ReFrESH – Relation-preserving Feedback-reliant Enhancement of Subjective Content Descriptions

Magnus Bender*, Tanya Braun†, Ralf Möller* and Marcel Gehrke*

*University of Lübeck
Institute of Information Systems
Ratzeburger Allee 160, 23562 Lübeck
{bender, moeller, gehrke}@ifis.uni-luebeck.de

†University of Münster
Computer Science Department
Einsteinstr. 62, 48155 Münster
tanya.braun@uni-muenster.de

***Abstract*—An agent providing an information retrieval service may work with a corpus of text documents. The documents in the corpus may contain annotations such as Subjective Content Descriptions (SCD)—additional data associated with different sentences of the documents. Each SCD is associated with multiple sentences of the corpus and has relations among each other. The agent uses the SCDs to create its answers in response to user supplied queries. However, a user of the agent may not be the creator of the SCDs for the corpus. Hence, answers may be considered faulty by an agent's user, because the SCDs may not exactly match the perceptions of an agent's user. A naive and very costly approach would be to ask each user to completely create all the SCD themselves. To circumvent this, this paper presents ReFrESH, an approach for Relation-preserving Feedback-reliant Enhancement of SCDs by Humans. An agent's user can give feedback about faulty answers to the agent. This feedback is then used by ReFrESH to update the SCDs incrementally. Using ReFrESH, SCDs can be refreshed with feedback by humans and it allows users to build even better SCDs for their needs.**

I. Introduction

A corpus of text documents may contain Subjective Content Descriptions (SCDs) [1], which are additional location-specific data associated with sentences of the text documents. SCDs highlight points of interest nearby their location, here the sentence of a document, by providing descriptions, references, or explanations. A human or an automated annotation technique, e.g., OpenIE [2] or USEM [3], may create SCDs for a specific corpus. In general, when reading a text document, each human gets its own perceptions and views of the text document. For example, think about studying for an exam. You take the script and start to annotate things you consider crucial with your understanding. Consequently, the SCDs added to a corpus by a human are slightly different and *subjective* depending on the particular human. Similarly, SCDs estimated with automated annotation techniques depend on the particular technique.

We assume we have a corpus associated with SCDs to start with. Together, the text documents and the associated SCDs build a model of the corpus. An agent [4], which is a rational and autonomous unit acting in a world fulfilling a defined task, uses this model and provides an Information Retrieval (IR) service. The IR agent answers queries of users which may be humans or other agents. A query is some unseen text to which the user is interested in finding similar and relevant documents from the agent's corpus. To answer a query, the IR agent uses the SCDs associated with the corpus and returns documents that are assumed to be similar because they share the same SCDs as the query. A user may respond to the IR agent's answer with feedback, i.e., may report a faulty or not similar document.

The IR agent heavily relies on SCDs. However, in most cases, a user of the agent will not create the initial SCDs of the corpus itself. At this point, there may be a difference between the SCDs used by the agent and the SCDs envisioned by the user. In other words, the SCDs used for IR do not necessarily represent the perceptions of the agent's user. One possibility to avoid the difference is to force each user to create the initial SCDs of the corpus itself. However, having each user annotate the whole corpus with its understanding is not practical. Therefore, we update the model in case the user determines a mismatch. Then, the IR agents can change the SCDs according to the feedback about the mismatch. To do so, it needs a technique to incrementally change SCDs.

A technique to incrementally change SCD-based models is the Feedback-reliant Enhancement of SCDs by Humans (FrESH) [5]. FrESH removes faulty sentences and their SCDs entirely from a corpus. However, removing the entire sentence does not solve the problem this paper addresses: A faulty association between an SCD and a sentence needs to be updated based on feedback from a user, but the sentence needs to remain in the corpus. As solution this paper presents ReFrESH, an approach for Relation-preserving Feedback-reliant Enhancement of SCDs by Humans.

ReFrESH assigns the sentence with the faulty association to a different and hopefully better fitting SCD. The other sentences are also consider and may be assigned to a different, a new, or the same SCD. Considering all sentences is necessary, as the creation of SCDs consists of multiple steps whereas each step influences the next steps. However, influences between steps can not be reproduced retrospectively and thus can not easily be considered by ReFrESH. The term relation-preserving emphasizes that relations among SCDs and other sentences associated with SCDs are considered by ReFrESH. A relation, e.g., *homonym*, between two SCDs, one about a river bank and one about a financial institution, is preserved.

The remainder of this paper is structured as follows: First, we look at related work. Second, we recap the basics of SCDs, the estimation of SCDs and sketch FrESH. Afterwards, we formalize the problem of updating a single SCD while preserving relations to other SCDs and sentences and present the solution ReFrESH. Finally, we present an evaluation of ReFrESH and conclude afterwards.

II. Related Work

Before we introduce the preliminaries of SCDs and present ReFrESH, we take a look at related work. Incrementally updating or changing already available models has been investigated in different ways, but not with SCDs.

One well-known approach is to pre-train a more general model first and fine-tune it later for a specific task. During fine-tuning the model is trained on new task specific data. A typical example are transformer models like BERT [6] or GPT [7].

Another possibility is to bring the task specific data in during the computation of the answer. In this case, the model is not updated, but to each query some user and case specific data is added before the query is processed by the model. Most chat bots like ChatGPT[1] or Bard[2] use this technique.

Both of these techniques incrementally update a model, like ReFrESH, and work with models based on deep learning. Techniques that update a model must be distinguished from techniques that completely remove an item from the model, such as FrESH. There are approaches to remove an item from a model, e.g., for k-Means [8], [9] or linear and logistic regression [10]. The common idea is to avoid retraining the model and instead only incrementally change the model by applying an inverse operation that removes a single item of the training data from the model.

In this work, we focus on incrementally updating faulty associations between SCDs and sentences, while leaving the corpus unchanged and preserving all sentences.

III. Preliminaries

This section specifies notations, recaps the basics of SCDs, and the estimation of SCDs. Additionally, we sketch FrESH.

A. Notations

First, we formalize our setting of a corpus.

- A word w_i is a basic unit of discrete data from a vocabulary $\mathcal{V} = \{w_1, \dots, w_L\}$, $L \in \mathbb{N}$.
- A sentence s is defined as a sequence of words $s = (w_1, \dots, w_N)$, $N \in \mathbb{N}$, where each word $w_i \in s$ is an element of vocabulary $\mathcal{V}$. Commonly, a sentence is terminated by punctuation symbols like ".", "!", or "?".
- A document d is defined as a sequence of sentences $d = (s_1^d, ..., s_M^d)$, $M \in \mathbb{N}$.
- A corpus $\mathcal{D}$ is a set of documents $\{d_1, \dots, d_D\}$, $D \in \mathbb{N}$.
- An SCD t is a tuple. The tuple contains the SCD's additional data $\mathcal{C}$, i.e., the label l of the SCD and a set R of relations to other SCDs. Additionally, each SCD's tuple contains references to the referenced sentences in documents of $\mathcal{D}$, while in the opposite direction a sentence is associated with an SCD.
- A sentence associated with an SCD is called SCD window, inspired by a tumbling window moving over the words of a document. Generally, an SCD window might not be equal to a sentence and may be a subsequence of a sentence or the concatenated subsequences of two sentences, too. Even though, in this paper, an SCD window always equals a sentence.
- For a corpus $\mathcal{D}$ there exists a set g called SCD set containing K associated SCDs $g(\mathcal{D}) = \left\{t_j = \left(\mathcal{C}_j, \bigcup_{d \in \mathcal{D}} \{s_1^d, ..., s_S^d\}\right)\right\}_{j=1}^{K}$. Given a document $d \in \mathcal{D}$, the term $g(d)$ refers to the set of SCDs associated with sentences from document d.
- Each word $w_i \in s^d$ is associated with an influence value $I(w_i, s^d)$ representing the relevance of w_i in the sentence s^d. For example, the closer w_i is positioned to the object of the sentence s^d, the higher its corresponding influence value $I(w_i, s^d)$. The influence value is chosen according to the task, e.g., distributed binomial, linear, or constant.

B. Subjective Content Descriptions

SCDs provide additional location-specific data for documents [1]. The data provided by SCDs may be of various types, like additional definitions or links to knowledge graphs.

Kuhr et al. use an SCD-word distribution represented by a matrix when working with SCDs [1]. The SCD-word distribution matrix, in short SCD matrix, can be interpreted as a generative model. A generative model for SCDs is characterized by the assumption that the SCDs generate the words of the documents. We assume that each SCD shows a specific distribution of words of the referenced sentences in the documents.

The SCD matrix $\delta(\mathcal{D})$ models the distributions of words for all SCDs $g(\mathcal{D})$ of a corpus $\mathcal{D}$ and is structured as follows:

$$\delta(\mathcal{D}) = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_K \end{array} \begin{array}{c} \begin{array}{ccccc} w_1 & w_2 & w_3 & \cdots & w_L \end{array} \\ \begin{pmatrix} v_{1,1} & v_{1,2} & v_{1,3} & \cdots & v_{1,L} \\ v_{2,1} & v_{2,2} & v_{2,3} & \cdots & v_{2,L} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{K,1} & v_{K,2} & v_{K,3} & \cdots & v_{K,L} \end{pmatrix} \end{array}$$

The SCD matrix consists of K rows, one for each SCD in $g(\mathcal{D})$. Each row contains the word probability distribution for an SCD. Therefore, the SCD matrix has L columns, one for each word in the vocabulary of the corpus $\mathcal{D}$.

C. Supervised and UnSupervised Estimator for SCD Matrices

The SCD matrix can be estimated in a supervised manner given the set $g(\mathcal{D})$ for a corpus $\mathcal{D}$. SEM is described in Algorithm 1. Given a corpus $\mathcal{D}$, the algorithm iterates over each document d in the corpus and the document's SCDs. For each associated SCD t, the referenced sentences $\{s_1^d, ..., s_S^d\}$ are used to update the SCD matrix. Thereby, the row of the matrix representing SCD t gets incremented for each word in each sentence by each word's influence value.

[1]https://chat.openai.com/
[2]https://bard.google.com/

Algorithm 1 Supervised Estimator of SCD Matrices $\delta(\mathcal{D})$

```
1: function SEM(D, g(D))
2:     Input: Corpus D; Set of SCDs g(D)
3:     Output: SCD-word distribution matrix δ(D)
4:     Initialize an K × L matrix δ(D) with zeros
5:     for each document d ∈ D do
6:         for each SCD t = (C, {s_1^d, ..., s_S^d}) ∈ g(d) do
7:             for j = 1, ..., S do        ▷ Iterate over sentences
8:                 for each word w_i ∈ s_j^d do
9:                     δ(D)[t][w_i] += I(w_i, s_j^d)
10:    return δ(D)
```

Finally, the SCD matrix needs to be normalized row-wise to meet the requirements of a probability distribution. However, the normalization is often skipped because later the cosine similarity is often used with the rows of the matrix and the cosine similarity does a normalization by definition.

Unlike SEM, USEM estimates an SCD matrix $\delta(\mathcal{D})$ without needing the SCD set $g(\mathcal{D})$. USEM initially starts by associating each sentence to one unique SCD, which leads to an initial SCD matrix consisting of a row for each sentence in the document's corpus. Then, USEM finds the sentences in the corpus that represent the same concept and groups them one by one into an SCD.

SEM, e.g., along with OpenIE for getting $g(\mathcal{D})$, and USEM are two techniques to get SCDs for a corpus. Especially USEM can be used by an IR agent to automatically create initial SCDs for a corpus. Afterwards, the agent may use ReFrESH and update the SCDs based in the users feedback.

D. Removing Sentences from Corpus and SCDs

A may corpus contain sentences with erroneous content or sentences which are protected by privacy regulations or copyright. Such sentences need to be removed entirely from the corpus and the associated SCDs. To do so, FrESH [5] provides a technique for incrementally updating the SCD matrix. FrESH does not allow to change SCDs and instead it entirely removes faulty sentences which are associated with an SCD. FrESH inverts the operations of SEM, mainly the addition in Line 9 of Algorithm 1.

Internally, ReFrESH will need to remove and add sentences of SCDs, too. Thus, **Re**FrESH is the **Re**lation-preserving update technique for SCD-based models, which builds on top of the much smaller FrESH. ReFrESHs uses ideas of FrESH, SEM, and USEM. Next, we present ReFrESH which provides incremental updates for SCDs while it preserves relations among SCDs and leaves the corpus unchanged.

IV. Relation-preserving Updates on SCD Matrices

This section introduces ReFrESH, the algorithm that allows to update an SCD-based model by correcting faulty associations between sentences and SCDs. First, we look at possible relations among SCDs and describe ReFrESH afterwards.

A. Relations to Preserve

Before we can compose a relation-preserving algorithm, we need to specific the different relations among SCDs. An SCD t_i consists of the referenced sentences $\{s_1, ..., s_S\}$, the word distribution $(v_{i,1}, ..., v_{i,L})$, and its additional data $\mathcal{C}_i$. One of the referenced sentences s_r has been marked as faulty and should be removed. However, the remaining $S-1$ sentences $\mathcal{S}_c = \{s_1, ..., s_S\} \setminus s_r$ are then considered as correct. The word distribution will be different after a sentence is removed from an SCD, but can be easily recalculated afterwards. All items of the additional data $\mathcal{C}_i$ are related, i.e., each item can be understood as related to its SCD and thus as a relation of the SCD to be preserved.

Summarized, ReFrESH needs to preserve the relations of the referenced sentences s_r and $\mathcal{S}_c$ to the items in $\mathcal{C}_i$. In the following, $\mathcal{C}_i$ contains a label l_i (computed by LESS [11]) and a set of relations to other SCDs R_i. Each Tuple in R_i models a relation of SCDs, e.g., (t_i, t_j) a relation between t_i and t_j.

In general, a sentence may have relations which directly belong to the sentence itself and not to its SCD. As ReFrESH only modifies SCDs, relations belonging to a sentence are not effected by ReFrESH. ReFrESH uses this by shifting relations from the SCD to the sentences.

On the left hand side of Figure 1, an SCD with the previously described parts is depicted. In this example, the SCD references three sentences. The sentences with the faulty association is already marked by a red cross. The additional data contains a label and two relations of t_i to other SCDs, i.e., t_j and t_l.

B. Four Steps for Updating an SCD

ReFrESH needs to preserve the relations between the referenced sentences and the items in the additional data of an SCD. Afterwards, the sentence s_r and the sentences $\mathcal{S}_c$ shall be associated with different SCDs.

We assume to have a corpus $\mathcal{D}$ with an SCD matrix $\delta(\mathcal{D})$ and the set of SCDs $g(\mathcal{D})$. Together, these three parts build an SCD-based model which is updated by ReFrESH. To do so, ReFrESH needs four steps:

1) Shift Relations to Sentences: The input of ReFrESH is a sentence s_r which is falsely associated with an SCD t_i. ReFrESH's task is to remove s_r from t_i and to reassign s_r to a better fitting SCD. Besides preserving the relations, ReFrESH also needs to consider the impact of s_r on the SCD t_i. When s_r is associated with t_i, the word-distribution of t_i also represents s_r. Thus, similar sentences to s_r might also be added to t_i, just because s_r is associated with t_i. This is based on the assumption that sentences are added to an SCD one after another and using the word-distribution to measure similarity. For example, USEM chooses best matching sentences in a greedy way one after one. Also, humans may build their own set of SCDs manually and incrementally (including the use of ReFrESH multiple times in a row). Thus, when s_r is removed, other sentences added because of s_r may also need to be removed from the SCD.

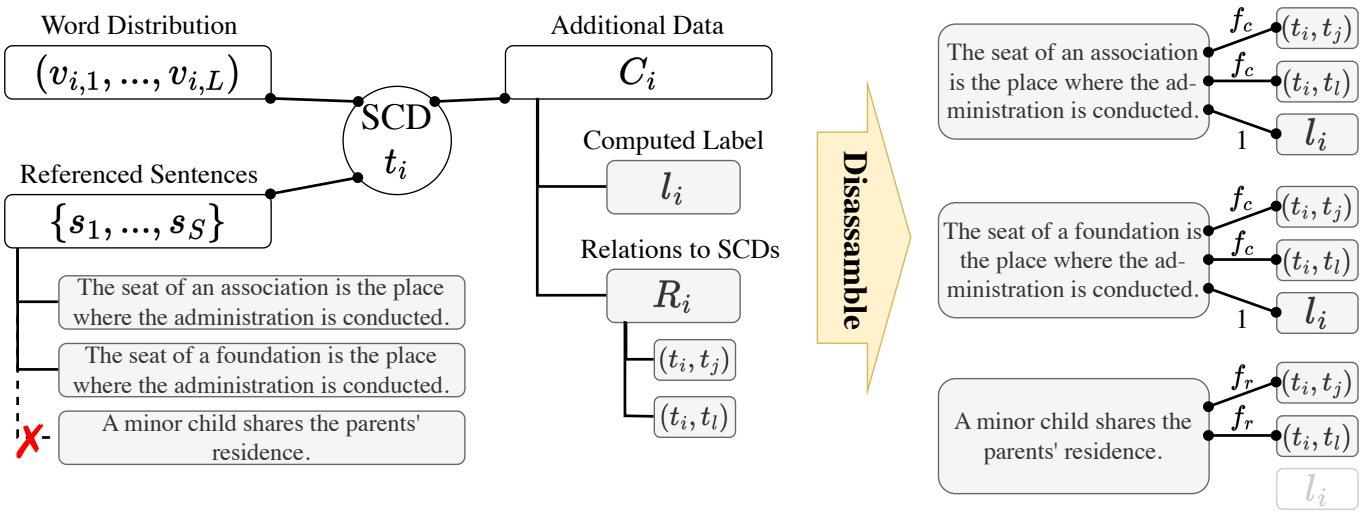

Figure 1. Left: An SCD with its referenced sentences, three in this example, its row of the word distribution, and its additional data, e.g., containing a label and two relations to other SCDs. The red cross marks the sentence to remove from the SCD. Right: The SCD after the disassemble step. Each of the three sentences now stand by itself, while the label and the two relations have been reassigned to the individual sentences and given a factor.

A comparable scenario arises when clustering data points using a density based clustering algorithm: If a point at a border of a cluster is sufficiently close to another cluster, both clusters may get merged. In this case, the point at the border becomes some type of bridge between both clusters and without this point both clusters would not have been merged. Hence, if this border point is removed later, both clusters should be separate, too.

To take this into account, ReFrESH needs to consider all referenced sentences $\{s_1, ..., s_S\}$ of t_i. To be able to consider every sentences individually, ReFrESH first shifts all the relations to preserve, the ones in $\mathcal{C}_i$, directly to the individual sentence. A sentence to remove might not share the topic of an SCD, thus, ReFrESH does not shift the label to s_r. However, the other sentences $\mathcal{S}_c$, which are considered correct, will share the topic of the SCD. Thus, the sentences in $\mathcal{S}_c$ are assigned with the label l_i and the factor 1 for this relation between sentence and label.

Next, ReFrESH does the shift for all the relations in R_i and again differentiates between sentences s_r and $\mathcal{S}_c$. The relations in R_i are added to $\mathcal{S}_c$ with the factor f_c for correct sentences. Thereby, f_c is defined as $f_c = \frac{S-1}{S}$ where S is the number of referenced sentences of t_i. In contrast, the relations in R_i are added to s_r with the factor f_r for removed sentences. Here, the factor is $f_r = \frac{1}{S}$. If some relation already has a factor, both factors are multiplied because the factors express uncertainty.

The relations from $\mathcal{C}_i$ are shifted to the sentences to make sure all relations are preserved. The factors make sure that relations are preserved differently for $\mathcal{S}_c$ and s_r, i.e., if a sentence is considered correct, the relations of the SCD are also more likely to be correct than if the sentence is falsely associated. Of course, it might be necessary to change the factor for specific relations and use-cases. All sentences and relations are stored in $\mathcal{P}$.

2) Disassemble SCD: Coming back to the problem that we do not know why a sentence was added to an SCD. ReFrESH can not determine which sentences haven been added because of s_r, too. The only solution to this problem is to disassemble the entire SCD t_i. After step 1), all relations to preserve are directly tied to each referenced sentence. Thus, the SCD t_i is not needed any more and can be disassembled without loosing important information. Afterwards, ReFrESH is able to consider each sentence separately.

Disassembling an SCD means deleting the word distribution $(v_{i,1}, ..., v_{i,L})$, the i-th row, from the SCD matrix and removing t_i from $g(\mathcal{D})$. Additionally, the SCDs in relation with t_i are informed that they are now in relation with the referenced sentences of t_i.

On the right hand side of Figure 1, an example of an disassembled SCD t_i is shown. The lowest sentences is s_r, in this case the factors are $f_r = \frac{1}{3}$ and $f_c = \frac{2}{3}$. The label gets a factor of 1 for the upper two sentences $\mathcal{S}_c$ and s_r has no label. The SCD t_i is temporarily removed completely.

3) Reassign Sentences to SCDs: Now, the previously referenced sentences stored in $\mathcal{P}$ need to be reassigned, i.e., each sentence needs to be associated with a new SCD. In this third step, ReFrESH needs to find the best SCD for each sentence. This best SCD may be an already known SCD of the corpus or newly composed SCDs while ReFrESH needs to assure that s_r does not get associated with a new SCD. A new SCD references only sentences from $\mathcal{S}_c$. If s_r gets associated with such new SCD, the association of s_r and the new SCD might be very similar or even equal to the initial SCD t_i before running ReFrESH to remove s_r.

Analogously, in the example about the clustering of data points, all points of both clusters and the border point would be considered again. Each point may be added to another cluster or one or more new clusters may be created, while the border

point will become an outlier or member of another cluster.

We still need an algorithm to reassign sentences to SCDs, which is similar to estimating SCDs in an unsupervised way. Thus, ReFrESH applies the idea of USEM and uses USEM's greedy method to reassign the sentences with new or known SCDs. The idea of USEM is to start by considering each sentences as an SCD with one referenced sentence. Afterwards, USEM uses its greedy method and identifies the two most similar SCDs to merge. Hence, in the first iteration of USEM two SCDs with one referenced sentence each get merged and become one SCD with two referenced sentences. To identify similar SCDs, the cosine similarity is used with the SCD matrix' rows.

For ReFrESH, the idea is applied as follows: First, the word-vector for s_r is calculated (as in Lines 8 and 9 of Alg. 1) and s_r is added to most similar and already known SCD of the corpus (Lines 17 and 18 in Alg. 2). For each sentence in $\mathcal{S}_c$ its word-vector is then compared to all rows in the SCD matrix and to the word-vectors of the other sentences (Lines 22 and 23 in Alg. 2). If a word-vector is most similar to one of the rows in the SCD matrix, and thus to an already known SCD of the corpus, the sentence is added to the SCD as referenced sentence (Lines 25 - 27 in Alg. 2). In the other case, if a word-vector is most similar to a word-vector of another sentence, both sentences are merged to form a new SCD, which is then added to $g(\mathcal{D})$ and $\delta(\mathcal{D})$ (Lines 29 - 32 in Alg. 2). This is repeated until all sentences of $\mathcal{S}_c$ are part of an SCD. In the end of step 3), ReFrESH recalculates the word distribution of all SCDs to which new referenced sentences have been added. The new and modified SCDs are stored in $\mathcal{N}$.

4) Propagate new Relations: Finally, the SCD-based model contains all sentences again and all SCDs in the model have their word-distribution and set of referenced sentences. However, (i) the additional data of all new SCDs is empty and (ii) the SCDs that have received one or more new referenced sentences do not have the relations of their new sentences. Algorithm 2 iterates through all modified SCDs in $\mathcal{N}$ including the sentences and relations and addresses both cases.

In the case of (i), the new SCD does not have any relations itself. Furthermore, the relations of the referenced sentences of this SCD are all the same, as all sentences originate from the same disassembled SCD. Thus, the relations can be shifted back from the sentences to the SCD including the factors. The factors outline some uncertainties, as relations may not originally originate from the SCD and its sentences. Finally, a new label for the SCD is calculated by LESS [11].

Otherwise, case (ii), we assume that x new sentences have been added to an SCD with previously S sentences. In contrast to (i), the relations stay with the sentences and are also propagated to the SCD. The relations from the sentences are added to the SCD's additional data and each factor is multiplied by $\frac{x}{S+x}$. By using this factor, each relation is weighted depending the ratio it has among the referenced sentences of the SCD. The label of the SCD will not be changed, but labels from the sentences are added like a relation including the factor. The factors used with the relations by ReFrESH are slightly

Algorithm 2 Relation-preserving Feedback-reliant Enhancement of SCDs by Humans

1: **function** REFRESH($(\mathcal{D}, \delta(\mathcal{D}), g(\mathcal{D}))$, s_r, t_i)
2: **Input**: SCD-based model $(\mathcal{D}, \delta(\mathcal{D}), g(\mathcal{D}))$,
3: sentence to remove s_r, and associated SCD t_i
4: **Output**: Updated model $(\mathcal{D}, \delta(\mathcal{D}), g(\mathcal{D}))$

▷ **Step 1)** Shift Relations to Sentences
5: $\mathcal{P} \leftarrow \emptyset$ ▷ Sentences with relations to preserve
6: **for** each referenced sentence $s_i \in \{s_1, ..., s_S\}$ **do**
7: **if** $s_i = s_r$ **then** ▷ Calculate factor
8: $f \leftarrow \frac{1}{S}$, $p_i \leftarrow \emptyset$
9: **else** ▷ Preserve label for sentences in $\mathcal{S}_c$
10: $f \leftarrow \frac{S-1}{S}$, $p_i \leftarrow \{(1, l_i)\}$
11: **for** each relation to preserve $r_i \in R_i$ **do**
12: $p_i \leftarrow p_i \cup \{(f, r_i)\}$ ▷ Store with factor
13: $\mathcal{P} \leftarrow \mathcal{P} \cup \{(s_i, p_i)\}$ ▷ Store sentence and rel.

▷ **Step 2)** Disassemble SCD
14: $g(\mathcal{D}) \leftarrow g(\mathcal{D}) \setminus t_i$
15: $\delta(\mathcal{D})[i] \leftarrow Nil$ ▷ Delete i-th row

▷ **Step 3)** Reassign Sentences to SCDs
16: $\mathcal{N} \leftarrow \emptyset$ ▷ Note changed SCDs
▷ First reassign s_r
17: $j \leftarrow$ MOSTSIMILARROW($\vec{s_r}, \delta(\mathcal{D})$)
18: $\delta(\mathcal{D})[j] \leftarrow \delta(\mathcal{D})[j] + \vec{s_r}$ ▷ Update matrix
19: $t_j \leftarrow (\mathcal{C}_j, \{s_1, ..., s_S\} \cup \{s_r\})$ ▷ Add s_r to t_j
20: $\mathcal{N} \leftarrow \mathcal{N} \cup \{(t_j, s_r, p_r)\}$ ▷ Note that t_j changed
▷ Reassign remaining sentences $\mathcal{S}_c$
21: **for** each sentence and rel. $(s_i, p_i) \in \mathcal{P} \setminus (s_r, p_r)$ **do**
22: $j \leftarrow$ MOSTSIMILAR($\vec{s_i}, \delta(\mathcal{D})$)
23: $k \leftarrow$ MOSTSIMILAR($\vec{s_i}, \vec{\mathcal{P}} \setminus \vec{s_i}$)
24: **if** SIMILARITY(j) > SIMILARITY(k) **then**
25: $\delta(\mathcal{D})[j] \leftarrow \delta(\mathcal{D})[j] + \vec{s_i}$ ▷ Update matrix
26: $t_j \leftarrow (\mathcal{C}_j, \{s_1, ..., s_S\} \cup \{s_i\})$ ▷ Add s_i to t_j
27: $\mathcal{N} \leftarrow \mathcal{N} \cup \{(t_j, s_i, p_i)\}$
28: **else** ▷ Build new SCD t_k with s_i and s_k
29: $\delta(\mathcal{D})[k] \leftarrow \vec{s_i} + \vec{s_k}$ ▷ Add row to matrix
30: $g(\mathcal{D}) \leftarrow g(\mathcal{D}) \cup (\mathcal{C}_k, \{s_i, s_k\})$ ▷ Create SCD
31: $\mathcal{P} \leftarrow \mathcal{P} \setminus (s_k, p_k)$ ▷ s_k already reassigned
32: $\mathcal{N} \leftarrow \mathcal{N} \cup \{(t_k, s_i, p_i), (t_k, s_k, p_k)\}$

▷ **Step 4)** Propagate new Relations
33: **for** each SCD, sentence, and rel. $(t_i, s_i, p_i) \in \mathcal{N}$ **do**
34: **if** $\mathcal{C}_i = \emptyset$ **then** ▷ New SCD
35: $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup p_i$
36: $l_i = \text{LESS}(t_i)$
37: **else**
38: **for** each factor and relation $(f_i, r_i) \in p_i$ **do**
39: $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup \left(\frac{x}{S-x} \cdot f_i, r_i\right)$
40: **return** $(\mathcal{D}, \delta(\mathcal{D}), g(\mathcal{D}))$

inspired by weighted model counting [12].

Finally, the original association of s_r with t_i has been removed and all former referenced sentences of t_i have been reassigned to a new and better fitting SCD. All relations have been preserved and propagated to other SCDs, too.

Generally, it is possible to slightly adapt ReFrESH and remove and reassign more than one sentence from an SCD, then the factors need to be adapted. Additionally, it would be possible to ask the human users for an advice to which SCD s_r should be reassigned.

C. Algorithm ReFrESH

Based on the previous subsection presenting the four steps of ReFrESH, it is entirely formulated in Algorithm 2. ReFrESH follows the four steps and returns the updated SCD-based model as triple $(\mathcal{D}, \delta(\mathcal{D}), g(\mathcal{D}))$. In addition, the input contains the sentence to remove s_r and its SCD t_i. SCD t_i might be omitted, then all SCDs in $g(\mathcal{D})$ must be searched for the SCD associated with s_r.

We have now proposed the algorithm of ReFrESH with four steps. Next, we describe and discuss the dataset, workflow, and metrics used in our evaluation along with the results.

V. Evaluation

After we have introduced ReFrESH, we present an evaluation. First, we introduce the corpus. Afterwards, we describe workflow of the evaluation and the used metrics. Finally, we present the results and discuss the performance of ReFrESH.

A. Dataset

In this evaluation we use the Bürgerliches Gesetzbuch (BGB)[3], the civil code of Germany, in German language as corpus. The BGB is freely available and can be downloaded as XML file. Therefore, it is easily parsable and processable. As the corpus is a law text it consists of correct language, i.e., punctuation and spelling follow the orthographic rules. Thus, less preprocessing and no data cleaning is needed. Furthermore, the words used in text documents have a clear meaning and mostly the same words are used instead of using synonyms.

We use the first part of the BGB, the so called "General Part": The entire corpus consists of 228 law paragraphs and overall 854 sentences which are used as SCD windows. Each law paragraph contains between 1 and 40 sentences with an average of 3.78 sentences. The vocabulary consists of 1 436 words, where each sentence is between 1 and 20 words long with an average of 7.11 words.

B. Workflow and Implementation

ReFrESH is implemented using Python and runs inside a Docker container. The implementation uses the libraries Gensim[4], NumPy[5], and NLTK[6]. The evaluation is performed on a machine featuring 8 Intel 6248 cores at 2.50GHz (up to 3.90 GHz) and 16GB RAM. We run the following workflow to evaluate ReFrESH:

(i) Randomly choose a set of pairs of sentences which do not share a similar concept, the set contains around one eighth of all the sentences of the corpus. Each pair of sentences is associated with the same SCD and then acts as faulty associations of SCD and sentences.

(ii) Estimate a SCD-based model which contains the faulty associations chosen in (i): Use USEM with the greedy method and estimate the *faulty* SCD matrix $\delta_f(\mathcal{D})$. We add a step to USEM after the initial SCD matrix is created. This step groups each pair of sentences from (i) into the same SCD, which leads to faulty associations in the model. Afterwards, USEM continues normally, i.e., finds similar sentences and groups them into SCDs.

(iii) Run ReFrESH to update $\delta_f(\mathcal{D})$ and remove all the faulty associations initiated by the pairs of sentences from (i). Meanwhile, keep a copy of $\delta_f(\mathcal{D})$ and create the new *refresh*ed SCD matrix $\delta_r(\mathcal{D})$, where all the faulty associations have been removed.

(iv) Create a baseline model which represents the correct model for the corpus $\mathcal{D}$. Estimate the *baseline* SCD matrix $\delta_b(\mathcal{D})$ using USEM without the additional step.

(v) Compare the differences between the three models, i.e., the matrices $\delta_f(\mathcal{D})$, $\delta_r(\mathcal{D})$, and $\delta_b(\mathcal{D})$.

This workflow mainly focusses on evaluating step two (disassemble) and step three (reassign). The reassignment of sentences to new and better SCDs is the crucial and approximative part of ReFrESH. It is important to maintain the relations of the SCDs and sentences, but their treatment is fixed by the algorithm and not approximate.

C. Metrics

Based on the three matrices $\delta_f(\mathcal{D})$, $\delta_r(\mathcal{D})$, and $\delta_b(\mathcal{D})$ we need to evaluate the performance of ReFrESH. The main idea is that the distributions of our baseline $\delta_b(\mathcal{D})$ and the refreshed $\delta_r(\mathcal{D})$ should be identical. For $\delta_r(\mathcal{D})$ first some faulty associations have been added and removed afterwards by ReFrESH, while $\delta_b(\mathcal{D})$ is trained straightforward. Thus, we need to measure the difference between matrices of distributions.

Using the Hellinger distance [13], the distance between two matrices P and Q can be calculated row-wise by:

$$h_i(P,Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{j=1}^{L} \left(\sqrt{P[i][j]} - \sqrt{Q[i][j]} \right)^2}$$

The resulting distance vector $H(P,Q)$ contains in each row $h_i(P,Q)$ the distances between the matrix' rows. Based on this distance vector, we calculate two metrics:

First, the proportion of differences, which is the proportion of rows in H which are not equal to zero. It shows how many SCDs are different between two SCD matrices. Second, the average Hellinger distance, it considers only the non equal rows in H and represents the average difference in H. It shows how similar the SCDs of two SCD matrices are.

[3]https://www.gesetze-im-internet.de/bgb/, English translation https://www.gesetze-im-internet.de/englisch_bgb/

[4]https://radimrehurek.com/gensim/

[5]https://numpy.org/

[6]https://www.nltk.org/

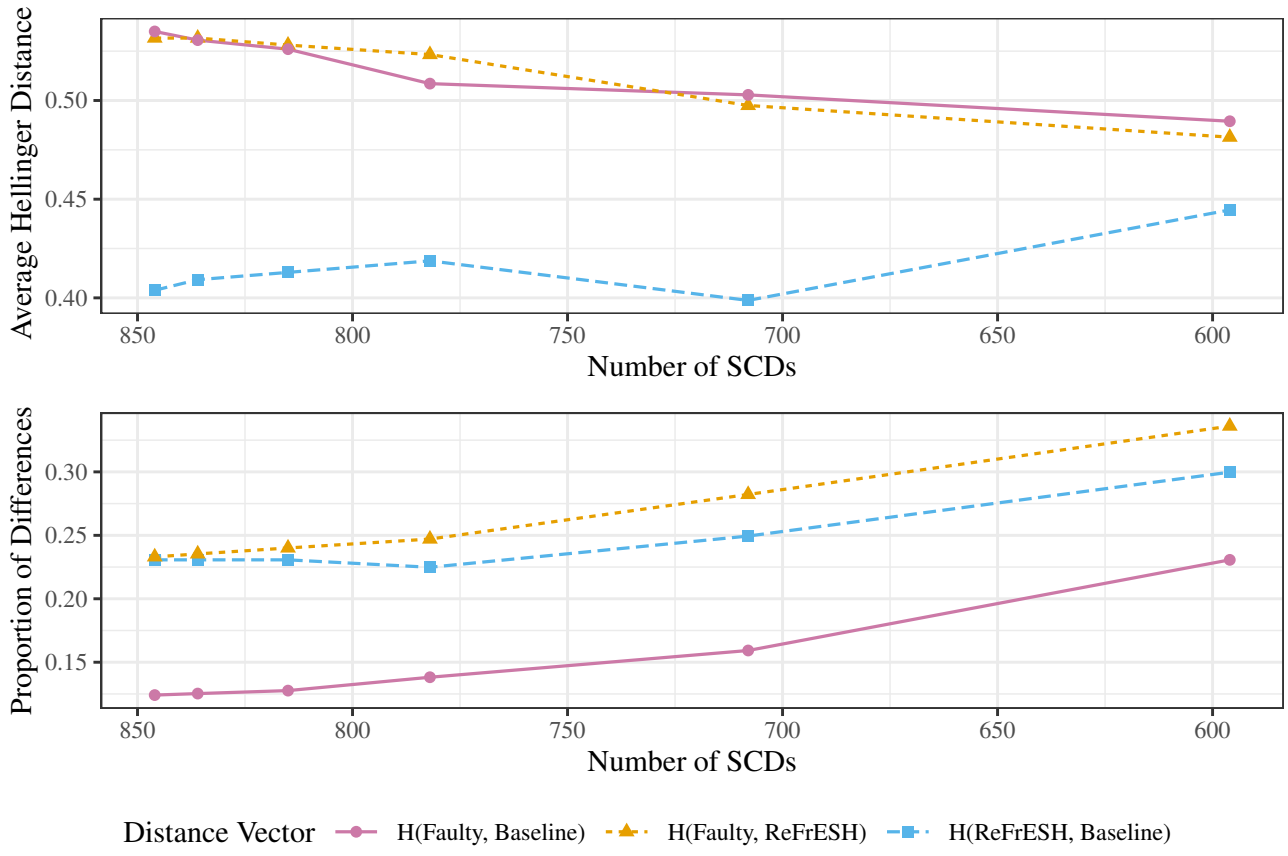

Figure 2. Proportion of different rows and average Hellinger distance value for multiple hyperparameters of USEM resulting in different numbers of SCDs. For each number of SCDs, three distances, each for one of the distance vectors between the three generated matrices, are shown.

A technical note: The distances can not be calculate between two SCD matrices directly, because the row numbers of an SCD matrix might change between multiple runs of USEM or ReFrESH. Thus, we first create intermediate matrices which use a globally equal window number and calculate the distances on these intermediate matrices.

D. Results

In this section, we present the results gained using ReFrESH and the previously described workflow. In the upper graph of Figure 2, the average Hellinger distance is shown for different numbers of SCDs. Each number of SCDs represents one run of the workflow and thus three matrices *faulty*, *baseline*, and *refresh*ed. Using the three matrices we calculate three Hellinger distance vectors $H(Faulty, Baseline)$, $H(Faulty, ReFrESH)$, and $H(ReFrESH, Baseline)$ and for each vector both metrics.

The performance of ReFrESH is especially shown by the dashed blue line of $H(ReFrESH, Baseline)$. A perfect deletion of faulty associations from the SCD matrix would result in a distance of zero. In this case, the distance is greater than zero, but well below the other two lines. The solid purple line shows $H(Faulty, Baseline)$, which can be interpreted as the *error* an SCD-based model would have without ReFrESH, because a faulty matrix with all the faulty associations is compared to the baseline. Since the dashed blue line is below the solid purple line, ReFrESH reduces the *error* of the model by removing faulty associations.

In the lower graph of Figure 2, the proportions of different rows are shown—again for different numbers of SCDs and based on three Hellinger distance vectors. The two dashed lines represent a distance to $ReFrESH$ and are above the solid purple line of $H(Faulty, Baseline)$. A smaller amount of different rows in $H(Faulty, Baseline)$ may be explained by the fact that both models use USEM. In contrast, ReFrESH is a different technique and reassigns sentences to other SCDs which in total affects more SCDs.

In Figure 3, the reduction of the Hellinger distance by running ReFrESH is shown. It shows the average difference of $H(Faulty, Baseline)$ and $H(ReFrESH, Baseline)$. In other words, the space between the dashed blue and solid purple line in the upper part of Figure 2. Hence, the value can be seen as an improvement of the model when using ReFrESH.

At first glance, the improvement might be a bit small. However, ReFrESH leads to matrices with more different rows but with smaller distances of each row. In comparison, the baseline and the faulty model share some identical rows, with each distance value being significantly larger. ReFrESH does the reassignment of the sentences to SCDs with the goal of

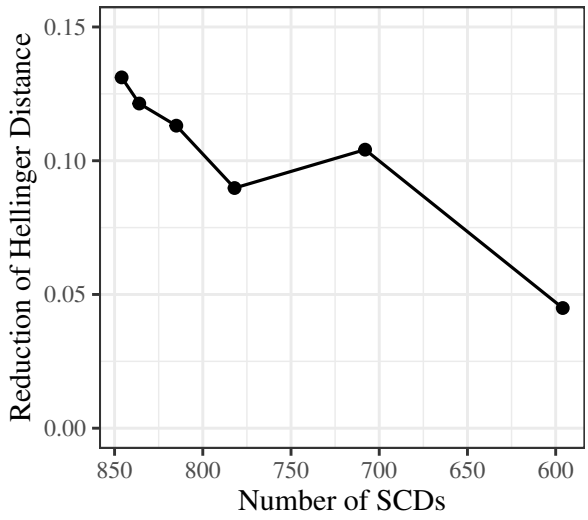

Figure 3. Reduction of the Hellinger distance when running ReFrESH on the faulty model and comparing to the baseline, i.e., the amount of correction done by ReFrESH.

finding a better matching SCD. To do so, ReFrESH needs to change many SCDs with the goal of getting a slightly changed but better model. Summarized, ReFrESH provides a good performance for refreshing an SCD-based model based on, e.g., human, feedback.

VI. Conclusion

This paper introduces ReFrESH, an approach consisting of four steps for incorporating feedback in SCD-based models. SCDs are additional information associated with corpora of text documents and highlight points of interest nearby their location. In general, when reading a text document, each human gets its own perceptions and views of the text document. Hence, SCDs are slightly different depending on the human or automated annotation technique used to create them. If SCDs are used by an IR agent, a user may consider some answers of the agent faulty and respond with feedback to the agent. In this case, the agent can use ReFrESH to update its SCDs. It allows incremental updates of SCD-based models based on user feedback and avoids the need for each user to create their own SCDs for each corpus from scratch.

In the first step, ReFrESH shifts all relations among SCDs to the sentences to ensure that relations between SCDs are preserved during the update. Second, the SCD to be updated is disassembled before each sentence is reassigned to a better fitting SCD in the third step. Finally, the preserved relations are propagated back from the sentences to the SCDs.

Overall, the evaluation shows that ReFrESH works well and provides a powerful technique to update SCD-based models based on human feedback. Using ReFrESH, faulty association between sentences and SCDs can be removed and the sentences get associated with new and better fitting SCDs. The evaluation focusses on step two (disassemble) and step thee (reassign) because the crucial and approximative part of ReFrESH is the reassignment of an SCD.

In the field of SCDs, ReFrESH provides an important step for using SCD-based models in IR agents. The agent maintains an SCD-based model to provide its IR service. Using ReFrESH the agent is now able to incorporate feedback from its users and update its model based on this feedback.

Additionally, ReFrESH introduces relations weighted by factors between SCDs and the sentences of a corpus. This provides a graph grounded to a corpus of text documents, which can possibly be used as input for techniques from the field of factor graphs.

Acknowledgment

The authors thank the AI Lab Lübeck for providing the hardware used in the evaluation.

References

[1] F. Kuhr, T. Braun, M. Bender, and R. Möller, "To Extend or not to Extend? Context-specific Corpus Enrichment," *Proceedings of AI 2019: Advances in Artificial Intelligence*, pp. 357–368, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-35288-2_29

[2] G. Angeli, M. J. Johnson Premkumar, and C. D. Manning, "Leveraging linguistic structure for open domain information extraction," *Proceedings of the Association of Computational Linguistics (ACL)*, pp. 344–354, 2015. [Online]. Available: https://doi.org/10.3115/v1/P15-1034

[3] M. Bender, T. Braun, R. Möller, and M. Gehrke, "Unsupervised estimation of subjective content descriptions," *Proceedings of the 17th IEEE International Conference on Semantic Computing (ICSC-23)*, 2023. [Online]. Available: https://doi.org/10.1109/ICSC56153.2023.00052

[4] S. Russell and P. Norvig, *Artificial Intelligence, Global Edition A Modern Approach*. Pearson Deutschland, 2021. [Online]. Available: https://elibrary.pearson.de/book/99.150005/9781292401171

[5] M. Bender, K. Schwandt, R. Möller, and M. Gehrke, "Fresh – feedback-reliant enhancement of subjective content descriptions by humans," in *Proceedings of the Workshop on Humanities-Centred Artificial Intelligence (CHAI 2023)*. CEUR Workshop Proceedings.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. [Online]. Available: https://arxiv.org/abs/1810.04805

[7] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[8] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[9] A. A. Ginart, M. Y. Guan, G. Valiant, and J. Zou, "Making AI forget you: Data deletion in machine learning," 2019.

[10] Z. Izzo, M. A. Smart, K. Chaudhuri, and J. Y. Zou, "Approximate data deletion from machine learning models," in *International Conference on Artificial Intelligence and Statistics*, 2021.

[11] M. Bender, T. Braun, R. Möller, and M. Gehrke, "LESS is More: LEan Computing for Selective Summaries," in *KI 2023: Advances in Artificial Intelligence*. Springer International Publishing, 2023, pp. 1–14. [Online]. Available: https://doi.org/10.1007/978-3-031-42608-7_1

[12] T. Sang, P. Beame, and H. A. Kautz, "Performing bayesian inference by weighted model counting," in *AAAI Conference on Artificial Intelligence*, 2005.

[13] E. Hellinger, "Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen," *Journal für die reine und angewandte Mathematik*, pp. 210–271, 1909.