | 0.639 | 0.610 | 0.752 | 0.621 | 0.747 | 0.647 | 0.740 | 0.685 | 0.746 | **0.818** | 0.546 | 0.000 | 0.592 | 0.608 | 0.600 | 0.666 | 0.615 | 0.710 | 0.642 | 0.724 | 0.672 | 0.755 | **0.818** |
| bx_wor | bx_half | 0.805 | 0.000 | 0.807 | 0.788 | 0.818 | 0.838 | 0.828 | 0.857 | 0.832 | 0.853 | 0.854 | 0.869 | **0.895** | 0.799 | 0.000 | 0.809 | 0.768 | 0.816 | 0.813 | 0.820 | 0.838 | 0.825 | 0.844 | 0.835 | 0.872 | **0.895** |
| bx_half | bx_wor | 0.740 | 0.000 | 0.743 | 0.639 | 0.748 | 0.752 | 0.748 | 0.747 | 0.748 | 0.740 | 0.755 | 0.746 | **0.818** | 0.745 | 0.000 | 0.743 | 0.608 | 0.741 | 0.666 | 0.741 | 0.710 | 0.747 | 0.724 | 0.749 | 0.755 | **0.818** |
| bx_wor | ban_wor | 0.730 | 0.000 | 0.731 | 0.543 | 0.740 | 0.774 | 0.742 | 0.754 | 0.746 | 0.780 | 0.764 | 0.805 | **0.83** | 0.732 | 0.000 | 0.727 | 0.800 | 0.732 | 0.798 | 0.735 | 0.797 | 0.735 | 0.795 | 0.743 | 0.817 | **0.83** |
| ban_wor | bx_wor | 0.454 | 0.000 | 0.479 | 0.639 | 0.510 | 0.752 | 0.520 | 0.747 | 0.539 | 0.740 | 0.589 | 0.746 | **0.818** | 0.454 | 0.000 | 0.475 | 0.608 | 0.495 | 0.666 | 0.507 | 0.710 | 0.529 | 0.724 | 0.594 | 0.755 | **0.818** |
| ban_half | wor_half | 0.843 | 0.000 | 0.844 | 0.790 | 0.842 | 0.823 | 0.845 | 0.824 | 0.844 | 0.830 | 0.848 | 0.807 | **0.864** | 0.848 | 0.000 | 0.845 | 0.707 | 0.844 | 0.782 | 0.841 | 0.782 | 0.841 | 0.792 | 0.848 | 0.804 | **0.864** |
| wor_half | ban_half | 0.897 | 0.000 | 0.908 | 0.749 | 0.920 | 0.925 | 0.935 | 0.944 | 0.940 | 0.947 | 0.949 | 0.953 | **0.969** | 0.902 | 0.000 | 0.914 | 0.855 | 0.911 | 0.929 | 0.909 | 0.946 | 0.919 | 0.947 | 0.934 | 0.954 | **0.969** |
| ban_half | bx_half | 0.781 | 0.000 | 0.798 | 0.788 | 0.837 | 0.838 | 0.837 | 0.857 | 0.841 | 0.853 | 0.856 | 0.869 | **0.895** | 0.790 | 0.000 | 0.809 | 0.768 | 0.833 | 0.813 | 0.851 | 0.838 | 0.855 | 0.844 | 0.859 | 0.872 | **0.895** |
| bx_half | ban_half | 0.841 | 0.000 | 0.881 | 0.749 | 0.908 | 0.925 | 0.910 | 0.944 | 0.922 | 0.947 | 0.937 | 0.953 | **0.969** | 0.832 | 0.000 | 0.891 | 0.855 | 0.904 | 0.929 | 0.919 | 0.946 | 0.918 | 0.947 | 0.930 | 0.954 | **0.969** |
| ban_half | ban_wor | 0.804 | 0.000 | 0.808 | 0.543 | 0.811 | 0.774 | 0.815 | 0.754 | 0.820 | 0.780 | 0.819 | 0.805 | **0.83** | 0.810 | 0.000 | 0.810 | 0.800 | 0.814 | 0.798 | 0.816 | 0.797 | 0.819 | 0.795 | 0.820 | 0.817 | **0.83** |
| ban_wor | ban_half | 0.899 | 0.000 | 0.905 | 0.749 | 0.903 | 0.925 | 0.910 | 0.944 | 0.912 | 0.947 | 0.921 | 0.953 | **0.969** | 0.907 | 0.000 | 0.903 | 0.855 | 0.904 | 0.929 | 0.908 | 0.946 | 0.910 | 0.947 | 0.919 | 0.954 | **0.969** |
| wor_half | bx_half | 0.648 | 0.000 | 0.713 | 0.788 | 0.777 | 0.838 | 0.779 | 0.857 | 0.806 | 0.853 | 0.827 | 0.869 | **0.895** | 0.665 | 0.000 | 0.686 | 0.768 | 0.726 | 0.813 | 0.745 | 0.838 | 0.773 | 0.844 | 0.834 | 0.872 | **0.895** |
| bx_half | wor_half | 0.764 | 0.000 | 0.775 | 0.790 | 0.781 | 0.823 | 0.800 | 0.824 | 0.792 | 0.830 | 0.801 | 0.807 | **0.864** | 0.737 | 0.000 | 0.731 | 0.707 | 0.785 | 0.782 | 0.780 | 0.782 | 0.803 | 0.792 | 0.818 | 0.804 | **0.864** |
| wor_half | ban_wor | 0.818 | 0.000 | 0.810 | 0.543 | 0.815 | 0.774 | 0.817 | 0.754 | 0.815 | 0.780 | 0.823 | 0.805 | **0.83** | 0.810 | 0.000 | 0.806 | 0.800 | 0.810 | 0.798 | 0.811 | 0.797 | 0.812 | 0.795 | 0.821 | 0.817 | **0.83** |
| ban_wor | wor_half | 0.797 | 0.000 | 0.791 | 0.790 | 0.789 | 0.823 | 0.784 | 0.824 | 0.799 | 0.830 | 0.805 | 0.807 | **0.864** | 0.796 | 0.000 | 0.800 | 0.707 | 0.792 | 0.782 | 0.798 | 0.782 | 0.800 | 0.792 | 0.814 | 0.804 | **0.864** |

Figure B.8: Topic Books - ATL results with Nearest-neighbor Weighting

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | random | | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ban_bx | bx_wor | 0.674 | 0.000 | 0.685 | 0.639 | 0.712 | 0.752 | 0.740 | 0.747 | 0.756 | 0.740 | 0.762 | 0.746 | **0.818** | 0.680 | 0.000 | 0.713 | 0.608 | 0.735 | 0.666 | 0.745 | 0.710 | 0.751 | 0.724 | 0.764 | 0.755 | **0.818** |
| bx_wor | ban_bx | 0.747 | 0.000 | 0.749 | 0.639 | 0.753 | 0.652 | 0.747 | 0.772 | 0.757 | 0.791 | 0.776 | 0.830 | **0.814** | 0.746 | 0.000 | 0.752 | 0.753 | 0.749 | 0.779 | 0.747 | 0.820 | 0.761 | 0.812 | 0.765 | 0.823 | **0.814** |
| ban_bx | ban_half | 0.946 | 0.000 | 0.949 | 0.749 | 0.953 | 0.925 | 0.953 | 0.944 | 0.953 | 0.947 | 0.961 | 0.953 | **0.969** | 0.946 | 0.000 | 0.947 | 0.855 | 0.948 | 0.929 | 0.950 | 0.946 | 0.949 | 0.947 | 0.948 | 0.954 | **0.969** |
| ban_half | ban_bx | 0.702 | 0.000 | 0.730 | 0.639 | 0.729 | 0.652 | 0.743 | 0.772 | 0.755 | 0.791 | 0.775 | 0.830 | **0.814** | 0.702 | 0.000 | 0.709 | 0.753 | 0.724 | 0.779 | 0.734 | 0.820 | 0.751 | 0.812 | 0.769 | 0.823 | **0.814** |
| ban_bx | bx_half | 0.892 | 0.000 | 0.892 | 0.788 | 0.892 | 0.838 | 0.894 | 0.857 | 0.890 | 0.853 | 0.893 | 0.869 | **0.895** | 0.890 | 0.000 | 0.897 | 0.768 | 0.893 | 0.813 | 0.893 | 0.838 | 0.889 | 0.844 | 0.891 | 0.872 | **0.895** |
| bx_half | ban_bx | 0.777 | 0.000 | 0.778 | 0.639 | 0.781 | 0.652 | 0.784 | 0.772 | 0.790 | 0.791 | 0.801 | 0.830 | **0.814** | 0.776 | 0.000 | 0.773 | 0.753 | 0.776 | 0.779 | 0.777 | 0.820 | 0.783 | 0.812 | 0.806 | 0.823 | **0.814** |
| ban_bx | ban_wor | 0.802 | 0.000 | 0.798 | 0.543 | 0.797 | 0.774 | 0.802 | 0.754 | 0.798 | 0.780 | 0.817 | 0.805 | **0.83** | 0.805 | 0.000 | 0.798 | 0.800 | 0.803 | 0.798 | 0.804 | 0.797 | 0.805 | 0.795 | 0.807 | 0.817 | **0.83** |
| ban_wor | ban_bx | 0.473 | 0.000 | 0.504 | 0.639 | 0.516 | 0.652 | 0.539 | 0.772 | 0.575 | 0.791 | 0.632 | 0.830 | **0.814** | 0.471 | 0.000 | 0.502 | 0.753 | 0.506 | 0.779 | 0.540 | 0.820 | 0.564 | 0.812 | 0.618 | 0.823 | **0.814** |
| bx_wor | wor_half | 0.692 | 0.000 | 0.699 | 0.790 | 0.701 | 0.823 | 0.706 | 0.824 | 0.725 | 0.830 | 0.758 | 0.807 | **0.864** | 0.695 | 0.000 | 0.700 | 0.707 | 0.705 | 0.782 | 0.712 | 0.782 | 0.723 | 0.792 | 0.741 | 0.804 | **0.864** |
| wor_half | bx_wor | 0.537 | 0.000 | 0.557 | 0.639 | 0.571 | 0.752 | 0.607 | 0.747 | 0.626 | 0.740 | 0.671 | 0.746 | **0.818** | 0.539 | 0.000 | 0.550 | 0.608 | 0.568 | 0.666 | 0.582 | 0.710 | 0.617 | 0.724 | 0.655 | 0.755 | **0.818** |
| bx_wor | bx_half | 0.786 | 0.000 | 0.801 | 0.788 | 0.805 | 0.838 | 0.805 | 0.857 | 0.822 | 0.853 | 0.837 | 0.869 | **0.895** | 0.785 | 0.000 | 0.795 | 0.768 | 0.801 | 0.813 | 0.805 | 0.838 | 0.807 | 0.844 | 0.816 | 0.872 | **0.895** |
| bx_half | bx_wor | 0.739 | 0.000 | 0.743 | 0.639 | 0.745 | 0.752 | 0.744 | 0.747 | 0.749 | 0.740 | 0.754 | 0.746 | **0.818** | 0.741 | 0.000 | 0.737 | 0.608 | 0.738 | 0.666 | 0.741 | 0.710 | 0.746 | 0.724 | 0.748 | 0.755 | **0.818** |
| bx_wor | ban_wor | 0.724 | 0.000 | 0.726 | 0.543 | 0.729 | 0.774 | 0.732 | 0.754 | 0.740 | 0.780 | 0.755 | 0.805 | **0.83** | 0.722 | 0.000 | 0.725 | 0.800 | 0.725 | 0.798 | 0.730 | 0.797 | 0.736 | 0.795 | 0.740 | 0.817 | **0.83** |
| ban_wor | bx_wor | 0.455 | 0.000 | 0.470 | 0.639 | 0.486 | 0.752 | 0.508 | 0.747 | 0.531 | 0.740 | 0.581 | 0.746 | **0.818** | 0.452 | 0.000 | 0.467 | 0.608 | 0.482 | 0.666 | 0.494 | 0.710 | 0.524 | 0.724 | 0.553 | 0.755 | **0.818** |
| ban_half | wor_half | 0.856 | 0.000 | 0.852 | 0.790 | 0.855 | 0.823 | 0.853 | 0.824 | 0.856 | 0.830 | 0.857 | 0.807 | **0.864** | 0.855 | 0.000 | 0.854 | 0.707 | 0.854 | 0.782 | 0.848 | 0.782 | 0.846 | 0.792 | 0.847 | 0.804 | **0.864** |
| wor_half | ban_half | 0.897 | 0.000 | 0.913 | 0.749 | 0.926 | 0.925 | 0.928 | 0.944 | 0.938 | 0.947 | 0.952 | 0.953 | **0.969** | 0.902 | 0.000 | 0.900 | 0.855 | 0.910 | 0.929 | 0.915 | 0.946 | 0.922 | 0.947 | 0.938 | 0.954 | **0.969** |
| ban_half | bx_half | 0.798 | 0.000 | 0.799 | 0.788 | 0.823 | 0.838 | 0.818 | 0.857 | 0.842 | 0.853 | 0.858 | 0.869 | **0.895** | 0.795 | 0.000 | 0.795 | 0.768 | 0.820 | 0.813 | 0.813 | 0.838 | 0.831 | 0.844 | 0.841 | 0.872 | **0.895** |
| bx_half | ban_half | 0.861 | 0.000 | 0.882 | 0.749 | 0.899 | 0.925 | 0.910 | 0.944 | 0.917 | 0.947 | 0.940 | 0.953 | **0.969** | 0.780 | 0.000 | 0.883 | 0.855 | 0.892 | 0.929 | 0.896 | 0.946 | 0.913 | 0.947 | 0.930 | 0.954 | **0.969** |
| ban_half | ban_wor | 0.807 | 0.000 | 0.807 | 0.543 | 0.808 | 0.774 | 0.808 | 0.754 | 0.818 | 0.780 | 0.819 | 0.805 | **0.83** | 0.804 | 0.000 | 0.804 | 0.800 | 0.811 | 0.798 | 0.816 | 0.797 | 0.818 | 0.795 | 0.825 | 0.817 | **0.83** |
| ban_wor | ban_half | 0.900 | 0.000 | 0.905 | 0.749 | 0.903 | 0.925 | 0.909 | 0.944 | 0.911 | 0.947 | 0.919 | 0.953 | **0.969** | 0.904 | 0.000 | 0.904 | 0.855 | 0.902 | 0.929 | 0.903 | 0.946 | 0.906 | 0.947 | 0.911 | 0.954 | **0.969** |
| wor_half | bx_half | 0.625 | 0.000 | 0.675 | 0.788 | 0.730 | 0.838 | 0.762 | 0.857 | 0.799 | 0.853 | 0.838 | 0.869 | **0.895** | 0.631 | 0.000 | 0.679 | 0.768 | 0.729 | 0.813 | 0.755 | 0.838 | 0.774 | 0.844 | 0.818 | 0.872 | **0.895** |
| bx_half | wor_half | 0.737 | 0.000 | 0.766 | 0.790 | 0.764 | 0.823 | 0.767 | 0.824 | 0.781 | 0.830 | 0.789 | 0.807 | **0.864** | 0.756 | 0.000 | 0.760 | 0.707 | 0.785 | 0.782 | 0.787 | 0.782 | 0.772 | 0.792 | 0.804 | 0.804 | **0.864** |
| wor_half | ban_wor | 0.802 | 0.000 | 0.808 | 0.543 | 0.805 | 0.774 | 0.806 | 0.754 | 0.815 | 0.780 | 0.818 | 0.805 | **0.83** | 0.803 | 0.000 | 0.810 | 0.800 | 0.811 | 0.798 | 0.813 | 0.797 | 0.818 | 0.795 | 0.820 | 0.817 | **0.83** |
| ban_wor | wor_half | 0.794 | 0.000 | 0.800 | 0.790 | 0.799 | 0.823 | 0.795 | 0.824 | 0.805 | 0.830 | 0.811 | 0.807 | **0.864** | 0.796 | 0.000 | 0.792 | 0.707 | 0.801 | 0.782 | 0.802 | 0.782 | 0.799 | 0.792 | 0.809 | 0.804 | **0.864** |

Figure B.9: Topic Books - ATL results with Logistic Regression Weighting

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | random | | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.692 | 0.000 | 0.692 | 0.519 | 0.681 | 0.534 | 0.694 | 0.562 | 0.699 | 0.623 | 0.717 | 0.692 | **0.822** | 0.689 | 0.000 | 0.689 | 0.461 | 0.687 | 0.601 | 0.694 | 0.584 | 0.697 | 0.593 | 0.707 | 0.682 | **0.822** |
| rewo_cdi | katom_cdi | 0.695 | 0.000 | 0.677 | 0.551 | 0.725 | 0.675 | 0.748 | 0.706 | 0.787 | 0.744 | 0.822 | 0.846 | **0.904** | 0.692 | 0.000 | 0.723 | 0.568 | 0.722 | 0.663 | 0.777 | 0.735 | 0.801 | 0.778 | 0.820 | 0.820 | **0.904** |
| katom_cdi | katom_rewo | 0.675 | 0.000 | 0.677 | 0.513 | 0.695 | 0.567 | 0.703 | 0.614 | 0.713 | 0.655 | 0.718 | 0.697 | **0.81** | 0.674 | 0.000 | 0.684 | 0.537 | 0.681 | 0.533 | 0.695 | 0.577 | 0.697 | 0.618 | 0.692 | 0.680 | **0.81** |
| katom_rewo | katom_cdi | 0.693 | 0.000 | 0.702 | 0.551 | 0.727 | 0.675 | 0.722 | 0.706 | 0.769 | 0.744 | 0.833 | 0.846 | **0.904** | 0.683 | 0.000 | 0.682 | 0.568 | 0.734 | 0.663 | 0.769 | 0.735 | 0.781 | 0.778 | 0.819 | 0.820 | **0.904** |
| rewo_cdi | katom_rewo | 0.642 | 0.000 | 0.646 | 0.513 | 0.667 | 0.567 | 0.672 | 0.614 | 0.689 | 0.655 | 0.716 | 0.697 | **0.81** | 0.643 | 0.000 | 0.645 | 0.537 | 0.663 | 0.533 | 0.668 | 0.577 | 0.679 | 0.618 | 0.710 | 0.680 | **0.81** |
| katom_rewo | rewo_cdi | 0.555 | 0.000 | 0.568 | 0.519 | 0.573 | 0.534 | 0.587 | 0.562 | 0.608 | 0.623 | 0.647 | 0.692 | **0.822** | 0.552 | 0.000 | 0.559 | 0.461 | 0.580 | 0.601 | 0.573 | 0.584 | 0.593 | 0.593 | 0.613 | 0.682 | **0.822** |

Figure B.10: Topic Kitchen - ATL results with no Domain Adaptation

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | random | | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.683 | 0.000 | 0.687 | 0.519 | 0.698 | 0.534 | 0.701 | 0.562 | 0.700 | 0.623 | 0.727 | 0.692 | **0.822** | 0.680 | 0.000 | 0.681 | 0.461 | 0.691 | 0.601 | 0.698 | 0.584 | 0.713 | 0.593 | 0.711 | 0.682 | **0.822** |
| rewo_cdi | katom_cdi | 0.690 | 0.000 | 0.737 | 0.551 | 0.758 | 0.675 | 0.769 | 0.706 | 0.827 | 0.744 | 0.853 | 0.846 | **0.904** | 0.680 | 0.000 | 0.691 | 0.568 | 0.750 | 0.663 | 0.750 | 0.735 | 0.806 | 0.778 | 0.844 | 0.820 | **0.904** |
| katom_cdi | katom_rewo | 0.674 | 0.000 | 0.677 | 0.513 | 0.704 | 0.567 | 0.695 | 0.614 | 0.717 | 0.655 | 0.737 | 0.697 | **0.81** | 0.680 | 0.000 | 0.687 | 0.537 | 0.677 | 0.533 | 0.685 | 0.577 | 0.698 | 0.618 | 0.712 | 0.680 | **0.81** |
| katom_rewo | katom_cdi | 0.678 | 0.000 | 0.700 | 0.551 | 0.731 | 0.675 | 0.761 | 0.706 | 0.812 | 0.744 | 0.840 | 0.846 | **0.904** | 0.704 | 0.000 | 0.699 | 0.568 | 0.756 | 0.663 | 0.762 | 0.735 | 0.805 | 0.778 | 0.848 | 0.820 | **0.904** |
| rewo_cdi | katom_rewo | 0.643 | 0.000 | 0.656 | 0.513 | 0.670 | 0.567 | 0.675 | 0.614 | 0.682 | 0.655 | 0.722 | 0.697 | **0.81** | 0.641 | 0.000 | 0.647 | 0.537 | 0.666 | 0.533 | 0.687 | 0.577 | 0.702 | 0.618 | 0.729 | 0.680 | **0.81** |
| katom_rewo | rewo_cdi | 0.560 | 0.000 | 0.566 | 0.519 | 0.578 | 0.534 | 0.606 | 0.562 | 0.620 | 0.623 | 0.667 | 0.692 | **0.822** | 0.572 | 0.000 | 0.585 | 0.461 | 0.608 | 0.601 | 0.605 | 0.584 | 0.630 | 0.593 | 0.672 | 0.682 | **0.822** |

Figure B.11: Topic Kitchen - ATL results with Nearest-neighbor Weighting

| | Estimator | rf | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QS | lr_lsvc_rf_dt | | | | | | | | | | | | | | random | | | | | | | | | | | | | | | |
| | Iteration | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | | 2nd | | 10th | | 20th | | 30th | | 50th | | 100th | | all | |
| | Results | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | ATL | AL | Tar_sup | |
| Source | Target | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| katom_cdi | rewo_cdi | 0.678 | 0.000 | 0.676 | 0.519 | 0.684 | 0.534 | 0.689 | 0.562 | 0.695 | 0.623 | 0.709 | 0.692 | **0.822** | | 0.677 | 0.000 | 0.685 | 0.461 | 0.693 | 0.601 | 0.687 | 0.584 | 0.701 | 0.593 | 0.722 | 0.682 | **0.822** | |
| rewo_cdi | katom_cdi | 0.750 | 0.000 | 0.684 | 0.551 | 0.694 | 0.675 | 0.719 | 0.706 | 0.754 | 0.744 | 0.826 | 0.846 | **0.904** | | 0.708 | 0.000 | 0.672 | 0.568 | 0.669 | 0.663 | 0.704 | 0.735 | 0.759 | 0.778 | 0.836 | 0.820 | **0.904** | |
| katom_cdi | katom_rewo | 0.664 | 0.000 | 0.676 | 0.513 | 0.679 | 0.567 | 0.686 | 0.614 | 0.700 | 0.655 | 0.716 | 0.697 | **0.81** | | 0.669 | 0.000 | 0.689 | 0.537 | 0.694 | 0.533 | 0.700 | 0.577 | 0.694 | 0.618 | 0.710 | 0.680 | **0.81** | |
| katom_rewo | katom_cdi | 0.668 | 0.000 | 0.695 | 0.551 | 0.748 | 0.675 | 0.792 | 0.706 | 0.810 | 0.744 | 0.852 | 0.846 | **0.904** | | 0.682 | 0.000 | 0.685 | 0.568 | 0.737 | 0.663 | 0.759 | 0.735 | 0.789 | 0.778 | 0.836 | 0.820 | **0.904** | |
| rewo_cdi | katom_rewo | 0.644 | 0.000 | 0.656 | 0.513 | 0.666 | 0.567 | 0.677 | 0.614 | 0.688 | 0.655 | 0.711 | 0.697 | **0.81** | | 0.638 | 0.000 | 0.644 | 0.537 | 0.654 | 0.533 | 0.663 | 0.577 | 0.683 | 0.618 | 0.711 | 0.680 | **0.81** | |
| katom_rewo | rewo_cdi | 0.539 | 0.000 | 0.551 | 0.519 | 0.569 | 0.534 | 0.579 | 0.562 | 0.612 | 0.623 | 0.656 | 0.692 | **0.822** | | 0.536 | 0.000 | 0.551 | 0.461 | 0.568 | 0.601 | 0.571 | 0.584 | 0.589 | 0.593 | 0.627 | 0.682 | **0.822** | |

Figure B.12: Topic Kitchen - ATL results with Logistic Regression Weighting

## B.4 ATLX Results

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | | | | | | | | |
| **ban_bx** | **bx_wor** | **-** | 2 | 0 | 0 | 0.410 | 0.674 | 0.752 | 0.078 | -0.009 | -0.020 | -0.066 | 0.030 | 0.011 | 0.008 |
| | | **nn** | 2 | 0 | 0 | 0.490 | 0.689 | 0.774 | 0.085 | 0.013 | 0.015 | -0.044 | 0.014 | 0.011 | 0.020 |
| | | **lp** | 2 | 0 | 0 | 0.410 | 0.661 | 0.750 | 0.089 | -0.011 | -0.009 | -0.068 | 0.034 | 0.011 | 0.047 |
| **bx_wor** | **ban_bx** | **-** | 2 | 0 | 0 | 0.450 | 0.730 | 0.837 | 0.107 | 0.039 | 0.009 | 0.023 | 0.017 | 0.034 | 0.008 |
| | | **nn** | 2 | 0 | 0 | 0.570 | 0.738 | 0.834 | 0.096 | 0.036 | 0.016 | 0.020 | 0.026 | 0.034 | 0.020 |
| | | **lp** | 2 | 0 | 0 | 0.440 | 0.734 | 0.846 | 0.112 | 0.048 | 0.040 | 0.032 | 0.007 | 0.034 | 0.017 |
| **ban_bx** | **bx_half** | **-** | 2 | 0 | 0 | 0.220 | 0.867 | 0.896 | 0.029 | 0.020 | 0.021 | 0.001 | 0.004 | 0.008 | 0.022 |
| | | **nn** | 2 | 0 | 0 | 0.270 | 0.892 | 0.896 | 0.004 | 0.020 | 0.003 | 0.001 | 0.008 | 0.008 | 0.012 |
| | | **lp** | 2 | 0 | 0 | 0.310 | 0.873 | 0.894 | 0.021 | 0.018 | -0.004 | -0.001 | 0.018 | 0.008 | 0.007 |
| **bx_half** | **ban_bx** | **-** | 2 | 0 | 1 | 0.330 | 0.728 | 0.830 | 0.102 | 0.032 | 0.033 | 0.016 | 0.009 | 0.034 | 0.025 |
| | | **nn** | 2 | 0 | 1 | 0.320 | 0.731 | 0.833 | 0.102 | 0.035 | -0.002 | 0.019 | 0.014 | 0.034 | 0.015 |
| | | **lp** | 2 | 0 | 1 | 0.320 | 0.751 | 0.813 | 0.062 | 0.015 | -0.013 | -0.001 | 0.016 | 0.034 | 0.023 |
| **bx_wor** | **bx_half** | **-** | 2 | 0 | 0 | 0.400 | 0.780 | 0.884 | 0.104 | 0.008 | -0.001 | -0.011 | 0.018 | 0.008 | 0.010 |
| | | **nn** | 2 | 0 | 0 | 0.530 | 0.793 | 0.900 | 0.107 | 0.024 | -0.004 | 0.005 | 0.007 | 0.008 | 0.008 |
| | | **lp** | 2 | 0 | 0 | 0.460 | 0.781 | 0.900 | 0.119 | 0.024 | 0.025 | 0.005 | 0.010 | 0.008 | 0.024 |
| **bx_half** | **bx_wor** | **-** | 2 | 0 | 0 | 0.340 | 0.721 | 0.747 | 0.026 | -0.014 | -0.023 | -0.071 | 0.025 | 0.011 | 0.011 |
| | | **nn** | 2 | 0 | 0 | 0.370 | 0.722 | 0.749 | 0.027 | -0.012 | 0.009 | -0.069 | 0.053 | 0.011 | 0.042 |
| | | **lp** | 2 | 0 | 0 | 0.340 | 0.725 | 0.781 | 0.056 | 0.020 | 0.034 | -0.037 | 0.010 | 0.011 | 0.027 |
| **ban_half** | **wor_half** | **-** | 2 | 0 | 0 | 0.270 | 0.842 | 0.839 | -0.003 | 0.014 | 0.038 | -0.025 | 0.009 | 0.006 | 0.030 |
| | | **nn** | 2 | 0 | 0 | 0.290 | 0.835 | 0.845 | 0.010 | 0.020 | 0.024 | -0.019 | 0.003 | 0.006 | 0.016 |
| | | **lp** | 2 | 0 | 0 | 0.270 | 0.838 | 0.845 | 0.007 | 0.020 | 0.040 | -0.019 | 0.008 | 0.006 | 0.038 |
| **wor_half** | **ban_half** | **-** | 2 | 0 | 0 | 0.320 | 0.869 | 0.955 | 0.086 | -0.001 | 0.010 | -0.014 | 0.007 | 0.007 | 0.010 |
| | | **nn** | 2 | 0 | 0 | 0.290 | 0.878 | 0.963 | 0.085 | 0.007 | 0.026 | -0.006 | 0.003 | 0.007 | 0.027 |
| | | **lp** | 2 | 0 | 0 | 0.290 | 0.883 | 0.954 | 0.071 | -0.002 | 0.005 | -0.015 | 0.003 | 0.007 | 0.007 |
| **ban_half** | **ban_wor** | **-** | 2 | 0 | 0 | 0.350 | 0.796 | 0.825 | 0.029 | 0.048 | 0.011 | -0.005 | 0.008 | 0.035 | 0.011 |
| | | **nn** | 2 | 0 | 0 | 0.290 | 0.797 | 0.835 | 0.038 | 0.058 | 0.026 | 0.005 | 0.005 | 0.035 | 0.027 |
| | | **lp** | 2 | 0 | 0 | 0.410 | 0.795 | 0.822 | 0.027 | 0.045 | -0.002 | -0.008 | 0.013 | 0.035 | 0.014 |
| **ban_wor** | **ban_half** | **-** | 2 | 0 | 0 | 0.440 | 0.881 | 0.961 | 0.080 | 0.005 | 0.014 | -0.008 | 0.003 | 0.007 | 0.015 |
| | | **nn** | 2 | 0 | 0 | 0.370 | 0.876 | 0.959 | 0.083 | 0.003 | 0.009 | -0.010 | 0.006 | 0.007 | 0.008 |
| | | **lp** | 2 | 0 | 0 | 0.410 | 0.876 | 0.961 | 0.085 | 0.005 | 0.010 | -0.008 | 0.003 | 0.007 | 0.006 |
| **wor_half** | **ban_wor** | **-** | 2 | 0 | 0 | 0.330 | 0.799 | 0.825 | 0.026 | 0.048 | 0.033 | -0.005 | 0.019 | 0.035 | 0.027 |
| | | **nn** | 2 | 0 | 0 | 0.290 | 0.806 | 0.823 | 0.017 | 0.046 | 0.013 | -0.007 | 0.008 | 0.035 | 0.011 |
| | | **lp** | 2 | 0 | 0 | 0.310 | 0.787 | 0.804 | 0.017 | 0.027 | -0.004 | -0.026 | 0.018 | 0.035 | 0.029 |
| **ban_wor** | **wor_half** | **-** | 2 | 0 | 0 | 0.340 | 0.772 | 0.839 | 0.067 | 0.014 | 0.009 | -0.025 | 0.013 | 0.006 | 0.010 |
| | | **nn** | 2 | 0 | 0 | 0.350 | 0.775 | 0.836 | 0.061 | 0.011 | 0.012 | -0.028 | 0.009 | 0.006 | 0.011 |
| | | **lp** | 2 | 0 | 0 | 0.400 | 0.759 | 0.838 | 0.079 | 0.013 | 0.010 | -0.026 | 0.005 | 0.006 | 0.007 |

Figure B.13: Topic Books - ATLX Performance Indicators. DAs compared (related combinations = $MCC < 0.3$)

| | | | ls | n_+ | n_- | cor | f1_out_0 | f1_out_-1 | Δ_out_f1 | Δ_AL_f1 | Δ_R_f1 | Δ_pas_f1 | sig_out | sig_AL | sig_R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | | | | | | | | |
| **ban_bx** | **ban_half** | **-** | 2 | 0 | 0 | 0.310 | 0.948 | 0.959 | 0.011 | 0.003 | 0.020 | -0.010 | 0.003 | 0.007 | 0.009 |
| | | **nn** | 2 | 0 | 0 | 0.320 | 0.942 | 0.961 | 0.019 | 0.005 | 0.013 | -0.008 | 0.003 | 0.007 | 0.011 |
| | | **lp** | 2 | 0 | 0 | 0.330 | 0.924 | 0.961 | 0.037 | 0.005 | 0.013 | -0.008 | 0.003 | 0.007 | 0.005 |
| **ban_half** | **ban_bx** | **-** | 2 | 0 | 0 | 0.420 | 0.670 | 0.833 | 0.163 | 0.035 | 0.002 | 0.019 | 0.015 | 0.034 | 0.017 |
| | | **nn** | 2 | 0 | 0 | 0.460 | 0.683 | 0.848 | 0.165 | 0.050 | 0.015 | 0.034 | 0.014 | 0.034 | 0.019 |
| | | **lp** | 2 | 0 | 0 | 0.460 | 0.654 | 0.828 | 0.174 | 0.030 | -0.001 | 0.014 | 0.024 | 0.034 | 0.009 |
| **ban_bx** | **ban_wor** | **-** | 2 | 0 | 0 | 0.410 | 0.797 | 0.799 | 0.002 | 0.022 | 0.001 | -0.031 | 0.030 | 0.035 | 0.034 |
| | | **nn** | 2 | 0 | 0 | 0.380 | 0.799 | 0.800 | 0.001 | 0.023 | -0.005 | -0.030 | 0.016 | 0.035 | 0.015 |
| | | **lp** | 2 | 0 | 0 | 0.360 | 0.797 | 0.802 | 0.005 | 0.025 | 0.012 | -0.028 | 0.027 | 0.035 | 0.027 |
| **ban_wor** | **ban_bx** | **-** | 2 | 0 | 1 | 0.500 | 0.462 | 0.821 | 0.359 | 0.023 | 0.000 | 0.007 | 0.011 | 0.034 | 0.017 |
| | | **nn** | 2 | 0 | 1 | 0.560 | 0.447 | 0.821 | 0.374 | 0.023 | 0.011 | 0.007 | 0.025 | 0.034 | 0.031 |
| | | **lp** | 2 | 0 | 1 | 0.510 | 0.463 | 0.832 | 0.369 | 0.034 | 0.013 | 0.018 | 0.014 | 0.034 | 0.032 |
| **bx_wor** | **wor_half** | **-** | 2 | 0 | 0 | 0.450 | 0.703 | 0.825 | 0.122 | 0.000 | 0.012 | -0.039 | 0.012 | 0.006 | 0.012 |
| | | **nn** | 2 | 0 | 0 | 0.430 | 0.707 | 0.817 | 0.110 | -0.008 | 0.023 | -0.047 | 0.024 | 0.006 | 0.037 |
| | | **lp** | 2 | 0 | 0 | 0.420 | 0.700 | 0.822 | 0.122 | -0.003 | 0.067 | -0.042 | 0.023 | 0.006 | 0.048 |
| **wor_half** | **bx_wor** | **-** | 2 | 0 | 0 | 0.470 | 0.509 | 0.769 | 0.260 | 0.008 | 0.008 | -0.049 | 0.020 | 0.011 | 0.025 |
| | | **nn** | 2 | 0 | 0 | 0.490 | 0.532 | 0.755 | 0.223 | -0.006 | -0.012 | -0.063 | 0.018 | 0.011 | 0.006 |
| | | **lp** | 2 | 0 | 0 | 0.540 | 0.489 | 0.759 | 0.270 | -0.002 | -0.001 | -0.059 | 0.021 | 0.011 | 0.014 |
| **bx_wor** | **ban_wor** | **-** | 2 | 0 | 0 | 0.450 | 0.747 | 0.797 | 0.050 | 0.020 | -0.003 | -0.033 | 0.020 | 0.035 | 0.016 |
| | | **nn** | 2 | 1 | 0 | 0.470 | 0.719 | 0.811 | 0.092 | 0.034 | 0.007 | -0.019 | 0.010 | 0.035 | 0.012 |
| | | **lp** | 2 | 0 | 0 | 0.420 | 0.732 | 0.817 | 0.085 | 0.040 | 0.022 | -0.013 | 0.012 | 0.035 | 0.045 |
| **ban_wor** | **bx_wor** | **-** | 2 | 0 | 0 | 0.450 | 0.420 | 0.749 | 0.329 | -0.012 | 0.020 | -0.069 | 0.026 | 0.011 | 0.042 |
| | | **nn** | 2 | 0 | 0 | 0.510 | 0.437 | 0.749 | 0.312 | -0.012 | 0.042 | -0.069 | 0.044 | 0.011 | 0.055 |
| | | **lp** | 2 | 0 | 0 | 0.470 | 0.379 | 0.766 | 0.387 | 0.005 | 0.014 | -0.052 | 0.019 | 0.011 | 0.016 |
| **ban_half** | **bx_half** | **-** | 2 | 0 | 0 | 0.420 | 0.750 | 0.894 | 0.144 | 0.018 | 0.030 | -0.001 | 0.009 | 0.008 | 0.039 |
| | | **nn** | 2 | 0 | 0 | 0.370 | 0.810 | 0.893 | 0.083 | 0.017 | 0.008 | -0.002 | 0.012 | 0.008 | 0.011 |
| | | **lp** | 2 | 0 | 0 | 0.450 | 0.735 | 0.895 | 0.160 | 0.019 | 0.023 | 0.000 | 0.012 | 0.008 | 0.030 |
| **bx_half** | **ban_half** | **-** | 2 | 0 | 0 | 0.440 | 0.736 | 0.963 | 0.227 | 0.007 | 0.005 | -0.006 | 0.004 | 0.007 | 0.003 |
| | | **nn** | 2 | 0 | 0 | 0.400 | 0.828 | 0.957 | 0.129 | 0.001 | 0.006 | -0.012 | 0.004 | 0.007 | 0.006 |
| | | **lp** | 2 | 0 | 0 | 0.430 | 0.725 | 0.962 | 0.237 | 0.006 | 0.006 | -0.007 | 0.002 | 0.007 | 0.004 |
| **wor_half** | **bx_half** | **-** | 2 | 0 | 0 | 0.560 | 0.580 | 0.885 | 0.305 | 0.009 | 0.023 | -0.010 | 0.011 | 0.008 | 0.024 |
| | | **nn** | 2 | 0 | 0 | 0.470 | 0.555 | 0.892 | 0.337 | 0.016 | 0.001 | -0.003 | 0.012 | 0.008 | 0.012 |
| | | **lp** | 2 | 0 | 0 | 0.410 | 0.581 | 0.889 | 0.308 | 0.013 | 0.008 | -0.006 | 0.032 | 0.008 | 0.029 |
| **bx_half** | **wor_half** | **-** | 2 | 0 | 0 | 0.430 | 0.693 | 0.811 | 0.118 | -0.014 | 0.009 | -0.053 | 0.022 | 0.006 | 0.022 |
| | | **nn** | 2 | 0 | 0 | 0.410 | 0.736 | 0.829 | 0.093 | 0.004 | 0.034 | -0.035 | 0.009 | 0.006 | 0.023 |
| | | **lp** | 2 | 0 | 0 | 0.410 | 0.706 | 0.819 | 0.113 | -0.006 | -0.001 | -0.045 | 0.023 | 0.006 | 0.012 |

Figure B.14: Topic Books - ATLX Performance Indicators. DAs compared (unrelated combinations = $MCC > 0.3$)

| | | | adt_0 | adt_-1 | c_0 | c_-1 | ordered_attr_import_0 | ordered_attr_import_-1 |
|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | |
| **ban_bx** | **bx_wor** | **-** | 24.300 | 7.670 | 0.341 | 0.225 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | **nn** | 23.700 | 7.205 | 0.344 | 0.248 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | **lp** | 23.700 | 7.981 | 0.351 | 0.226 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| **bx_wor** | **ban_bx** | **-** | 37.200 | 7.870 | 0.305 | 0.264 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 35.000 | 7.668 | 0.302 | 0.279 | ['pubdate', 'author', 'publisher', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 36.400 | 7.783 | 0.326 | 0.269 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| **ban_bx** | **bx_half** | **-** | 24.300 | 7.040 | 0.410 | 0.350 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 21.100 | 6.920 | 0.413 | 0.356 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 23.200 | 7.046 | 0.415 | 0.348 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **bx_half** | **ban_bx** | **-** | 20.400 | 6.695 | 0.375 | 0.297 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 19.900 | 6.562 | 0.372 | 0.316 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 21.700 | 7.021 | 0.377 | 0.307 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **bx_wor** | **bx_half** | **-** | 37.200 | 7.880 | 0.343 | 0.313 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 36.600 | 7.372 | 0.340 | 0.341 | ['pubdate', 'author', 'publisher', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 38.300 | 7.653 | 0.344 | 0.341 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **bx_half** | **bx_wor** | **-** | 20.400 | 7.647 | 0.375 | 0.250 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 21.500 | 7.577 | 0.372 | 0.206 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 21.600 | 7.678 | 0.375 | 0.230 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| **ban_half** | **wor_half** | **-** | 13.700 | 7.160 | 0.413 | 0.277 | ['pubdate', 'title', 'pages', 'publisher', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **nn** | 14.600 | 7.026 | 0.410 | 0.288 | ['pubdate', 'title', 'pages', 'publisher', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **lp** | 13.600 | 7.223 | 0.411 | 0.253 | ['pubdate', 'title', 'pages', 'publisher', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| **wor_half** | **ban_half** | **-** | 26.400 | 6.636 | 0.377 | 0.356 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'title', 'publisher', 'author'] |
| | | **nn** | 25.700 | 6.429 | 0.368 | 0.369 | ['pubdate', 'publisher', 'pages', 'title', 'author'] | ['pubdate', 'pages', 'title', 'publisher', 'author'] |
| | | **lp** | 25.400 | 6.349 | 0.372 | 0.362 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'title', 'publisher', 'author'] |
| **ban_half** | **ban_wor** | **-** | 13.700 | 6.630 | 0.389 | 0.276 | ['pubdate', 'title', 'pages', 'publisher', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **nn** | 15.000 | 6.842 | 0.389 | 0.263 | ['pubdate', 'title', 'pages', 'publisher', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **lp** | 14.200 | 6.815 | 0.389 | 0.259 | ['pubdate', 'title', 'pages', 'publisher', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| **ban_wor** | **ban_half** | **-** | 23.700 | 6.319 | 0.378 | 0.368 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'title', 'publisher', 'author'] |
| | | **nn** | 21.600 | 6.117 | 0.376 | 0.370 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'title', 'publisher', 'author'] |
| | | **lp** | 21.500 | 5.871 | 0.376 | 0.380 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'title', 'publisher', 'author'] |
| **wor_half** | **ban_wor** | **-** | 26.400 | 7.257 | 0.334 | 0.238 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **nn** | 26.100 | 7.403 | 0.344 | 0.253 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **lp** | 26.200 | 7.652 | 0.332 | 0.233 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| **ban_wor** | **wor_half** | **-** | 23.700 | 7.577 | 0.363 | 0.264 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **nn** | 22.600 | 7.652 | 0.359 | 0.278 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |
| | | **lp** | 23.000 | 7.374 | 0.355 | 0.282 | ['pubdate', 'pages', 'publisher', 'title', 'author'] | ['pubdate', 'pages', 'publisher', 'title', 'author'] |

Figure B.15: Topic Books - ATLX Further Analysis. DAs compared (related combinations = $MCC < 0.3$)

| | | | adt_0 | adt_-1 | c_0 | c_-1 | ordered_attr_import_0 | ordered_attr_import_-1 |
|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | |
| **ban_bx** | **ban_half** | **-** | 24.300 | 6.384 | 0.434 | 0.389 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'title', 'publisher', 'author'] |
| | | **nn** | 21.800 | 6.393 | 0.380 | 0.393 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 21.900 | 6.347 | 0.418 | 0.400 | ['pubdate', 'title', 'publisher', 'author'] | ['pubdate', 'title', 'publisher', 'author'] |
| **ban_half** | **ban_bx** | **-** | 15.700 | 6.606 | 0.339 | 0.261 | ['pubdate', 'title', 'author', 'publisher'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 14.300 | 6.828 | 0.364 | 0.285 | ['pubdate', 'title', 'publisher', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 15.000 | 6.833 | 0.342 | 0.293 | ['pubdate', 'title', 'author', 'publisher'] | ['pubdate', 'publisher', 'title', 'author'] |
| **ban_bx** | **ban_wor** | **-** | 24.300 | 7.882 | 0.392 | 0.242 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 22.800 | 7.480 | 0.392 | 0.274 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 20.200 | 7.636 | 0.370 | 0.256 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **ban_wor** | **ban_bx** | **-** | 24.600 | 7.044 | 0.312 | 0.286 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 22.500 | 6.689 | 0.311 | 0.293 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 23.300 | 7.134 | 0.309 | 0.288 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **bx_wor** | **wor_half** | **-** | 37.200 | 8.865 | 0.329 | 0.266 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 38.700 | 8.873 | 0.318 | 0.272 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 37.000 | 8.606 | 0.338 | 0.292 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| **wor_half** | **bx_wor** | **-** | 28.500 | 7.977 | 0.275 | 0.220 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | **nn** | 26.900 | 7.531 | 0.272 | 0.236 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | **lp** | 26.300 | 7.756 | 0.266 | 0.225 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| **bx_wor** | **ban_wor** | **-** | 37.200 | 7.986 | 0.325 | 0.247 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 37.000 | 7.792 | 0.313 | 0.304 | ['pubdate', 'author', 'publisher', 'title'] | ['pubdate', 'publisher', 'author', 'title'] |
| | | **lp** | 36.300 | 8.078 | 0.320 | 0.271 | ['pubdate', 'publisher', 'author', 'title'] | ['pubdate', 'publisher', 'title', 'author'] |
| **ban_wor** | **bx_wor** | **-** | 24.600 | 7.309 | 0.284 | 0.247 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 23.200 | 7.275 | 0.298 | 0.211 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 24.400 | 7.832 | 0.291 | 0.212 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'author', 'title'] |
| **ban_half** | **bx_half** | **-** | 15.700 | 6.735 | 0.349 | 0.353 | ['pubdate', 'title', 'author', 'publisher'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 14.500 | 6.582 | 0.373 | 0.330 | ['pubdate', 'title', 'publisher', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 14.300 | 6.479 | 0.349 | 0.349 | ['pubdate', 'title', 'publisher', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **bx_half** | **ban_half** | **-** | 20.400 | 5.902 | 0.355 | 0.393 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'title', 'publisher', 'author'] |
| | | **nn** | 20.400 | 6.088 | 0.369 | 0.376 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'title', 'publisher', 'author'] |
| | | **lp** | 21.300 | 5.951 | 0.349 | 0.392 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'title', 'publisher', 'author'] |
| **wor_half** | **bx_half** | **-** | 28.500 | 7.263 | 0.330 | 0.331 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 28.800 | 7.394 | 0.309 | 0.345 | ['pubdate', 'title', 'publisher', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 26.300 | 7.110 | 0.331 | 0.325 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| **bx_half** | **wor_half** | **-** | 20.400 | 8.177 | 0.359 | 0.271 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **nn** | 21.200 | 8.278 | 0.360 | 0.288 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |
| | | **lp** | 20.200 | 8.256 | 0.366 | 0.260 | ['pubdate', 'publisher', 'title', 'author'] | ['pubdate', 'publisher', 'title', 'author'] |

Figure B.16: Topic Books - ATLX Further Analysis. DAs compared (unrelated combinations = $MCC > 0.3$)

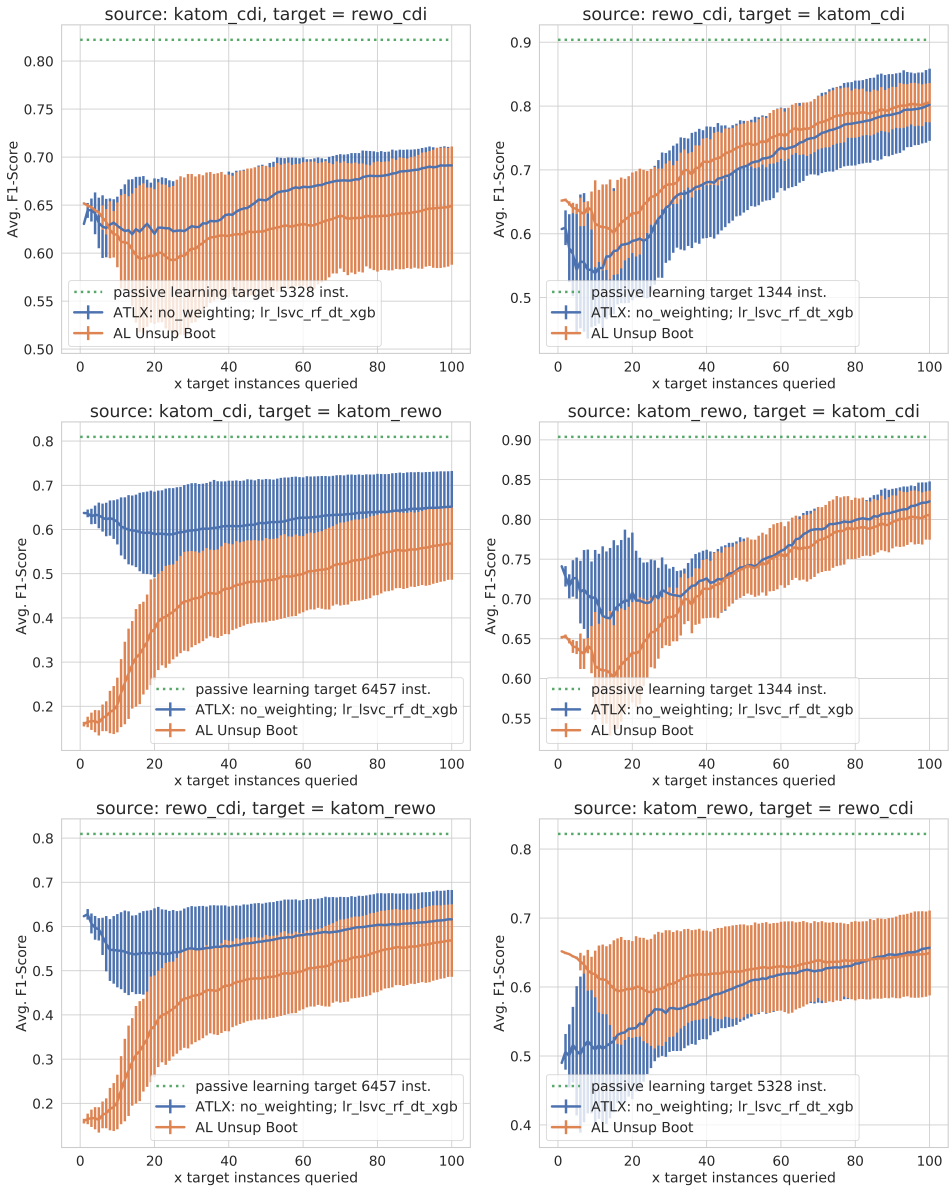


Figure B.17: Topic Kitchen - ATLX without DA and Baseline 2 (errorbars)

| | | | ls | adt_0 | adt_-1 | c_0 | c_-1 | ordered_attr_import_0 | ordered_attr_import_-1 |
|---|---|---|---|---|---|---|---|---|---|
| Source | Target | DA | | | | | | | |
| katom_cdi | rewo_cdi | - | 2 | 19.200 | 7.150 | 0.229 | 0.141 | ['base', 'product', 'style', 'height', 'title', 'material', 'color', 'capacity', 'finish', 'brand'] | ['title', 'style', 'base', 'product', 'material', 'finish', 'brand', 'color', 'capacity', 'height'] |
| | | nn | 2 | 18.400 | 7.055 | 0.238 | 0.142 | ['product', 'height', 'style', 'base', 'title', 'material', 'capacity', 'finish', 'color', 'brand'] | ['title', 'product', 'style', 'material', 'base', 'finish', 'color', 'brand', 'height', 'capacity'] |
| | | lp | 2 | 19.100 | 7.081 | 0.297 | 0.140 | ['title', 'style', 'height', 'base', 'capacity', 'product', 'material', 'color', 'brand', 'finish'] | ['title', 'product', 'style', 'base', 'finish', 'material', 'capacity', 'brand', 'height', 'color'] |
| rewo_cdi | katom_cdi | - | 2 | 29.500 | 7.042 | 0.174 | 0.166 | ['title', 'style', 'product', 'base', 'material', 'brand', 'finish', 'capacity', 'height', 'color'] | ['title', 'style', 'product', 'base', 'height', 'material', 'capacity', 'color', 'finish', 'brand'] |
| | | nn | 2 | 28.500 | 7.009 | 0.157 | 0.167 | ['title', 'product', 'style', 'base', 'capacity', 'material', 'brand', 'finish', 'height', 'color'] | ['title', 'style', 'product', 'base', 'height', 'material', 'capacity', 'finish', 'color', 'brand'] |
| | | lp | 2 | 30.900 | 7.361 | 0.180 | 0.168 | ['title', 'product', 'brand', 'height', 'base', 'style', 'capacity', 'material', 'finish', 'color'] | ['title', 'product', 'style', 'base', 'height', 'material', 'capacity', 'finish', 'color', 'brand'] |
| katom_cdi | katom_rewo | - | 2 | 19.200 | 6.618 | 0.222 | 0.123 | ['base', 'product', 'style', 'height', 'title', 'material', 'color', 'capacity', 'finish', 'brand'] | ['title', 'style', 'product', 'base', 'capacity', 'material', 'color', 'height', 'finish', 'brand'] |
| | | nn | 2 | 17.700 | 6.305 | 0.235 | 0.137 | ['height', 'style', 'base', 'product', 'title', 'material', 'color', 'capacity', 'finish', 'brand'] | ['title', 'style', 'color', 'product', 'height', 'capacity', 'material', 'base', 'finish', 'brand'] |
| | | lp | 2 | 19.000 | 6.454 | 0.223 | 0.119 | ['title', 'style', 'height', 'base', 'product', 'finish', 'capacity', 'color', 'material', 'brand'] | ['title', 'style', 'product', 'color', 'height', 'base', 'capacity', 'material', 'finish', 'brand'] |
| katom_rewo | katom_cdi | - | 2 | 26.500 | 6.667 | 0.224 | 0.179 | ['title', 'color', 'style', 'product', 'capacity', 'material', 'base', 'height', 'finish', 'brand'] | ['style', 'title', 'base', 'product', 'height', 'material', 'capacity', 'color', 'finish', 'brand'] |
| | | nn | 2 | 24.400 | 6.895 | 0.165 | 0.181 | ['title', 'style', 'base', 'color', 'capacity', 'height', 'product', 'material', 'finish', 'brand'] | ['title', 'style', 'base', 'product', 'height', 'material', 'capacity', 'color', 'finish', 'brand'] |
| | | lp | 2 | 24.500 | 6.404 | 0.207 | 0.190 | ['title', 'style', 'color', 'product', 'finish', 'capacity', 'height', 'material', 'base', 'brand'] | ['title', 'style', 'product', 'base', 'height', 'material', 'color', 'capacity', 'finish', 'brand'] |
| rewo_cdi | katom_rewo | - | 2 | 29.500 | 6.669 | 0.216 | 0.134 | ['title', 'style', 'product', 'base', 'material', 'brand', 'finish', 'capacity', 'height', 'color'] | ['title', 'style', 'product', 'base', 'capacity', 'brand', 'color', 'material', 'finish', 'height'] |
| | | nn | 2 | 29.800 | 6.932 | 0.207 | 0.105 | ['title', 'brand', 'base', 'product', 'style', 'material', 'height', 'finish', 'color', 'capacity'] | ['title', 'style', 'capacity', 'color', 'product', 'base', 'material', 'height', 'finish', 'brand'] |
| | | lp | 2 | 31.100 | 7.012 | 0.231 | 0.111 | ['title', 'base', 'product', 'capacity', 'material', 'style', 'height', 'finish', 'color', 'brand'] | ['title', 'style', 'product', 'color', 'capacity', 'base', 'finish', 'material', 'height', 'brand'] |
| katom_rewo | rewo_cdi | - | 2 | 26.500 | 7.252 | 0.186 | 0.130 | ['title', 'color', 'style', 'product', 'capacity', 'material', 'base', 'height', 'finish', 'brand'] | ['title', 'base', 'style', 'product', 'finish', 'material', 'brand', 'color', 'capacity', 'height'] |
| | | nn | 2 | 25.200 | 7.178 | 0.195 | 0.114 | ['title', 'style', 'color', 'product', 'base', 'capacity', 'material', 'finish', 'brand', 'height'] | ['title', 'style', 'product', 'capacity', 'material', 'finish', 'base', 'color', 'brand', 'height'] |
| | | lp | 2 | 23.500 | 6.940 | 0.211 | 0.111 | ['title', 'product', 'style', 'color', 'capacity', 'material', 'finish', 'base', 'height', 'brand'] | ['title', 'style', 'product', 'material', 'finish', 'base', 'brand', 'color', 'capacity', 'height'] |

Figure B.18: Topic Kitchen - ATLX Further Analysis. DAs compared

| | | | ls | adt_0 | adt_-1 | c_0 | c_-1 | ordered_attr_import_0 | ordered_attr_import_-1 |
|---|---|---|---|---|---|---|---|---|---|
| **Source** | **Target** | **DA** | | | | | | | |
| **dbp_dnb** | **dbp_wiki** | **-** | 2 | 19.500 | 5.973 | 0.480 | 0.440 | ['birthdate', 'name', 'deathdate', 'gender'] | ['name', 'birthdate', 'deathdate', 'gender'] |
| | | **nn** | 2 | 21.200 | 6.562 | 0.477 | 0.438 | ['birthdate', 'name', 'deathdate', 'gender'] | ['name', 'birthdate', 'deathdate', 'gender'] |
| | | **lp** | 2 | 19.700 | 6.216 | 0.478 | 0.440 | ['birthdate', 'name', 'deathdate', 'gender'] | ['name', 'birthdate', 'deathdate', 'gender'] |
| **dbp_wiki** | **dbp_dnb** | **-** | 2 | 20.000 | 5.265 | 0.479 | 0.427 | ['name', 'birthdate', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **nn** | 2 | 19.500 | 5.170 | 0.478 | 0.425 | ['name', 'birthdate', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **lp** | 2 | 19.500 | 5.432 | 0.483 | 0.422 | ['birthdate', 'name', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| **dbp_dnb** | **dbp_viaf** | **-** | 2 | 19.500 | 5.860 | 0.492 | 0.438 | ['birthdate', 'name', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **nn** | 2 | 21.000 | 5.746 | 0.492 | 0.430 | ['birthdate', 'name', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **lp** | 2 | 20.700 | 5.869 | 0.492 | 0.447 | ['birthdate', 'deathdate', 'name', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| **dbp_viaf** | **dbp_dnb** | **-** | 2 | 17.200 | 5.577 | 0.480 | 0.423 | ['birthdate', 'name', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **nn** | 2 | 16.900 | 5.485 | 0.471 | 0.420 | ['birthdate', 'name', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **lp** | 2 | 17.700 | 5.361 | 0.476 | 0.426 | ['birthdate', 'name', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| **dbp_wiki** | **dbp_viaf** | **-** | 2 | 20.000 | 5.666 | 0.496 | 0.441 | ['name', 'birthdate', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **nn** | 2 | 20.000 | 5.706 | 0.496 | 0.441 | ['name', 'birthdate', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| | | **lp** | 2 | 20.200 | 5.820 | 0.496 | 0.440 | ['name', 'birthdate', 'deathdate', 'gender'] | ['birthdate', 'name', 'deathdate', 'gender'] |
| **dbp_viaf** | **dbp_wiki** | **-** | 2 | 17.200 | 6.164 | 0.494 | 0.438 | ['birthdate', 'name', 'deathdate', 'gender'] | ['name', 'birthdate', 'deathdate', 'gender'] |
| | | **nn** | 2 | 18.300 | 5.915 | 0.493 | 0.446 | ['birthdate', 'name', 'deathdate', 'gender'] | ['name', 'birthdate', 'deathdate', 'gender'] |
| | | **lp** | 2 | 17.400 | 6.138 | 0.494 | 0.437 | ['birthdate', 'name', 'deathdate', 'gender'] | ['name', 'birthdate', 'deathdate', 'gender'] |

Figure B.19: Topic Authors - ATLX Further Analysis. DAs compared

## Ehrenwörtliche Erklärung

Hiermit versichere ich, dass diese Arbeit von mir persönlich verfasst wurde und dass ich keinerlei fremde Hilfe in Anspruch genommen habe. Ebenso versichere ich, dass diese Arbeit oder Teile daraus weder von mir selbst noch von anderen als Leistungsnachweise andernorts eingereicht wurden. Wörtliche oder sinngemäße Übernahmen aus anderen Schriften und Veröffentlichungen in gedruckter oder elektronischer Form sind gekennzeichnet. Sämtliche Sekundärliteratur und sonstige Quellen sind nachgewiesen und in der Bibliographie aufgeführt. Das Gleiche gilt für graphische Darstellungen und Bilder sowie für alle Internet-Quellen. Ich bin ferner damit einverstanden, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs in elektronischer Form anonymisiert versendet und gespeichert werden kann.

Heidelberg, den 31.07.2020 Unterschrift