Estimating Context-Specific Subjective Content Descriptions using BERT

Magnus Bender, Felix Kuhr, Tanya Braun, Ralf Möller
University of Lübeck
Institute of Information Systems
Ratzeburger Allee 160, 23562 Lübeck
{bender, kuhr, braun, moeller}@ifis.uni-luebeck.de

***Abstract*—An agent in pursuit of a task may work with a corpus containing documents associated with Subjective Content Descriptions (SCDs) that add value in the context of the agent's task. On the pursuit of new documents to add to the corpus, an agent may come across documents without associated SCDs or documents where *content* and SCDs are interleaved. Therefore, this paper presents approaches estimating SCDs using the well-known BERT [1] language model. Furthermore, the paper presents approaches separating SCDs and actual *content* given interleaved in a document also using BERT. An extensive evaluation compares the performance of the approaches using BERT to prior approaches.**

I. Introduction

An agent in pursuit of a task, explicitly or implicitly defined, may work with a corpus of text documents as a reference library. From an agent-theoretic perspective, an agent is a rational, autonomous unit acting in a world fulfilling a defined task, e.g., providing document retrieval services given requests from users. We assume that the corpus represents the context of the task, since collecting documents is not an end in itself. Further, documents in a given corpus might be associated with additional location-specific data making the content nearby the location explicit by providing descriptions, references, or explanations. We refer to these location-specific data as Subjective Content Descriptions (SCDs) [2].

For example, humans reading the word *bank* in a text dealing about money, would assume the *bank* to be a financial institution and not something to sit on. To clarify, the SCD could be another sentence, defining the *bank* as a financial institution, or a link to an entity *financial institution* in an external source.

Coming back to an agent providing a document retrieval service, we assume the agent maintains a corpus of text documents and then retrieves documents from its corpus based on the documents' SCDs. However, typically text documents are not associated with SCDs or SCDs and actual *content* are interleaved in a text document. In both cases, the agent has to estimate the SCDs for the documents in its corpus.

The contributions of this paper are approaches to estimate SCDs in both cases using the Transformer Language Model (TLM) [3] Bidirectional Encoder Representations from Transformers (BERT) [1]. We present how to separate SCDs and actual *content* given interleaved in text documents using BERT. Further on, we describe how to use BERT to automatically associate SCDs to a text document without associated SCDs.

In an extensive evaluation, we compare the performance of the approaches using BERT with the approaches introduced by Kuhr et al. [4], [2]. Further, we demonstrate the capacity of our approaches by solving a more realistic and advanced scenario. Given two corpora featuring two contexts, e.g., scientific papers and children's books, a model representing both corpora has to estimate SCDs depending on the context. Thus, for a document without associated SCDs the model first has to determine the context and then has to estimate context-specific SCDs. We present a *context-sensitive model* using our approaches and differing only in the way the training data is arranged.

The remainder of this paper is structured as follows: We start with related work and then recap the basics of SCDs and TLMs. Afterwards, we describe two problems from the field of SCDs, namely the Most Probably Suited SCD and inline SCD problem. We introduce approaches solving each problem using BERT. Finally, we evaluate our approaches on the well-known 20 newsgroups dataset[1] and generate SCDs using definitions from the online dictionary Wiktionary[2].

II. Related Work

Adding data to corpora of text documents has been investigated for a long time. Often the data associated with a corpus is denoted an annotation. Thus, an SCD is an annotation subjectively describing the content depending on the corpus' context. The Brown Corpus [5] is one of the first corpora used to analyze natural language. First, the distribution of words among different categories and contexts of natural language was analyzed. Later, *part-of-speech* tags were added, these tags can already be interpreted as annotations assigning a class to each word.

In the beginning of natural language annotation, most annotations had to be manually added to the corpora. Even today, crowdsourcing can be used to manually annotate text documents [6]. Furthermore, semi-automatic and automatic annotation systems were developed, too, e.g, DBpedia[3] and OpenCalais[4].

[1] http://qwone.com/~jason/20Newsgroups/
[2] https://en.wiktionary.org/
[3] https://www.dbpedia.org/

Since 2017, TLMs have shown that they are a powerful technique processing natural language and reached remarkable improvements in the field of Natural Language Processing (NLP). Compared to approaches used in the field of NLP before, TLMs can handle much larger amounts of data using a special architecture of neural networks. Thus, TLMs allow to build larger and more powerful models, while they can be used for the same NLP related tasks. TLMs can be used to translate text documents [3] and generate sentences [7]. BERT, the TLM used in this paper, is an encoder and calculates vector representations for text documents.

III. Preliminaries

This section specifies notations and describes the concept of SCDs and BERT.

A. Notations

Before we introduce SCDs and BERT, we formalize our setting of a corpus.

- A word w_i, $i = 1, ..., L$, is a basic unit of discrete data from a vocabulary $\mathcal{V} = \{w_1, \ldots, w_L\}$, $L \in \mathbb{N}$.
- A document d is defined as a sequence of words $(w_1^d, \ldots, w_N^d)$, $N \in \mathbb{N}$, where each word $w_i^d \in d$ is an element of vocabulary $\mathcal{V}$.
- A subsequence of words from d can be represented as $w_{i,j}^d = (w_i^d, ..., w_j^d)$ where $1 \leq i < j \leq N$. Commonly used subsequences are sentences, they are defined as a sequence of words terminated by punctuation symbols like ".", "!", or "?".
- A corpus $\mathcal{D}$ represents a set of documents $\{d_1, \ldots, d_{|\mathcal{D}|}\}$, $|\mathcal{D}| \in \mathbb{N}$.
- An SCD $t = (w'_1, ..., w'_l)$, $l \in \mathbb{N}$, is a sequence of words. The SCD t can be associated with a position $\rho \in [i, j]$ in a document d. We use the term located SCD interchangeably for associated SCD and represent a located SCD t by the tuple (t, ρ).
- For each document $d \in \mathcal{D}$ there exists a set g denoted as SCD set containing M located SCDs $\{(t_j, \rho_j)\}_{j=1}^{M}$. Given a document d, the term $g(d)$ refers to the set of located SCDs associated with document d. The set of all located SCD tuples in $\mathcal{D}$ is given by $g(\mathcal{D}) = \bigcup_{d \in \mathcal{D}} g(d)$.
- For each located SCD $(t, \rho) \in g(d)$ there exists an SCD window $win_{d,\rho} \subseteq w_{1,N}^d$ that refers to a sequence of words in d surrounding the word w_ρ^d. In our case the SCD window is represented by the sentence w_ρ^d belongs to.
- Each word $w^d \in win_{d,\rho}$ is associated with an influence value $I(w^d, win_{d,\rho})$ representing the distance in the text between w^d and position ρ. The closer w^d is positioned to ρ in $win_{d,\rho}$, the higher its corresponding influence value $I(w^d, win_{d,\rho})$. The influence value is chosen according to the task and might be distributed binomial, linear, or constant.

B. Subjective Content Descriptions

Kuhr et al. have introduced SCDs in [2]. SCDs provide additional location-specific data for documents. The data provided by SCDs may be of various types, like additional definitions or links to knowledge graphs. However, in our evaluation and use-cases we use text documents annotated with additional textual definitions.

Algorithm 1 Training the SCD-word distribution matrix $\delta(\mathcal{D})$

```
function BUILDMATRIX(D, g(D))
    Input: Corpus D, Set of SCDs g(D)
    Output: SCD-word distribution matrix δ(D)
    Initialize an M × L matrix δ(D) with zeros
    for each d ∈ D do
        for each (t, ρ) ∈ g(d) do
            for each w^d ∈ win_{d,ρ} do
                δ(D)[t][w^d] += I(w^d, win_{d,ρ})
    return δ(D)
```

Kuhr et al. use an SCD-word distribution represented by a matrix when working with SCDs. 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 near the SCD's location in the document.

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:

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

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

The supervised training of an SCD matrix is described in Algorithm 1. Given a corpus $\mathcal{D}$, the algorithm iterates over each document in the corpus and the document's located SCDs. For each located SCD given by a tuple (t, ρ), the SCD matrix is updated following a sentence-wise approach: First, the sentence in d at position ρ is reconstructed and represented by $win_{d,\rho}$. Next, the row of the matrix representing SCD t gets incremented for each word in the sentence $win_{d,\rho}$..

Kuhr et al. use a sliding window instead of our previously described sentence-wise approach. The authors assume an SCD generates the words in a certain radius around the SCD's location, while we assume an SCD generates the words of the sentence at the SCD's location. The sentence-wise approach is required in this paper due to the comparability to BERT working on whole sentences. Furthermore, a sliding window results in more computations and as we use larger corpora than Kuhr et al., sentence-wise iteration allows us to keep the number of computations sufficiently low.

After Algorithm 1 has finished, the SCD matrix needs to be normalized row-wise to meet the requirements of a probability distribution. However, we skip the normalization because multiple calculations on small decimal values on a computer reduce the accuracy. Later, we use the cosine similarity with the rows of the matrix and the cosine similarity does a normalization by definition. Thus, by skipping the normalization, we save computational resources and get slightly more accurate results.

The SCD matrix is a basic model used in IV to solve two problems from the field of SCDs.

C. Transformer Language Models

In this subsection, we describe Transformer Language Models (TLMs) and focus on the Bidirectional Encoder Representations from Transformers (BERT) [1].

Vaswani et al. [3] have introduced TLMs in 2017. TLMs are special neural networks using so-called transformer units and the so-called attention mechanism. Today, TLMs are a common technique in the field of NLP, thus, we omit further descriptions and refer to [3]. TLMs have been adapted in several ways and many succeeding models have been created.

A well-known TLM is BERT, introduced by Devlin et al. [1] in 2019. BERT is an encoder and encodes a sequence of input tokens, i.e. words, into a sequence of vector representations. In addition to the encoded vector for each word, BERT features a *class* output representing the entire input sequence in one vector.

Training BERT requires two stages, the pre-training and fine-tuning. The pre-training is done on a huge corpus and pursues the objective to understand natural language. A pre-trained model provides a general language understanding and context dependent encoding of input tokens. Normally, such pre-trained models do not facilitate a special task and are available for public download.

Afterwards, the fine-tuning is done on a previously pre-trained model. During the fine-tuning the model is trained by labeled inputs and "learns" how to solve its special use-case, often including learning linear classifiers for the encoded vectors from scratch. As the model already provides an understanding of natural language, the fine-tuning may be done on less data and runs much faster.

In the following sections, BERT is depicted as a box getting the input tokens at the bottom and yielding the encoded vector outputs at the top. The sequence of input tokens contains two special tokens, `[CLS]` marks the beginning and `[SEP]` separates two sentences.

Next, we apply use-cases of BERT to two problems from the field of SCDs and fine-tune a pre-trained model for each use-case.

IV. Transformer Language Models for Subjective Content Descriptions

Using the theoretical foundations of SCDs and TLMs introduced in the previous section, the contribution of this paper is to combine the concept of SCDs with TLMs and use BERT to solve problems from the field of SCD. We select two problems Kuhr et al. have introduced along with SCDs and describe the solution presented by Kuhr et al. as well as present our solution using BERT. For each of the two problems, we present how to apply two different use-cases of BERT. Thus, in the evaluation we compare the performance of both approaches using BERT with the approach presented by Kuhr et al.

BERT gets a sequence of tokens as input. We decide to use one sentence as one input sequence and a token for each word in the sentence. We argue that this sentence-wise approach maintains the logical structure of the documents. As the most influential words of a word belong to the same sentence as the word itself.

Next, we present the first problem along with the solution by Kuhr et al. and the solution using BERT before addressing the second problem.

A. Identifying Subjective Content Descriptions

In the scenario of the inline SCD (iSCD) problem [4], documents are annotated with SCDs, but the SCDs are not separated from the *content* of the documents. For each word of each document, the agent has then to decide whether the word is part of an SCD or belongs to the *content*. The iSCD problem asks to separate SCDs and *content* given interleaved in a document d'. Formalized, the iSCD problem's input is a document $d' = (w_1^{d'}, ..., w_N^{d'})$ and the output is the *content* d as sequence of words and a set of SCDs $g(d)$.

Example 1 (Inline SCD Example)**.** *Assume a new document d' contains the following sentence with two SCDs interleaved. The underlined words represent the SCDs, while other the words form the* content*.*

"We visited the bisons large animals in the zoo a place where non-domestic animals are exhibited."

Document d' can be represented as a sequence of words.

$$d' = (w_1^{d'}, w_2^{d'}, w_3^{d'}, w_4^{d'}, \underline{w_5^{d'}, w_6^{d'}}, w_7^{d'}, w_8^{d'}, w_9^{d'}, \underline{w_{10}^{d'}, w_{11}^{d'}, w_{12}^{d'}, w_{13}^{d'}, w_{14}^{d'}, w_{15}^{d'}, w_{16}^{d'}})$$

After solving the iSCD problem, the result would be:

$$\begin{aligned} d &= (w_1^{d'}, w_2^{d'}, w_3^{d'}, w_4^{d'}, w_7^{d'}, w_8^{d'}, w_9^{d'}) \\ g(d) &= \{ \quad \{(w_5^{d'}, w_6^{d'}), 4\}, \\ &\qquad \{(w_{10}^{d'}, w_{11}^{d'}, w_{12}^{d'}, w_{13}^{d'}, w_{14}^{d'}, w_{15}^{d'}, w_{16}^{d'}), 9\} \quad \} \end{aligned}$$

Applying the sentence-wise approach, the input for the iSCD problem is a sequence of sentences to be distinguished into the sentences being SCDs and the sentences being *content*. Thus, the iSCD problem is a classification problem with two classes, namely SCD and *content*.

1) Solving the iSCD Problem with the SCD Matrix: In [4] the authors propose the following three approaches solving the iSCD problem: (i) the word-based approach, (ii) the threshold-based approach, and (iii) the Hidden Markov Model (HMM)-based approach. Approach (ii) and (iii) use the SCD matrix described in III-B.

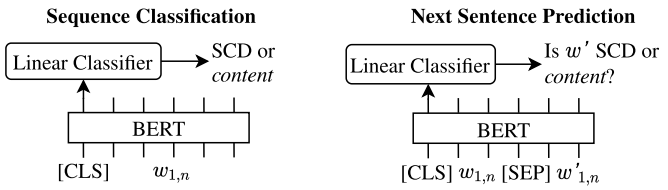

Figure 1. Use-cases of BERT solving the iSCD problem.

We decide to use the threshold-based approach for our evaluation. Even though the performance of the HMM-based approach is slightly better, choosing a good threshold th results in a nearly equal performance and the threshold-based approach simplifies the implementation significantly.

2) Solving the iSCD Problem with BERT: BERT's use-cases sequence classification and next sentence prediction can be used to classify the input sequence. In Figure 1 both use-cases are depicted already adjusted to SCDs.

For each use-case we describe the general operating principle and the adaptations made for SCDs.

a) Sequence Classification: A single sentence is used as input. The task of the model is to assign a class to the sentence. The classes assigned are taken from the class labels in the training data used for fine-tuning. The classification uses a linear classifier at BERT's *class* output.

Using sequence classification to solve the iSCD problem by BERT is straightforward. For each sentence as input sequence the encoded representation is calculated. From the encoded representation, only the vector at the *class* output is needed to classify the sentence as SCD or *content*. We train a linear classifier for the vectors at the *class* output.

We fine-tune the model with $\mathcal{D}$ and measure the model's performance on a different $\mathcal{D}'$. Especially, the sets of SCDs are disjoint, i.e., $g(\mathcal{D}) \cap g(\mathcal{D}') = \emptyset$. The disjoint sets of SCDs are important to prevent the model from simply memorizing all SCDs. To prevent the model from remembering undesired relations, we also randomly shuffle the sentences such that no pattern of occurrence between SCD and *content* exists.

A disadvantage when using sequence classification is that the relation between sentences and their associated SCDs is not modeled. Each sentence, whether SCD or *content*, gets classified independently.

b) Next Sentence Prediction: A pair of two sentences is used as input. The task of the model is to classify if both sentences are in a relation. The relation between both sentences is given by boolean labels in the training data used for fine-tuning. Again, a linear classifier at BERT's *class* output is used.

Applied to SCDs, we specify to model the relation between a sentence from the *content* and its associated SCD. The model is fine-tuned on tuples of two sentences, meaning the first sentence is always a sentence from the *content* and the second sentence may be a related SCD or the subsequent sentence from the *content*. Thus, the model classifies the second sentence as (related) SCD or no SCD.

As with sequence classification, we use different corpora and disjoint sets of SCDs, as well as shuffle the tuples of sentences.

Next, we present the second problem from the field of SCDs along with solutions.

B. Estimating Most Probably Suited Subjective Content Descriptions

In the scenario of the Most Probably Suited Subjective Content Descriptions (MPS²CDs) problem documents are not associated with SCDs. The MPS²CD problem asks for the M most probably suited SCDs $t_1, ..., t_M$ for a document d' given the SCD matrix $\delta(\mathcal{D})$:

$$g(d') = \underset{t_1,...,t_M \in g(\mathcal{D})}{\arg\max} P(t_1, ..., t_M \mid d', \delta(\mathcal{D}))$$

The definition of the MPS²CD problem does not consider the sentence-wise iteration used while training the SCD matrix. We can reformulate the MPS²CD problem to consider the sentence-wise iteration:

$$g(d') = \bigcup_{\text{sentences } win_{d',\rho} \in d'} \underset{t \in g(\mathcal{D})}{\arg\max} P(t \mid win_{d',\rho}, \delta(\mathcal{D}))$$

Solving the MPS²CD problem allows us to estimate SCDs from the set of SCDs known by $\delta(\mathcal{D})$ for each sentence in a document.

1) Solving the MPS²CD Problem with the SCD Matrix: Analogous to the reformulated MPS²CD problem, an algorithm solving the problem iterates over each sentence of d'. For each sentence the algorithm creates a vector representing the words of the sentence $\delta(win_{d',\rho})$. The vector is created using the approach that was used for the rows of the matrix in Algorithm 1. Then the cosine similarity is used to compare $\delta(win_{d',\rho})$ with each row of the SCD matrix $\delta(\mathcal{D})[t]$ representing SCD t. The most probably suited SCD t' is defined as the SCD belonging to the row resulting in the highest cosine similarity value.

$$t' \leftarrow \underset{t \in g(\mathcal{D})}{\arg\max} \frac{\delta(\mathcal{D})[t] \quad \cdot \quad \delta(win_{d',\rho})}{\|\delta(\mathcal{D})[t]\|_2 \cdot \|\delta(win_{d',\rho})\|_2}$$

The MPS²CD algorithm allows us to estimate the most probably suited SCDs for any sentence given the words of the sentence and the SCD matrix.

2) Solving the MPS²CD Problem with BERT: BERT's use-cases multiple choice and question answering can be used to choose, given an input sequence, a most probably suited answer from set of possible answers. In Figure 2 both use-cases are depicted already adjusted to SCDs.

For each use-case we describe the general operating principle and the adaptations made for SCDs. Finally, we outline the difference between the solution by Kuhr et al. and our solutions using BERT.

a) Multiple Choice: A single sentence, the query, together with a set of sentences, the possible answers, is used as input. The task of the model is to select the best answer from the set of answers for the query.

Multiple choice uses next sentence prediction multiple times. For each possible answer to the query, a pair of query

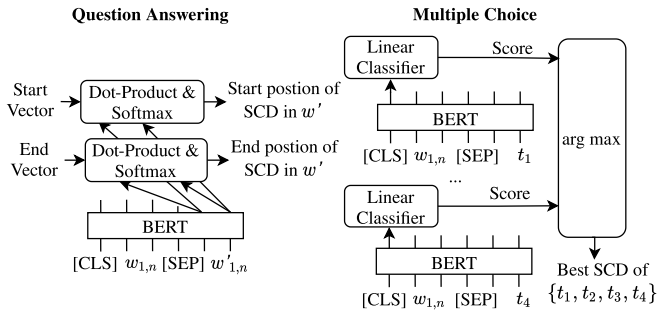

Figure 2. Use-cases of BERT solving the MPS²CD problem.

and answer is passed to the next sentence prediction. Then for each pair, the next sentence prediction returns a score and the answer in the pair reaching the best score is returned as solution for the multiple choice use-case.

Using the multiple choice use-case to solve the MPS²CD problem by BERT is straightforward. For each sentence a set of four SCDs $\{t_1, t_2, t_3, t_4\}$ is given. We offer four SCDs to BERT because the duration running BERT four times is acceptable and well-known tasks like SWAG [8] also provide four options to choose from. In the set of four SCDs, the solution always has to be unique, i.e., one SCD may be associated with the sentence while the other three must not be associated.

We fine-tune and measure the performance of the model again on different corpora and disjoint sets of SCDs. However, we do not shuffle the ordering of the sentences in the documents and only randomly choose at which of the four positions the correct SCD is presented to the model.

b) Question Answering: A single sentence, the question, and a short document containing the answer are used as input. The task of the model is to select the answer to the question in the short document. The selection is done by returning a start and end position and thus an interval. Then, the answer generated by the model is formed by the words of the short document contained in the interval.

During the fine-tuning two vectors are trained. The *start* vector is used to calculate the start position and the *end* vector to calculate the end position, respectively. The data used during fine-tuning contains start and end positions as labels. To calculate the start position, each output vector resulting from a token of the short document is dot-multiplied by the *start* vector. The softmax across all dot-products gives the start position. The end position is calculated the same way using the *end* vector.

Analogous to multiple choice, the question answering use-case gets a sentence and four SCDs to select one SCD from. The four SCDs are randomly shuffled and concatenated to a short document. The sentence and the short document are then fed into BERT. BERT returns an interval and selects the words in this interval of the short document as SCD for the sentence.

The input sequence must not exceed BERT's size of 512 tokens, concatenating four SCDs might result in too long documents. If a document gets too long, we first try to omit one or two SCDs and ignore the entire sample in the end.

We fine-tune and measure the performance of the model again on different corpora. Once more, we use disjoint sets of SCDs as well as, in addition, the same set of possible SCDs for both corpora.

c) Notable Difference: The MPS²CD algorithm provided by Kuhr et al. returns an SCD known by the model and does not select a best SCD from a given set. To validate the solution of the MPS²CD algorithm, we have to compare two SCDs, the SCD labeled as correct in the given set and the SCD returned by the MPS²CD algorithm. Especially, when the sets of SCDs used for training and testing are disjoint, the two SCDs will never be equal but might be both correct.

In our evaluation, the SCDs are gained from an agent. This agent allows inverse queries, i.e., the agent can tell if it would annotate a sentence with both annotations. If the agent would do so for both SCDs, we assume the MPS²CD algorithm selected the correct SCD.

Next, we present how to combine multiple corpora with different contexts and SCDs such that our approaches can handle multiple corpora jointly without further changes to the approaches.

C. Context-Specific Subjective Content Descriptions

The title of this paper states to estimate context-specific SCDs. The estimation is primarily situated in the MPS²CD problem, but we didn't consider the specific contexts until now.

In the previous sections, we have always assumed context-specific SCDs because the collection of documents in a corpus always represents a specific context for us. The corpora and the SCDs represent a context and the model silently learns that context. However, in a more realistic scenario, a model also has to disambiguate between multiple contexts. For example, a sentence in a scientific paper should be annotated with different SCDs than a sentence in a children's book.

Determining a context and estimating MPS²CDs for a document can be realized with two distinct models. The first model selects the contextually most similar corpus to the document from a set of known corpora, and depending on the selected corpus, a second model trained on the selected corpus then estimates the MPS²CDs of the document. However, we propose a *context-sensitive model* combining both steps. We can use the same approaches and problems introduced previously in IV-A and IV-B. The corpora of different contexts and their SCDs only need to be combined into one large corpus as follows:

Given are two corpora $\mathcal{D}_{c_1}, \mathcal{D}_{c_2}$ representing two different contexts c_1, c_2 with their context dependent SCDs $g(\mathcal{D}_{c_1}), g(\mathcal{D}_{c_2})$. We create a combined corpus $\mathcal{D}$ and its SCDs $g(\mathcal{D})$ to train the *context-sensitive model* on:

$$\mathcal{D} = \mathcal{D}_{c_1} \cup \mathcal{D}_{c_2}, \quad g(\mathcal{D}) = g(\mathcal{D}_{c_1}) \cup g(\mathcal{D}_{c_2})$$

$\mathcal{D}$ and $g(\mathcal{D})$ are formed by the union while taking care to update the positions of the located SCDs.

In the combined corpus the *context-sensitive model* now represents the originating corpus and the MPS²CD of the

originating corpus together. Thus, the *context-sensitive model* estimates for a sentence an MPS²CD matching the sentence's context.

Next, we present an extensive evaluation of our approaches using BERT to solve two problems from the field of SCDs and evaluate the *context-sensitive model*.

V. Evaluation

After we have introduced overall four approaches solving the iSCD or MPS²CD problem with BERT, we present an evaluation. For each problem we compare the performance of the two approaches using BERT to the approach by Kuhr et al. using an SCD matrix. Especially, we demonstrate that both, BERT and the SCD matrix, are capable techniques to model the relations of SCDs and sentences.

A. Datasets

In this evaluation we use the 20 newsgroups[5] dataset. 20 newsgroups is a well-known corpus consisting of e-mails from 20 e-mail newsgroups. Thematically, the 20 newsgroups can be divided into six topics. The entire corpus consists of 18 828 text documents. The documents have between 1 and 39 682 words with a median of 160 words.

For training the SCD matrix and fine-tuning BERT on the 20 newsgroups dataset, we need not only the documents but also SCDs associated with each sentence in the documents. However, documents in the 20 newsgroups dataset are not associated with SCDs. Therefore, we use definitions from the online dictionary Wiktionary[6] and annotate each sentence with a definition from Wiktionary acting as SCD. The set of definitions from Wiktionary contains in total 293 296 definitions for 201 688 different words and phrases.

The Wiktionary annotation agent allows us to automate the annotation of documents and generates $g(\mathcal{D})$ for any corpus $\mathcal{D}$. For the same sentence, the agent always returns the same SCD, while trying to maximize the variance of SCDs for similar but different sentences. The Wiktionary annotation agent also allows to generate non-matching SCDs for a sentence and thus gives negative samples. Furthermore, it is possible to query the agent inversely and check if an SCD describes a sentence.

To evaluate the *context-sensitive model* proposed in IV-C, a second annotation agent and a second dataset is needed. The second annotation agent works similar to the Wiktionary annotation agent, but uses annotations from the 500 000 quotes [9] dataset. As second dataset we use Manuscript cultures[7], an openly accessible journal publishing exhibition catalogues and articles from the field of written artefacts. The two datasets Manuscript cultures and 20 newsgroups provide two different contexts, namely *written artefacts* and *computer and science*.

B. Workflow

In the evaluation workflow, the experiments run on two different platforms. All experiments using the SCD matrix run on a virtual machine featuring 8 Intel 6248 cores at 2.50GHz (up to 3.90GHz) and 16GB RAM. However, the virtual machine does not provide a graphics card needed for the used Huggingface Transformers implementation of BERT. Thus, all experiments using BERT run on an NVIDIA A100 40GB graphics card of an NVIDIA DGX.

We run all experiments five times and take the arithmetic mean of the resulting scores to increase the statistical correctness. Each experiment follows a similar procedure:

1) Download the corpus and the set of SCDs for the annotation agent.
2) Lowercase all characters, stem the words, tokenize the sentences and eliminate stop words from a wordlist containing 179 words. These four tasks are called preprocessing tasks. We perform them on the corpus and the set of SCDs for the annotation agent. Preprocessing a text of a document transforms the text in a more digestible form for machine learning algorithms and increases their performance [10].
3) Split the corpus randomly into a training set containing 80% of the documents and a test set containing the remaining 20%. If a disjoint set of SCDs is used in the current experiment, the set of SCDs for the annotation agent is also split into 80% and 20% of the definitions.
4) Generate the SCDs for the training set and test set by the annotation agent. If the iSCD problem is evaluated in the current experiment, also generate documents containing randomly interleaved SCDs and *content*.
5) On the training set train the SCD matrix or fine-tune BERT, depending on the current experiment. We use the pre-trained `bert-base-uncased`[8] version of BERT. This version of BERT is case insensitive and a standard model to fine-tune for downstream tasks.
6) Evaluate the performance of the trained model on the test set.

C. Performance Measures

For each sample in the test set a prediction is generated using the trained model and the predicted value is compared against the sample's label. Given the number of samples predicted correct and wrong, the accuracy is defined by

$$accuracy = \frac{\#correct}{\#correct + \#wrong}.$$

Furthermore, we calculate for the intervals returned by BERT's question answering use-case the following measures: The interval similarity introduced by Kabir et al. [11] takes the overlapping ratio of intervals into account and yields a score between 0 and 1. We use the mean interval similarity across all samples in the test set.

Allen's interval algebra [12] provides the relations *during* and *contains* between two intervals. The predicted interval may *contain* the SCD or the predicted interval may be located *during* the SCD. We count across all samples in the test set,

[5] http://qwone.com/~jason/20Newsgroups/

[6] https://en.wiktionary.org/

[7] https://www.csmc.uni-hamburg.de/publications/mc.html

[8] https://huggingface.co/bert-base-uncased

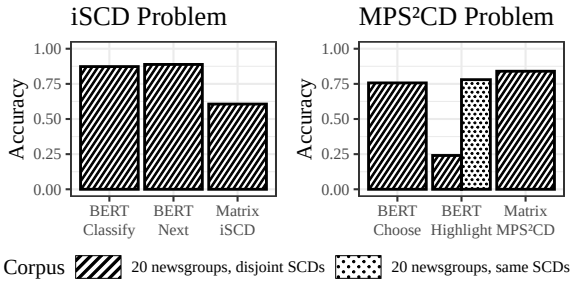

Figure 3. Accuracies gained for all scenarios.

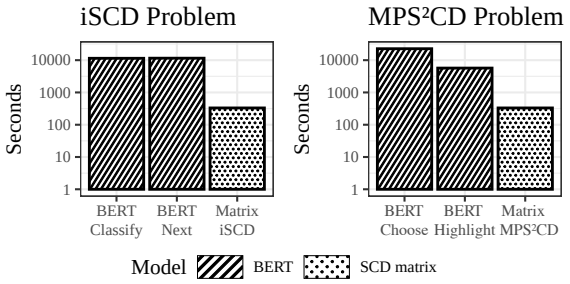

Figure 4. Time needed training the models for all scenarios.

how often the predicted and labeled intervals are equal or in each of the two relations. Using this counts, we can calculate accuracies for *during* and *contains*, too.

D. Results

In this section, we present results gained using the previously described workflow for experiments. First, we define for each problem and each approach a scenario to measure the performance in an experiment:

`Matrix iSCD`: This scenario uses an SCD matrix to solve the iSCD problem (IV-A1). As threshold we use the 0.7 percentile of all similarity values yielded by the SCDs in the training data.

`BERT Classify`: This scenario uses the sequence classification use-case of BERT to solve the iSCD problem (IV-A2a).

`BERT Next`: This scenario uses the next sentence prediction use-case of BERT to solve the iSCD problem (IV-A2b).

`Matrix MPS²CD`: This scenario uses an SCD matrix to solve the MPS²CD problem (IV-B1).

`BERT Choose`: This scenario uses the multiple choice use-case of BERT to solve the MPS²CD problem (IV-B2a).

`BERT Highlight`: This scenario uses the question answering use-case of BERT to solve the MPS²CD problem (IV-B2b). The interval returned by BERT highlights the MPS²CD for a given sentence.

We do not need to specify more hyperparameters for the scenarios using the SCD matrix. In contrast, there are multiple hyperparameters to specify for BERT. We use a batch size of 40 during fine-tuning (10 for `BERT Choose`), because 40 samples fit into the 40GB of memory of the graphics card. The pre-trained model `bert-base-uncased` uses a dropout of 0.1 and cross-entropy loss to determine the model's error.

We run the fine-tuning for 3 epochs and use AdamW [13]. We test multiple values for the hyperparameters of AdamW and select the best, $\alpha = 5 \cdot 10^{-5}$ (also called learning rate), $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$ and $\lambda = 0.01$ (also called weight decay). The learning rate rises linear from 0 to α in the first 500 steps of the fine-tuning.

The accuracies in Figure 3 demonstrate that BERT is good at solving the iSCD problem. There is only a very small difference between `BERT Classify` and `BERT Next`. The small difference indicates that BERT does not benefit much when getting a pair of sentence and associated SCD simultaneously. The scenario using the SCD matrix reaches an accuracy of 0.61 only and thus `Matrix iSCD` is clearly worse than `BERT Classify` and `BERT Next`. In our scenario of the iSCD problem, the chance for a sentence being an SCD is 50%. Thus, an accuracy of 0.61 gained by `Matrix iSCD` comes close to random guessing. Also choosing different thresholds does not lead to better accuracies. Even though the authors of [4] got similar results, they identify sliding windows of words containing the SCDs. Thus, the authors solve a slightly different and a more difficult problem, meaning they can not randomly guess to reach an accuracy of 0.5.

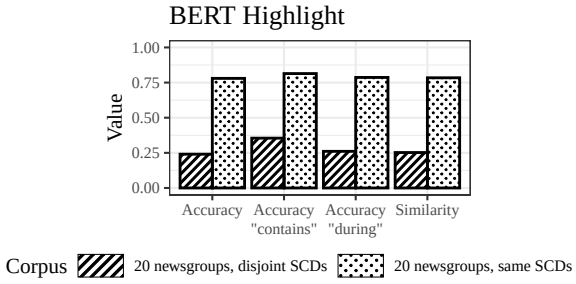

Figure 5. Accuracies, interval accuracies, and interval similarities gained from `BERT Highlight`.

For the MPS²CD problem, the scenarios using BERT and the SCD matrix result in similar values. Only `BERT Highlight` with a disjoint set of SCDs achieves a very low accuracy. As `BERT Highlight` asks to highlight the matching SCD out of four SCDs, the accuracy of 0.25 is as worse as randomly highlighting an SCD. Hence, we simplify the problem for `BERT Highlight` and do not split the set of SCDs. Using `BERT Highlight` with the same set of SCDs, then, shows a similar performance as the other two scenarios.

Besides the performance of all scenarios, also the runtime and the computational resources needed for training are relevant. In Figure 4, the duration for training each of the models is shown with a logarithmic scale. The training time of an SCD matrix is always similar and very fast in contrast to the fine-tuning of BERT.

Along with Figure 3 we only have considered the accuracy for `BERT Highlight`, but `BERT Highlight` returns intervals and we have already defined an interval similarity and two more accuracies based on Allen's interval algebra. In Figure 5 all four measures for `BERT Highlight` are shown, again distinguished by using the same set or a disjoint set of SCDs.

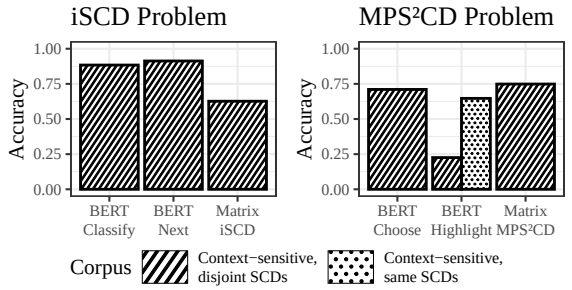

Figure 6. Accuracies of the *context-sensitive model* using the 20 newsgroups and the Manuscript cultures dataset.

The accuracy, accuracy *during* and interval similarity yield similar values. The accuracy *contains* yields slightly higher values, i.e., `BERT Highlight` returns occasionally a larger interval containing the correct SCD.

Overall `BERT Choose` and `Matrix` MPS²CD show the best performance solving the MPS²CD problem and `BERT Next` yields the best accuracy for the iSCD problem.

E. Context-Specific Subjective Content Descriptions

So far we only have a single context, the 20 newsgroups dataset and the Wiktionary annotation agent. However, as proposed in IV-C, we can use the same scenarios from the previous subsection to train a *context-sensitive model* representing different corpora with different contexts. We use a combined corpus of two different corpora with two different annotation agents, where both pairs of corpus and agent represent a different context.

The accuracies of the *context-sensitive model* in Figure 6 are very similar to the accuracies of the single-context model in Figure 3. The *context-sensitive model* reaches slightly smaller values solving the MPS²CD problem, while the relations between the accuracies of the scenarios remain the same. Surprisingly, solving the iSCD problem, the values reached by the *context-sensitive model* are slightly better.

The *context-sensitive model* shows the potential of BERT and the SCD matrix for estimating SCDs. Although, the model takes a second objective, i.e., detecting the context, the overall performance of the model is minimally reduced at most.

VI. Conclusion

This paper introduces approaches using BERT to solve two problems from the field of SCDs, namely the iSCD and MPS²CD problem. In an extensive evaluation, two introduced approaches solving each problem are compared to prior approaches for each problem. Summarized, the evaluation shows that BERT can be fined-tuned well to represent SCDs for text documents. On the iSCD problem, BERT performs better than the prior approach using the SCD matrix. On the MPS²CD problem, the prior approach using the SCD matrix performs slightly better.

Furthermore, the evaluation shows that the SCD matrix and BERT provide enough capacity to additionally learn the context of corpora and yield context-specific results. The proposed *context-sensitive model* uses the same approaches introduced and only differs in the corpus used for training. The model thereby detects the context of a sentence before estimating the SCD depending on the detected context.

We conclude that TLMs such as BERT are able to grasp the concept of SCDs, in a way that TLMs can be trained to solve SCD-related tasks.

As future work, it would be interesting to rerun the evaluation using other TLMs. DistilBERT [14] is a distilled version of BERT. Distillation is a mechanism to reduce the size of a model by trying to imitate a larger model focusing on the model's task. Distilled TLMs often provide better results and use less computational resources. Longformer [15] has no limit on the length of the input sequence and would allow to run `BERT Highlight` with more than four SCDs to select from.

Acknowledgment

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

References

[1] 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

[2] 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.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[4] M. Bender, T. Braun, M. Gehrke, F. Kuhr, R. Möller, and S. Schiff, "Identifying subjective content descriptions among text," *Proceedings of the 15th IEEE International Conference on Semantic Computing (ICSC-21)*, 2021.

[5] G. V. Maverick, "Computational analysis of present-day american english. henry kučera, w. nelson francis," *International Journal of American Linguistics*, vol. 35, no. 1, pp. 71–75, 1969.

[6] M. Sabou, K. Bontcheva, L. Derczynski, and A. Scharl, "Corpus annotation through crowdsourcing: Towards best practice guidelines," *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC'14)*, 01 2014.

[7] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:160025533

[8] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "Swag: A large-scale adversarial dataset for grounded commonsense inference," 2018. [Online]. Available: https://arxiv.org/abs/1808.05326

[9] S. Goel, R. Madhok, and S. Garg, "Proposing contextually relevant quotes for images," *Advances in Information Retrieval*, 2018.

[10] S. Vijayarani, J. Ilamathi, and S. Nithya, "Preprocessing techniques for text mining - an overview," *International Journal of Computer Science & Communication Networks*, vol. 5, p. 7–16, 2015.

[11] S. Kabir, C. Wagner, T. C. Havens, D. T. Anderson, and U. Aickelin, "Novel similarity measure for interval-valued data based on overlapping ratio," *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–6, 2017.

[12] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, p. 832–843, Nov. 1983.

[13] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019. [Online]. Available: https://arxiv.org/abs/1711.05101

[14] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter," *CoRR*, 2019.

[15] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *CoRR*, 2020.