Prevent Low-Quality Analytics by Automatic Selection of the Best-Fitting Training Data

Cornelia Kiefer
GSaME
University of Stuttgart, Germany
cornelia.kiefer@gsame.uni-stuttgart.de

Peter Reimann
GSaME
University of Stuttgart, Germany

Bernhard Mitschang
IPVS
University of Stuttgart, Germany

Abstract

Data analysis pipelines consist of a sequence of various analysis tools. Most of these tools are based on supervised machine learning techniques and thus rely on labeled training data. Selecting appropriate training data has a crucial impact on analytics quality. Yet, most of the times, domain experts who construct analysis pipelines neglect the task of selecting appropriate training data. They rely on default training data sets, e.g., since they do not know which other training data sets exist and what they are used for. Yet, default training data sets may be very different from the domain-specific input data that is to be analyzed, leading to low-quality results. Moreover, these input data sets are usually unlabeled. Thus, information on analytics quality is not measurable with evaluation metrics. Our contribution comprises a method that (1) indicates the expected quality to the domain expert while constructing the analysis pipeline, without need for labels and (2) automatically selects the best-fitting training data. It is based on a measurement of the similarity between input and training data. In our evaluation, we consider the part-of-speech tagger tool and show that Latent Semantic Analysis (LSA) and Cosine Similarity are suited as indicators for the quality of analysis results and as basis for an automatic selection of the best-fitting training data.

1. Introduction

In difference to data scientists, who oftentimes write program code, domain experts use simplified analysis toolkits such as RapidMiner [1] to construct analysis pipelines from scratch. In an analysis pipeline, various analysis tools are applied to the domain-specific data consecutively. For example, an analysis pipeline for the extraction of opinions from texts may consist of a first analysis tool that annotates where a least meaningful text unit such as a word begins and where it ends ('tokenizer'). A subsequent analysis tool in this pipeline then adds information on the part of speech of these tokens such as adjective, noun or verb ('part-of-speech tagger'). Afterwards, a third analysis tool adds information on opinions such as positive, negative and neutral ('sentiment analysis tool').

Most of the times, the domain expert who constructs such an analysis pipeline does not know for each step in the analysis pipeline which training data sets are available and what these training data may be used for. Thus, in many domain-specific analysis pipelines, 'out-of-the-box' tools with default training data sets are used. For example, news texts are used as default training data for the tokenizer and part-of-speech tagger analysis tools mentioned above. However, these default training data sets are very different from the operational input data sets of real industry and domain-specific use cases. Thus, **the use of default training data sets can lead to low-quality analytics results**.

Of course, it would be optimal to compile a new labeled data set for each analysis tool and domain-specific input data set. Yet, preparing labeled training data sets is very time-consuming, expensive and demands expert knowledge (e.g., see Ide et al. [2]). Therefore, it is not realistic to have a 'perfect' training data set for each analysis tool and domain-specific input data set and oftentimes default training data sets are used instead.

Moreover, the domain expert has no information on the impact of selected training data on the quality of analytics results. Traditional evaluation metrics for supervised analysis tools such as accuracy rely on labeled domain-specific input data [3]. Since labels are oftentimes not available for domain-specific data, information on analytics quality is not measurable with evaluation metrics. Thus, no information on the quality of analysis results is available to the domain expert who constructs a domain-specific analysis pipeline. So, the domain expert usually cannot verify and thus cannot know, that the employment of default training data may lead to low-quality analytics results.

In this paper, we suggest a method which adresses

URI: https://hdl.handle.net/10125/63868
978-0-9981331-3-3

this quality issue arising from non-fitting default training data. In contrast to analytics as carried out by data scientists, the methods only apply to analytics carried out with simplified analysis toolkits such as Rapid Miner and by domain experts who are not IT or data analytics experts. Besides the focus on domain experts, our method is crucial to many use cases, since domain experts and 'citizen scientists' make up the majority of the users of simplified analysis toolkits. We furthermore assume that the domain expert who constructs an analysis pipeline based on simplified analysis tools neglects the initial task of selecting the best fitting training data available. The major reason is that s/he often does not know which training data sets exist and what they are used for.

Our main contributions comprise (1) a concept for an **assessment of the expected quality of analytics results** via a measurement of the similarity between operational input data and training data. To this end, we suggest a quality indicator based on similarity metrics. Depending on the data type, similarity metrics for images, speech, structured data or texts may be employed (see Section 3). Moreover, based on these similarity measurements and based on a repository of available training data sets, we suggest a method for (2) automatic **selection of the best-fitting training data** for a given analysis tool and operational input data set. Then, we present (3) a **prototypical implementation** of these concepts. Our method is exemplary integrated into the FlexMash toolkit [4], which is a data processing and analysis toolkit suited for domain experts with little IT skills. In future, it may be easily integrated into other simplified analysis toolkits such as RapidMiner. Finally, we present an (4) **evaluation** of the concepts for textual data and the part-of-speech tagger analysis tool. We examine several text similarity metrics and find that the accuracy of part-of-speech taggers can be crucially increased by an automatic selection of the best-fitting training data when compared to the accuracy of 'out-of-the-box' tools which employ default training data.

First we motivate our method with an use case scenario (Section 2). In Section 3, we outline related work, before we present our concept in Section 4. Then, we describe evaluation results for textual data sets and the part-of-speech tagger (POS-tagger) analysis module in Section 5. Finally, we conclude our work (Section 6).

2. Use Case

As a sample use case, we consider a team leader in industry who is responsible for a production line. He wants to get information on frequent reasons for downtimes on the production line. To this end he has access to a data set with a free text field on causes and actions related to machine downtimes as well as to a simplified data analysis toolkit such as RapidMiner [1] or FlexMash [4]. He starts to build an analysis pipeline from scratch. In doing so, he employs many simplified 'out-of-the-box' analysis tools which rely on default training data sets such as news texts.

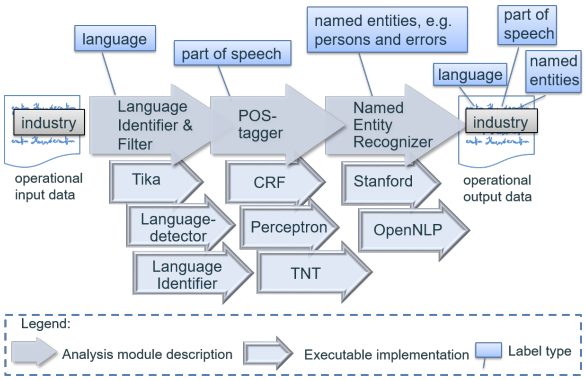

Figure 1. Sample analysis pipeline for textual industry data with concrete examples for 'out-of-the-box' analysis tools such as Tika

In Figure 1, we illustrate a sample analysis pipeline build by the domain expert for his use case. The *industry* data with free-text information on causes and actions related to machine downtimes is the operational input data to the analysis pipeline. The texts are analyzed by consecutive analysis modules. The domain expert decides to annotate the language of each text in a first step by the *Language Identifier*. He filters the data and only uses free texts with the language annotation 'German' in the next steps. Then a part of speech such as verb, noun or adjective is assigned to each word by the *POS-tagger*. Finally, named entities such as persons, errors and machines are recognized automatically by the *Named Entity Recognizer* (NER).

For each step in the pipeline, various 'out-of-the-box' implementations and tools exist. For example, Tika [5], Language-detector [6] and Language Identifier [7] can be employed in automatic identification of the language. Various out-of-the-box implementations for the POS-tagger module exist, e.g., CRF [8], Perceptron [8] and TNT [9]. Tools for NER exist in the Stanford tool collection [10] as well as in OpenNLP [11]. To our domain expert moreover internal pre-trained models for NER are available. The domain expert applies the analysis tools 'out-of-the-box' with default settings and default training data.

Thus, without the method suggested in this paper, the domain expert employs default and oftentimes non-fitting training data leading to low-quality results,

without even knowing about the issue. For example, non-fitting news texts are employed as training data by Tika (Language Identifier) and CRF (POS-tagger). We suggest a method to prevent low-quality analytics resulting from non-fitting training data by measuring automatically how good the training data (e.g., news) and operational data set (e.g., industry data) fit together and by automatically selecting the best-fitting training data set available.

3. Related Work

We suggest to employ the similarity of operational and training data as quality indicator. Wang and Strong [12] define data quality as "fitness for use by data consumers". We extend this definition and state that data also needs to be fit for use by analytical processing tools (cf. Kiefer [13]). Many data quality frameworks and quality indicators for structured data exist (e.g., Sebastian-Coleman [14]). However, data quality frameworks and quality indicators for unstructured data are demanded, but only first conceptual indicators are suggested by Batini and Sonntag [15, 16]. Executable quality indicators for unstructured data are missing. In Section 5.5, we investigate if text similarity correlates with the expected accuracy, and thus if text similarity may serve as a new quality indicator for textual data sets.

Our approach is based on similarity metrics. Several such metrics exist that are applicable to various types of data, such as database tables, images, videos and text (see Shirkhorshidi et al. [17], Mielke [18], Fuentes et al. [19] and Section 5.4). In our evaluation, we focus on textual data and thus employ text similarity metrics. Text similarity metrics play a crucial role in many research areas. For example, they are employed in automatic plagiarism detection [20] and automated assessment of student exams [21]. To the best of our knowledge, no previous work employs such similarity metrics as quality indicator based on the automatic measurement of the similarity between input and training data in analysis pipelines.

Moreover, we suggest to employ the similarity between operational and training data for the automatic selection of the best-fitting training data. In the machine learning community, automatically creating, improving and enriching training data for a specific analysis purpose, is a heavily discussed topic. Many works termed as 'training data selection' try to reduce the size of the training data without (drastically) affecting the quality in a negative way. For example, Wang et al. present an approach for support vector machines [22]. They show that, while they reduce the size of the training data set, accuracy does not deteriorate. Work on 'instance selection' tries to compile perfect training data with respect to runtime and accuracy on an instance level [23]. Traditional instance selection methods rely on labeled operational data, which in our setting is not available. Moore and Lewis select training data on an instance level for building a language model [24]. Here, the selection is based on a pre-compiled in-domain model. Several approaches moreover extend labeled training data sets by means of unlabeled data. Axelrod et al. adapt training data sets for statistical machine translation systems to new domains [25]. Li et al. adress semi-supervised text classification [26]. Blum et al. suggest a co-training approach combining labeled and unlabeled data for the classification of web pages [27]. In 'transfer learning', a predictive function is learnt from training data of one domain and then adjusted to a new domain by transferring knowledge from the source domain to the new domain [28]. All of these approaches are different from our method, since they assume that the initial training data set was already chosen or is composed, e.g., by means of 'instance selection'. They build up on initial training data or mining models, e.g., in terms of improving performance or coverage. The initial step of selecting training data is often neglected by domain experts. In difference to existing works, our approach adresses the first task of analysts of selecting the best training data available. In contrast to working upon already selected training data, we focus on impeding failures by domain experts who employ out-of-the-box analysis tools to build analysis pipelines from scratch and neglect the initial training data selection task.

4. Method to Prevent Low-Quality Domain-Specific Analytics

We start this section with a formalization of analysis pipelines, since our concept is to be applied to all elements within such pipelines based on supervised machine learning tools (Section 4.1). Then we go on by describing a training data and mining model repository, which builds the basis for our concept (Section 4.2). Finally, we describe our concept in Sections 4.3 and 4.4.

4.1. Data Analysis Pipelines

In this section, we give a formalization of analysis pipelines that are based on supervised analysis modules. In Figure 2, we present a sample formalization of an excerpt of an analysis pipeline. Concrete examples for these formalizations are given in Figure 1. The operational data o is read and then analyzed by consecutive analysis modules m_k. Each analysis module has a corresponding description, annotation

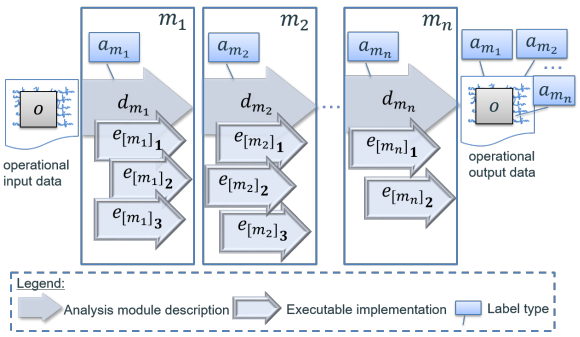

Figure 2. Sample formalized excerpt of an analysis process based on supervised machine learning modules.

type and several executable implementations. To indicate the affiliation of pipeline elements to the analysis module we add a subscript m_k to each corresponding element. The module description d_{mk} is the name of the analysis module m_k, which we add for easy comprehensibility. For example, module descriptions such as 'Optical Character Recognition' (OCR), 'speech-to-text', 'POS-tagger' and 'Named Entity Recognizer' exist. A module m_k adds labels of type a_{mk} to the operational data, where a_{mk} denotes one label type. A Named Entity Recognizer for example adds labels of type named entity. Further examples of label types are 'character', 'language', 'part of speech' and 'named entity'. Moreover, several executable implementations / analysis tools $e_{[m_k]_g}$ are possible for each module m_k. For example, various concrete implementations of POS-taggers exist (e.g., see section 5.1). The result of the analysis pipeline is the operational data set now enriched with annotations. Our concept is applied to each supervised-learning-based step separately, e.g., in an analysis pipeline as illustrated in Figure 2. Hence it is furthermore applicable to more complex, also non-sequential workflows.

4.2. Training Data and Mining Model Repository

To enable access to the raw training data sets, we introduce the concept of a **training data repository**. The training data sets are labeled with different label types (see Section 4.1). Several training data sets may be available for the same label type, and all of these may be stored in the repository. For example, several training data sets with annotations of type 'part of speech' are available, which are suitable training data sets for a POS-tagger module (see Section 5.1).

Fast processing could furthermore be ensured by linking pre-trained mining models to the training data sets stored in a **mining model repository**. This repository would need to contain the training data sets, information on the label types and available mining models. A similar repository was already suggested in a patent by Sas Institute Inc [29] and may serve as a more detailed conceptual basis.

4.3. Measure the Similarity Between Input and Training Data: Fit of Training Data as Quality Indicator (FiT)

Operational input data sets from real use cases do not come with labels for text analysis modules. For these data sets, no accuracy values are calculable. Thus, the quality of many analysis steps in domain-specific text analysis pipelines is usually not known. For example, if a domain expert analyzes an industry data set without labels, he cannot determine the accuracy of a OCR, speech-to-text or POS-tagger module. Nevertheless, if the similarity metric and the accuracy correlate, the similarity metric can be used as a quality indicator that informs the data analyst on how well the operational data and the used training data fit together. This goes along with the definition of data quality as "fitness for use by data consumers" in Wang and Strong [12]. Pointing at this definition, we denote the "similarity between input and training data" by "fit of training data" and suggest it as a new indicator for the quality of data within analytics pipelines. In Section 5 we discuss evaluation results, which show that this indicator correlates positively with accuracy.

The domain expert works with an analysis toolkit such as RapidMiner [1]. Here, the domain expert can build analysis pipelines in a graphical user interface. Therein, each analysis module is represented graphically, e.g. by a carat or a circle. This graphic representation may be highlighted with a green or red colour, thus indicating high or low quality. With our approach, this quality judgement may be based on the similarity of the operational and the training data. Each concrete analysis tool $e_{[m_k]_g}$ comes with metainformation on the training data set it employs as default. This default training data set can be retrieved from the repository described in Section 4.2. Then it's similarity to the operational data set may be measured. Since similarity and accuracy correlate positively (see Section 5.5), the module may moreover be highlighted, e.g., with colours corresponding to quality levels. Thus, the analyst is informed and moreover may react directly on quality issues, e.g. by changing the analysis tool or training data set employed. These steps may be performed for each (supervised learning) module in the analysis pipeline (1) before they are actually carried

out and (2) without need for labels in the operational data. This impedes failures and moreover saves time and resources in the construction of domain-specific analysis pipelines.

4.4. Automatic Selection of the Best-Fitting Training Data (SeT)

Besides giving feedback on the quality, the best-fitting training data available may be automatically selected from the training data repository, to improve analytics quality. Here, various training data sets may be available with the same label type. These may be retrieved from the repository. We denote the result with $R_{T_{m_k}}$. We moreover denote a training data set which is in a certain set of available training data sets $R_{T_{m_k}}$, i.e., which has label a_{m_k}, by $t_{[a_{m_k}]_i}$. Based on these notions, we illustrate the selection method for a sample module m_2 in Figure 3. In the first step, all training data sets with the appropriate label for the module are selected from the training data repository, i.e., $R_{T_{m_2}}$ is calculated (see step(1) in Figure 3). In the example illustrated in Figure 3, six possible training data sets exist which have labels of type a_{m_2} and thus could be used by m_2. Then, the similarity is calculated for each pair of o and $t_{[a_{m_2}]_i}$ (see step(2) in Figure 3). In the last step, the training data set $t_{[a_{m_2}]_i}$ with the highest similarity to o is chosen. In the example given in Figure 3, $t_{[a_{m_2}]_6}$ has the highest similiarity 0.65 and is thus selected as the training data set.

In Formula 1, we show the selection function. For each training data in the set of all available training data sets $R_{T_{m_k}}$, the score function $sim(t, o)$ is calculated, i.e., the similarity metric sim is applied to the training data and the operational data. Then, the t_i which maximizes this score function is selected.

$$\underset{t \in R_{T_{m_k}}}{\arg\max}\, sim(t, o) \tag{1}$$

After the most similar training data set is selected from the set of available training data sets, the concrete implementation $e_{[m_k]_g}$ maybe needs to be re-executed using the selected training data set. Based on a mining model repository (see Section 4.2), instead of generating a new mining model, which might take too long, analysis pipelines can be directly equipped with the corresponding pre-compiled mining models.

5. Evaluation of the Concept for Textual Data and the POS-Tagger Module

While the general concept is applicable to various data types, in the remainder of this work we focus on textual data. For textual data it is especially hard to create perfectly fitting training data (see Ide et al. [2]) and thus text analysis pipelines may benefit from selecting the most similar training data set available. We evaluate our concept for a text analysis module which is present in almost any text analysis pipeline: the POS-tagger.

We start this section with a description of the POS-tagger module. We continue by describing the data sets used. Then, we outline our prototypical implementation and describe the selection of relevant text similarity metrics. Finally, we detail the evaluation results.

5.1. A Text Analysis Module: the Part-of-Speech Tagger

A part-of-speech tagger (POS-tagger) automatically assigns a part of speech, such as *verb*, *adjective*, *noun*, to each word in a text. Consider the following sample sentence together with the part of speech tags as assigned by the NLTK POS-tagger (CRF): Can/*verb* we/*pronoun* automatically/*adverb* select/*verb* the/*article* optimal/*adjective* training/*noun* data/*noun* ?/*punctuation mark*

A POS-tagger module relies on training data. Several training data sets with manually assigned labels exist, see the next section. For the evaluation of our concept, we employ three high quality, feature-rich and heavily used implementations of the POS-tagger module:

- CRF POS-tagger [8]
- Perceptron tagger [30]
- TNT tagger [9, 31]

The evaluation scripts are available on GitHub, together with the prototype (see Section 5.3). The quality of a POS-tagger is determined by comparing the tags predicted by the system (by the tagger) with manually annotated labels. The standard metric for measuring the quality of a tagger is it's Token Accuracy (ACC, cf. Manning [32]). Token Accuracy in percent is given in Equation 2.

$$ACC = \frac{(\#\ correct\ POS\ tags\ in\ tagged\ data)}{(\#\ total\ POS\ tags\ in\ tagged\ data)} * 100 \tag{2}$$

5.2. Data Sets used in the Experiments

We conduct our experiments on **18 data sets from different domains**, such as news, prose, reviews,

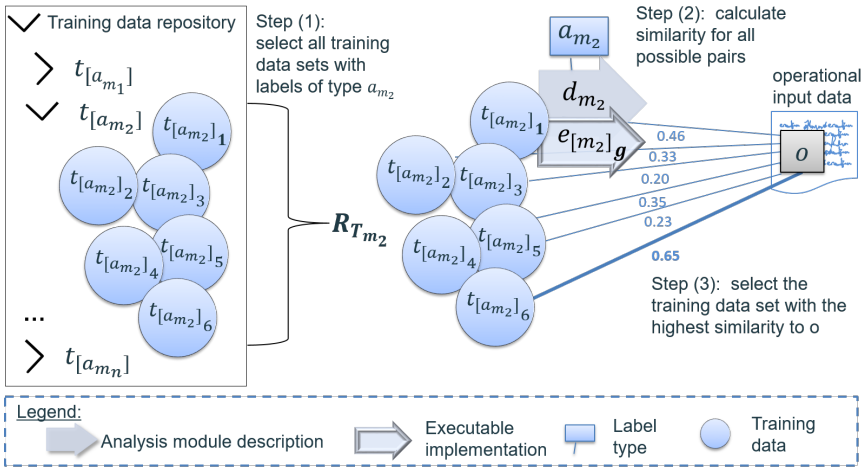

Figure 3. Selection of the training data with highest similarity to the operational data.

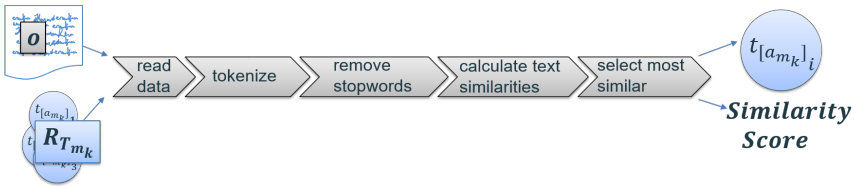

Figure 4. Processing pipeline for the prototypical implementation for SeT. For FiT the last processing step is not needed.

governmental texts, humorous texts, tweets and chat posts. 14 of the data sets are part of the huge 'Brown corpus collection' which contains 1.15M tokens and consists of data sets of different genres such as religion, mystery and reviews. Furthermore, a subset of the Penn Treebank (news), the CONLL 2000 data set (news), a Twitter corpus and a data set with chat posts is used in the experiments. The Twitter corpus was taken from Gimpel et al. [33], all other data sets are taken from the collection of corpora which comes with NLTK [34]. All of the data sets come with manually annotated part of speech tags and are freely available. Thus, in our evaluational setting, we are able to calculate the Token Accuracies and, furthermore, all evaluation results are reproducible.

5.3. Prototypical Implementation

In this section we describe a prototypical implementation of the concept explained in Section 4, which we developed for a first validation. In Figure 4, we illustrate the processing pipeline.

Several software frameworks implement text similarity measures. We chose the DKPro Similarity library [35], since it is open-source, actively developed and easy to use. It is part of DKPro, a collection of tools and programming libraries for Natural Language Processing developed at the technical university in Darmstadt, Germany. DKPro Similarity is based on the Apache UIMA framework [36] which ensures easy extensibility and scalability. We provide a standalone prototypical implementation on GitHub [37]. At first, the prototype **reads data** from two different folders in the file system in txt format. One folder contains the operational data set, e.g., in our use case the industry data. The other folder contains all training data sets available, e.g., annotated news, tweets and chat data. Before measuring the text similarity, standard preprocessing is executed: the texts are **tokenized** (split into the least meaningful units such as words), **stopwords may be removed** based on a stopword list. We employ the standard stopword list in NLTK [38] and all remaining words may additionally be normalized to it's lemma (=base form of the word). We checked all evaluational results with and without stopword removement and lemmatization, but the results do not differ significantly. Then the resulting lists of tokens are compared using the **text similarity** metric, e.g. Cosine Similarity or LSA (see Section 5.4). In measuring the FiT (see Section 4.3), the similarity between the default training data set and the operational data set is returned. A low value, e.g., caused by non-fitting

news texts employed as training data for the analysis of operational industry data, is indicated to the domain expert. In SeT (see Section 4.4), for each possible pair of operational and training data, the text similarity is calculated, the training data sets are ordered by similarity and the **most similar training data set is selected**. The name of the best-fitting training corpus together with the score is returned as result. For example, instead of news training data a related industry data set or annotated tweets may be selected in the use case scenario in section 2. We moreover package our method as webservice and integrate it into the flexible and easy-to-use data processing and analytics toolkit FlexMash [4]. Analogously it may be integrated into further analysis toolkits such as RapidMiner. Thus, our method will be available to domain-experts who want to build analysis pipelines from scratch.

5.4. Similarity Metrics for Textual Data Sets

Various similarity metrics for textual data exist. They focus on different aspects such as string-based similarity or semantic similarity. Further metrics consider structural, phonetic or stylistic characteristics of the texts. For an overview, we refer the reader to Bär et al. [35]. Most text analysis modules employ features which are semantic and string-based. Thus, we focus on these two metric types and test 2-3 well-known concrete metrics for each type (see Table 1). For the task of quickly supporting the domain expert in the construction of analysis pipelines the selection must furthermore be fast. We test the performance of SeT (see Section 4.4) based on our prototypical implementation (see Section 5.3) with a PC with 64-bit system, Intel(R) Core(TM) i7-6600U, 2.60 GHz, 2 cores and 16 GB RAM. We calculate processing times for all 18 data sets (see Section 5.2) and the experimental settings as described in Section 5.6.

Two string-based metrics, GreedyStringTiling and Levenshtein are not applicable to big data sets with billions of characters. Time complexity is $O(n^3)$, hence they result in out-of-memory errors after considerably longer processing times of several 10-minutes. Yet, with processing times of 16.7, 22.7 and 47.1 seconds, Cosine Similarity, LSA and 'WordNGramJaccard' are applicable. Thus, for a first evaluation of our concept, we focus on the bold-faced metrics listed in Table 1.

5.5. Evaluation Results on the Automatic Measurement of the Fit of Training Data (FiT)

In this section, we examine if similarity is a suitable quality indicator. Therefore, we will evaluate how good $sim(t, o)$ correlates with $ACC(t, o, e)$, see Equation 3.

$$sim(t, o) \sim ACC(t, o, e) \tag{3}$$

We employ all 18 data sets (see Section 5.2) and compile all possible pairs of operational and training data, where operational and training data are not the same. Thus, we have 18*17=306 pairs of operational and training data. The values for all pairs build two data rows, one with the similarity metrics and one with the accuracies. For each pair of operational and training data, we check how good similarity and accuracy correlate (as defined in Equation 3).

Consider the following two concrete examples, where a domain expert wants to build a text analysis pipeline and immediately gets feedback on quality. In the *first example*, the domain expert wants to analyze newspaper texts such as the texts in the CoNLL data set (see Section 5.2) with out-of-the-box text analysis modules. The graphical representation of the module turns green, thus indicating high quality. Operational news text data such as the CoNLL data set and out-of-the-box training data such as the Treebank data set (news) are very similar (Cosine Similarity of 0.95). In the *second example*, the domain expert wants to analyze reviews with a text analysis module that was trained on chat data. The graphic representation of the module turns red, thus indicating low quality. The reason is that operational review data and chat training data are not similar (Cosine Similarity of 0.51).

In our experimental setting, we only employed data sets with manually annotated labels for part of speech. Thus, we can check if these quality judgements are valid by comparing them to the accuracy values. The accuracies may be calculated as shown in Equation 2. For the *first example*, Cosine Similarity is 0.95 and accuracy is 0.89. For the *second example*, Cosine Similarity is 0.51 and accuracy is 0.56. This indicates a high correlation between Cosine Similarity and accuracy for these two value pairs. Beside looking at single value pairs, the two complete data rows of similarity and accuracy values may be compared by calculating correlation metrics. We calculate spearman's rho, see Kaltenbach [43] using the implementation which is part of the scipy package in python [44]. It is applied to two data rows X and Y with n values: $X = x_1, ...x_n$ and $Y = y_1, ...y_n$. A positive value for spearman's rho implies that if x rises, y rises as well. A negative rho value means that x and y correlate in the opposite direction: when x rises, y falls.

In Table 2, we give spearman's rho together with the p-value for the similarity rows generated by LSA, Cosine and 'WordNGramJaccard' and the average accuracy of the executable implementations

Table 1. Relevant measures: Semantic and string-based text similarity measures

Type	Concrete metric	Time in seconds	Reference
Semantic	**Cosine Similarity**	16.7	[39]
	Latent Semantic Analysis (LSA)	22.7	[40]
String-based	**'WordNGramJaccard'**	47.1	[39]
	GreedyStringTiling	out-of-memory	[41]
	Levenshtein	out-of-memory	[42]

Table 2. Correlation of Text Similarity and Accuracy

Text Similarity Metric	Spearman's rho	p-value
LSA	$0,84$	$1,58e^{-83}$
Cosine Similarity	$0,80$	$1,52e^{-69}$
'WordNGramJaccard'	$0,75$	$4,20e^{-57}$

e, i.e. the POS-tagger (see Section 5.1). For detailed result tables, which compare the similarity metrics to the full distribution of training data sets and corresponding accuracy values, see the GitHub repository (see Section 5.3). Here we also list separate values for each executable implementation tested. While the POS-taggers have different overall accuracy levels, for all three taggers the accuracy values do correlate with text similarity.

The p-value represents the probability that a non-correlating system produces data sets X and Y with the specified spearman correlation (for details on the p-value see Fisher [45]). The very low p-values for all similarity metrics indicate that the calculated correlations are valid and that it is very unlikely that data with such correlations could be generated by chance. The 'WordNGramJaccard' metric and accuracy have a solid positive spearman correlation. The Cosine and LSA similarity metrics have a strong positive correlation with accuracy.

5.6. Evaluation Results on the Automatic Selection of Training Data (SeT)

In Section 4.4, we suggested to automatically select the best-fitting training data within analysis pipelines built by domain experts. Here, we present first promising evaluation results with respect to this task. To this end we again employ all 18 data sets (see Section 5.2) and compile 18 experimental settings. Each experimental setting consists of an operational data set and a training data repository containing 17 disjunct available training data sets.

For the experiments, *for each* operational data set **o** and *for each* training data set **t** and *for each* text similarity metric **sim** and *for each* executable implementation **e** we perform the following steps:

- *step (1):* we calculate the text similarity $sim(t, o)$
- *step (2):* we train POS-tagger implementation e with t and test it on o (i.e., we calculate its 'Token Accuracy', see Equation 2)

In Table 3, we report the gain in accuracy yielded by our method. To this end, we compare the accuracies we get with SeT with those for **default** training data and with the **worst** selection that could be made by the domain expert. We consider the accuracy values for all 3 taggers and all 3 similarity metrics and present the arithmetic means and the maximum values in Table 3. The full distribution of possible results across the different training data sets and taggers can be found at GitHub (see Section 5.3). The POS-tagger implementations used are listed in Section 5.1. While the POS-taggers have different overall accuracy levels, the accuracy of all three taggers is equally sensible with respect to employing default, non-fitting or well-fitting training data.

When compared with the worst selection which could be made, a selection based on text similarity improves accuracy by 26 %-points in average and by up to 44. When compared with the out-of-the-box versions of the taggers represented by the Treebank training data set most heavily used in out-of-the-box POS-tagger modules (see Marcus et al. [46]), improvements of 12 %-points in average and up to 27 were found. While default training data, as often employed in out-of-the-box modules, performs well compared to a bad selection of training data, our method crucially improves the quality of the POS-tagging text analysis modules in both cases.

A possible problem of machine learning models is that they overfit to the training data. This may lead to bad generalization and worse performance on new unseen test data. This overfitting problem is, e.g., adressed by a compilation of 'universal' training data sets. These are a better basis for machine learning models which need to be able to generalize well. In our method, though, we select a specific training data set for each specific input data set anew and do not want to be able to generalize. Nevertheless, in future experiments, special attention needs to be given to universal training

Table 3. Gain in Accuracy (ACC) in %-points with the suggested SeT method compared to a default and worst selection of training data

Accuracy Gain by SeT compared to applying a default training corpus		Accuracy Gain by SeT compared to the worst selection of training data that could be made	
arithmetic mean	maximum value	arithmetic mean	maximum value
12	27	26	44

data sets and whether specific training data sets as would be selected by our method excel universal training data in terms of accuracy or not.

6. Conclusion and Future Work

Domain experts without IT / data analytics background build analysis pipelines from scratch. In this paper, we provided a concept that prevents low-quality analytics within these analysis pipelines. This concept employs the similarity between input data and training data as quality indicator and automatically selects the best-fitting training data. Thus, it prevents the domain expert from employing non-fitting default training data or wrongly selected training data that lead to low accuracies. In the first part of this work, we presented our concept, which is based on a training data repository and similarity metrics. While the concept presented is applicable to various data types, we focused on unstructured textual data in a first prototypical implementation and evaluation. To this end, we made a choice of useful text similarity metrics and presented evaluation results for the POS-tagger module present in most text analysis pipelines. The results are very promising. First, we showed that the similarity metrics Cosine Similarity and LSA correlate positively with the evaluation metric 'Token Accuracy'. Thus, Cosine Similarity and LSA may be used as quality indicators. Finally, we showed evaluation results with respect to the automatic selection of best-fitting training data. The method leads to higher accuracies of POS-tagger tools without any additional effort for the domain expert.

In future work, we will consider additional analysis modules, conduct more experiments with respect to the automatic selection of training data and adress the implementation details for the training data (and model) repository. Moreover, methods such as 'instance selection' and 'transfer learning' (see Section 3) should be investigated and made available to domain experts in easy-to-use analytics toolkits. Finally, we will explore the potential to expand the concept to other application domains such as employing it within the teaching process.

Acknowledgments

The authors would like to thank the German Research Foundation (DFG) for funding the project in the Graduate School of Excellence advanced Manufacturing Engineering (GSaME). Moreover, we would like to thank Andreas Laukart for crucial implementation work and Pascal Hirmer, Manuel Fritz and Holger Schwarz for important comments.

References

[1] *RapidMiner*. Available at `https://rapidminer.com/`.

[2] N. Ide and J. Pustejovsky, *Handbook of Linguistic Annotation*. Springer Netherlands, 2017.

[3] M. Hossin and M. N. Sulaiman, "A Review on Evaluation Metrics for Data Classification Evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, pp. 01–11, 2015.

[4] P. Hirmer and M. Behringer, "FlexMash 2.0 - Flexible Modeling and Execution of Data Mashups," in *RMC*, 2016.

[5] *Apache Tika - a content analysis toolkit*. Available at `https://tika.apache.org/`.

[6] *Language-detector*. Available at `https://github.com/optimaize/language-detector`.

[7] *DKPro Core*. Available at `https://dkpro.github.io/dkpro-core/`.

[8] *Source code for nltk.tag.crf*. Available at `http://www.nltk.org/_modules/nltk/tag/crf.html`.

[9] *Source code for nltk.tag.tnt*. Available at `https://www.nltk.org/_modules/nltk/tag/tnt.html`.

[10] *Stanford CoreNLP – Natural language software*. Available at `https://stanfordnlp.github.io/CoreNLP/`.

[11] *Apache OpenNLP*. Available at `https://opennlp.apache.org/`.

[12] R. Y. Wang and D. M. Strong, "Beyond Accuracy: What Data Quality Means to Data Consumers," *J. Manage. Inf. Syst.*, pp. 5–33, 1996.

[13] C. Kiefer, "Assessing the Quality of Unstructured Data: An Initial Overview," in *Proceedings of the LWDA* (Ralf Krestel, Davide Mottin, and Emmanuel Müller, eds.), CEUR Workshop Proceedings, (Aachen), pp. 62–73, 2016.

[14] L. Sebastian-Coleman, *Measuring Data Quality for Ongoing Improvement: A Data Quality Assessment Framework*. Burlington: Elsevier Science, 2013.

[15] C. Batini and M. Scannapieco, *Data and Information Quality*. Cham: Springer International Publishing, 2016.

[16] D. Sonntag, "Assessing the Quality of Natural Language Text Data," in *GI Jahrestagung*, pp. 259–263, 2004.

[17] A. S. Shirkhorshidi, S. Aghabozorgi, and T. Y. Wah, "A comparison study on similarity and dissimilarity measures in clustering continuous data," *PloS one*, vol. 10 e0144059, no. 12, 2015.

[18] J. Mielke, "A phonetically based metric of sound similarity," *Lingua*, vol. 122, no. 2, pp. 145–163, 2012.

[19] D. Fuentes, R. Bardeli, J. A. Ortega, and L. Gonzalez-Abril, "A similarity measure between videos using alignment, graphical and speech features," *Expert Systems with Applications*, vol. 39, no. 11, pp. 10278–10282, 2012.

[20] O. Hourrane and E. H. Benlahmar, "Survey of plagiarism detection approaches and big data techniques related to plagiarism candidate retrieval," in *Proceedings of the 2Nd International Conference on Big Data, Cloud and Applications*, BDCA'17, (New York, NY, USA), pp. 15:1–15:6, ACM, 2017.

[21] R. Ziai, N. Ott, and D. Meurers, "Short answer assessment: Establishing links between research strands," in *Proceedings of the 7th Workshop on Innovative Use of NLP for Building Educational Applications (BEA7)*, (Montreal, Canada), Association for Computational Linguistics, 2012.

[22] J. Wang, P. Neskovic, and L. N. Cooper, "Training data selection for support vector machines," in *Advances in Natural Computation* (L. Wang, K. Chen, and Y. S. Ong, eds.), (Berlin, Heidelberg), pp. 554–564, Springer Berlin Heidelberg, 2005.

[23] J. Olvera-López, J. Ariel Carrasco-Ochoa, J. F. Martínez-Trinidad, and J. Kittler, "A Review of Instance Selection Methods," *Artif. Intell. Rev.*, vol. 34, pp. 133–143, 2010.

[24] R. C. Moore and W. Lewis, "Intelligent selection of language model training data," in *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, (Stroudsburg, PA, USA), pp. 220–224, Association for Computational Linguistics, 2010.

[25] A. Axelrod, X. He, and J. Gao, "Domain adaptation via pseudo in-domain data selection," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, (Stroudsburg, PA, USA), pp. 355–362, Association for Computational Linguistics, 2011.

[26] Y. Li and J. Ye, "Learning Adversarial Networks for Semi-Supervised Text Classification via Policy Gradient," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, (New York, NY, USA), pp. 1715–1723, ACM, 2018.

[27] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, (New York, NY, USA), pp. 92–100, ACM, 1998.

[28] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[29] R. C. Chengwen and S. C. Tideman, *Model repository*. Available at `https://patents.google.com/patent/US6920458B1/en`.

[30] *Source code for nltk.tag.perceptron*. Available at `http://www.nltk.org/_modules/nltk/tag/perceptron.html`.

[31] T. Brants, "TnT: A Statistical Part-of-speech Tagger," in *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, (Stroudsburg, PA, USA), pp. 224–231, Association for Computational Linguistics, 2000.

[32] C. D. Manning, "Part-of-speech tagging from 97% to 100%: Is it time for some linguistics?," in *Computational Linguistics and Intelligent Text Processing* (A. F. Gelbukh, ed.), (Berlin, Heidelberg), pp. 171–189, Springer Berlin Heidelberg, 2011.

[33] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith, "Part-of-speech Tagging for Twitter: Annotation, Features, and Experiments," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, (Stroudsburg, PA, USA), pp. 42–47, Association for Computational Linguistics, 2011.

[34] *NLTK Corpora*. Available at `http://www.nltk.org/nltk_data/`.

[35] Daniel Bär, Torsten Zesch, and Iryna Gurevych, "DKPro Similarity: An Open Source Framework for Text Similarity," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, (Stroudsburg, USA), pp. 121–126, 2013.

[36] *Apache UIMA*. Available at `https://uima.apache.org/`.

[37] *training-data-selection*. Available at `https://github.com/kieferca/training-data-selection`.

[38] *NLTK's list of english stopwords*. Available at `https://gist.github.com/sebleier/554280`.

[39] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York: Cambridge University Press, 2008.

[40] T. K. Landauer, P. W. Foltz, and D. Laham, "An Introduction to Latent Semantic Analysis," *Discourse Processes*, vol. 25, no. 2-3, pp. 259–284, 1998.

[41] M. Wise, "String similarity via greedy string tiling and running karp−rabin matching," *Unpublished Basser Department of Computer Science Report*, 1993.

[42] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

[43] H.-M. Kaltenbach, "Hypothesis testing," in *A Concise Guide to Statistics* (H.-M. Kaltenbach, ed.), pp. 53–75, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[44] *scipy.stats.spearmanr*. Available at `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.spearmanr.html`.

[45] R. A. Fisher, *The design of experiments*. Oxford, England: Oliver & Boyd, 1935.

[46] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a Large Annotated Corpus of English: The Penn Treebank," *Computational Linguistics - Special issue on using large corpora*, no. 2, pp. 313–330, 1993.