Highlighter: automatic highlighting of electronic learning documents

Highlighter: automatic highlighting of electronic learning documents

Elena Baralis, *Member, IEEE,* and Luca Cagliero *Member, IEEE*

Abstract—Electronic textual documents are among the most popular teaching content accessible through e-learning platforms. Teachers or learners with different levels of knowledge can access the platform and highlight portions of textual content which are deemed as particularly relevant. The highlighted documents can be shared with the learning community in support of oral lessons or individual learning. However, highlights are often incomplete or unsuitable for learners with different levels of knowledge. This paper addresses the problem of predicting new highlights of partly highlighted electronic learning documents. With the goal of enriching teaching content with additional features, text classification techniques are exploited to automatically analyze portions of documents enriched with manual highlights made by users with different levels of knowledge and to generate ad hoc prediction models. Then, the generated models are applied to the remaining content to suggest highlights. To improve the quality of the learning experience, learners may explore highlights generated by models tailored to different levels of knowledge. We tested the prediction system on real and benchmark documents highlighted by domain experts and we compared the performance of various classifiers in generating highlights. The achieved results demonstrated the high accuracy of the predictions and the applicability of the proposed approach to real teaching documents.

Index Terms—E-learning, Text mining, Classification.

1 Introduction

E-LEARNING platforms are complex systems aimed at efficiently supporting learning activities with the help of electronic devices (e.g. laptops, tablets, mobile phones). Compared to traditional approaches to learning, they simplify the interaction between teachers and learners [1], because they allow (i) sharing electronic teaching materials with multiple users, (ii) access video lectures and other teaching content through electronic devices (PCs, laptops, tablets, mobile phones), and (iii) exchanging feedbacks on practices, exercises, or theoretical lessons through dedicated communication channels. The most commonly shared electronic teaching materials are textual documents [2]. They encompass lecture notes, e-books, scientific articles, or technical reports. However, due to the ever increasing amount of electronic documents retrievable from heterogeneous sources, the manual inspection of these teaching materials may become practically unfeasible. Hence, there is a need for automated analytics solutions to analyze electronic teaching content and to automatically infer potentially useful information.

In this paper we address the issue of automatically generating document highlights. Highlights are graphical signs that are usually exploited to mark part of the textual content. For example, the most significant parts of the text can be underlined, colored, or circled. The importance of text highlights in learning activities has been confirmed by previous studies on educational psychology (e.g. [3]) and visual document analysis (e.g. [4]). The highlighted documents can be easily shared between teachers and learners through e-learning platforms [2]. However, the manual generation of text highlights is time-consuming, i.e., it cannot be applied to very large document collections without a significant human effort, and prone to errors for learners who have limited knowledge on the document subject. Automating the process of text highlighting requires generating advanced analytical models able to (i) capture the underlying correlations between textual contents and (ii) scale towards large document collections.

The contribution of this paper is twofold: (1) It proposes to use text classification techniques to automate the process of highlighting learning documents. (2) It considers the proficiency level of the highlighting users to drive the generation of new highlights.

Objective 1 - Highlight generation based on classification techniques. Given a set of partially highlighted learning documents we aim at automatically generating new highlights by applying classification techniques. Classifiers are established data mining algorithms which have found application in various application domains. Their applicability to textual data is established [5]. Starting from a set of manually highlighted sentences, we build an abstract model, called classifier, which incorporates all the salient information needed to automatically predict whether a sentence should be highlighted or not. Our approach is data-driven and (almost) language-independent, i.e., it does not rely on advanced language processing techniques. Specifically, we analyze the content of previously highlighted documents ranging over the same topic to study the correlations between the occurrence of terms (or sequences of terms) in sentences and the presence/absence of highlights. Such correlations will be exploited to predict new highlights. Our approach is applicable to homogenous documents (i.e., documents ranging over the same topic), because it relies on frequency-based text analyses. For the sake of simplicity,

• *E. Baralis and L. Cagliero are with Dipartimento di Automatica e Informatica, Politecnico di Torino, Torino, C.so Duca degli Abruzzi, 24 10129. E-mail: {name.surname}@polito.it*

hereafter we will assume that a sentence is highlighted if at least a portion of its textual content is highlighted. The extension of the proposed approach to documents highlighted at different granularity levels (e.g. at the levels of single words or of paragraphs) is straightforward and its results are discussed in Section 6.

To build the classifier we tested multiple strategies, among which Bayesian classifiers [6], decision trees [7], Support Vector Machines [8], rule-based [9], Neural Networks [9], and associative classifiers [10]. To characterize the sentences of the learning documents, the classifier considers the following features: (i) the occurrences of single terms (unigrams), (ii) the occurrence of sequences of terms (n-grams), and (iii) the level of knowledge of the user who highlighted the sentence (if available). We tested our approach on benchmark documents highlighted by domain experts, i.e., the Document Understanding Conference 2005 SCU-marked documents [11]. Specifically, we compared the performance of various classifiers in generating highlights. The classifiers achieved good accuracy values in predicting highlights.

Objective 2 - Highlight generation driven by the knowledge level of the highlighting users. The reliability and usability of text highlights strongly depend on the level of expertise of the highlighting users [12]. For example, thanks to their proficiency on the covered topic, expert users can produce more reliable highlights than beginners. However, in some cases, the highlights made by users with lower levels of knowledge can be useful for supporting learning activities as well. For example, they may cover background knowledge commonly disregarded by advanced readers. Learning platforms often allow users to specify their current knowledge level on specific topics. In some cases, this information is not explicitly available, but it can be either inferred from the user role (e.g. academic professor, student of a B.Sc.university-level course) or assessed using ad hoc evaluation strategies (e.g. [13]).

Our aim is to exploit the information about the level of knowledge of the highlighting users during highlight generation and exploration. Since users with the same knowledge level are most likely to highlight the same parts of the text [12], we learn one classification model per level. Each model captures the underlying correlations hidden in the text highlighted by users with the same level. Hence, per-level models generate highlights tailored to different levels of knowledge. To improve the quality of the learning experience, learners may perform a per-level exploration of the newly generated highlights by adapting the level of exploration to their needs. The applicability of the proposed approach was validated on real teaching materials provided to the students of a B.Sc. university-level course.

This paper is organized as follows. Section 2 compares the proposed work with state-of-the-art related approaches. Section 3 introduces the preliminary concepts and the notation used throughout the paper. Sections 4 and 5 thoroughly describe the proposed approach and its applicability to related issues, respectively. Section 6 presents the results of the experimental evaluation conducted on benchmark and real documents. Finally, Section 7 draws conclusions and presents future developments of this work.

2 Related work

Some efforts to automatically generate highlights of generic documents have already been made. For example, in [14]–[16] information highlighting facilities have been proposed to assist users in evaluating relevance of accessed documents. The accessed documents are identified by a search engine in response to a user query. The parts of the text that are deemed as worth highlighting are identified by matching salient keywords in contextual vocabularies. In [17] the authors addressed the complementary issue of automatically recording the marks applied to paper documents on their electronic originals. In this paper, highlight generation is data-driven and not driven by user-generated queries. This approach, unlike keyword driven ones, does not require any a priori knowledge on the learner's interests and is applicable to a broader set of users.

The main contribution of this work is in the area of **learning analytics**, which entails the measurement, collection, analysis, and reporting of data about learners and their contexts [18]. It combines different disciplines such as computer science, statistics, psychology, and pedagogy. A prominent branch of research, called educational data mining, concerns the application of data mining techniques to data generated from educational settings (e.g. universities) [19]. Learning analytics tools have different goals, among which (a) the analysis and prediction of students' performance (e.g. [20], [21]), (b) the improvement of the quality of the learning experience by offering personalized and/or subject-wise services (e.g. [22], [23]), and (c) the extraction of salient content from large teaching data and its exploitation through online or mobile platforms (e.g. [24], [25]). The system proposed in this paper falls into category (c). The tool proposed in [25] focuses on automatically answering to learners' questions by applying text summarization techniques, while in [24] summaries of textual documents are generated to improve the accessibility of the learning materials through mobile devices. Unlike [24], [25], the approach proposed in this paper is not query-driven and relies on text classification techniques rather than on summarization algorithms.

Text classification aims at defining an abstract model of a set of classes, called classifier, which is built from a set of labeled textual data, i.e., the training set. The classifier is then used to appropriately classify new textual data for which the class label is unknown. In our context, the training set consists of a set of document sentences manually labeled as *highlighted* or *non-highlighted* by teachers or learners with different levels of knowledge. The prediction task focuses on deciding whether a sentence belonging to a non-highlighted (portion of) document is worth being highlighted or not. Many text classifiers have been proposed in literature. Amongst others, Support Vector Machines (SVMs) (e.g. [8]) and Neural Networks (NNs) (e.g. [26]) are commonly the mostly used classification models, because they are able to perform fairly accurate predictions. Alternative solutions include Bayesian algorithms (e.g. [27] and Decision trees (e.g. [28]). A survey of text classification techniques is given in [5]. Some attempts to use existing classification algorithms in learning analytics have already been performed.

For example, in [21], [29] the authors focused on predicting the final outcomes of students in different contexts. The goal of this work is different. To the best of our knowledge, text classification techniques have never been used to predict text highlights of teaching documents.

Text summarization entails generating a concise summary of a collection of textual documents. Sentence-based summarizers are automated tools that generate a summary consisting of a selection of the most significant document sentences in the collection. Many summarization approaches have been proposed in literature. Depending on the strategy used to perform sentence selection, they can be classified as (i) *Clustering*-based approaches (e.g., [30]), if they exploit clustering algorithms to group similar sentences and then pick the most significant sentences within each group. (ii) *Graph*-based approaches (e.g., [31]), if they rely on graph indexing algorithms. (iii) *Optimization*-based strategies, if they exploit Singular Value Decomposition [32] or Integer Linear Programming [33], or similar strategies to select salient document sentences. (iv) *Itemset*-based approaches (e.g., [34]), if they exploit frequent itemsets, which represent sets of document terms of arbitrary length, to capture the underlying correlations among multiple terms. While the classification problem addressed by this paper is a prediction task based on past humanly generated predictions (i.e., the set of previously highlighted sentences), in the summarization problem the goal is to describe most salient document features without any a priori information. An experimental comparison between text summarizers and classification techniques in the context under analysis is given in Section 6.

3 PRELIMINARIES

We consider textual documents as they are the most common teaching materials accessible through e-learning platforms. For example, teachers usually share lecture notes, books, reports, articles, or scientific papers in electronic form with their students. These electronic documents can be easily explored through multiple devices, such as laptops, tablets, and smart-phones [1]. In this study, we will consider only the textual content of the documents, while disregarding references, figures, charts, and their corresponding subtitles. Users of e-learning platforms can be teachers or learners. For example, to share the material related to university-level courses, academic professors and teaching assistants can upload books and lecture notes prior to each lesson. Depending on their proficiency on the course topics, students can be categorized in discrete knowledge levels (e.g. *beginner*, *intermediate*, *expert*). The level of knowledge can be either inferred from the user role (e.g. professor, teaching assistant, 1st-year student of a B.Sc.university-level course) or it can be assessed using ad hoc evaluation strategies (e.g. [13]). For the sake of simplicity, we assume that each user of the platform has a unique knowledge level.

Users can enrich electronic documents with highlights. We denote as highlight any graphical sign (e.g. underline, circle) that is used to mark part of the textual content. For our purposes, hereafter we will disregard all the textual annotations made on the margin of the text. We will consider only the highlights of the original text, which are deemed as more reliable than textual annotations. In general, highlights can be associated with single words, combinations of words, sentences, or entire paragraphs. For the sake of simplicity, hereafter we will assume that one sentence is highlighted if at least part of its textual content is highlighted. The problem of applying the proposed approach at different text granularities is discussed in Section 5.

Sentences are labeled with class values which indicate whether the sentence is highlighted or not. Since multiple users can highlight the same sentence, a sentence may have multiple class values, each one referring to a distinct level (possibly assigned by several highlighting users with the same level). For each class value assigned to the sentence, the corresponding knowledge level is known.

Let $\mathbf{D}$ be the document collection under analysis. It consists of a set of documents $d_1, d_2, \ldots, d_n$. Each document can be modeled as a set of sentences (i.e., portions of text separated by periods, question marks, or exclamation marks). Let s_j^i be the j-th sentence of document $d_i \in \mathbf{D}$. Let U be the set of users and let L be a discrete set of levels of knowledge (e.g. *beginner*, *intermediate*, *expert*). Each user $u_k \in U$ has a unique knowledge level $l_k \in L$. Users manually evaluated the sentences in $\mathbf{D}$ to decide whether they should be highlighted or not. We denote as $c(s_j^i, l_k)$ the class value associated with sentence s_j^i by users with level l_k. It takes value *highlighted* if the sentence is highlighted, *non-highlighted* otherwise. The class value is missing for non-evaluated sentences.

We will denote as *training document collection* $\mathbf{D}_{tr}$ a collection of documents whose sentences have already been evaluated, i.e., every sentence has one or more class values, each one corresponding to a specific knowledge level. Furthermore, we will denote as *test document collection* $\mathbf{D}_{te}$ a collection of documents for which sentences have no evaluations (i.e., the corresponding class values are missing). For convenience, hereafter we will denote as training (test) sentence $s_{tr} \in \mathbf{D}_{tr}$ ($s_{te} \in \mathbf{D}_{te}$) an arbitrary sentence in the training (test) document collection.

Given a training document collection $\mathbf{D}_{tr}$, a knowledge level $l_k \in L$, and a test document sentence $s_{te} \in \mathbf{D}_{te}$, the problem addressed by this work is to determine the value of $c(s_{te}, l_k)$ based on $\mathbf{D}_{tr}$ and l_k. Note that we aim at generating per-level highlight predictions and not per-user predictions. Hence, we target the prediction to the subset of users with level l_k and not to a specific user. Therefore, predictions for a given sentence will be same for all the users having the same level, but it may change for users with different levels. If the levels of knowledge in the training data are unknown, the problem reduces to determining the class based on the content of the training document collection solely.

4 HIGHLIGHTER: THE PROPOSED APPROACH

HIGHLIGHTER is a new approach to automatically highlighting learning documents based on the previously highlighted content. Figure 1 depicts the main steps of the proposed approach.

The manually highlighted documents are first collected into a training dataset (see Section 4.1). Some established text processing steps are then applied to prepare the raw

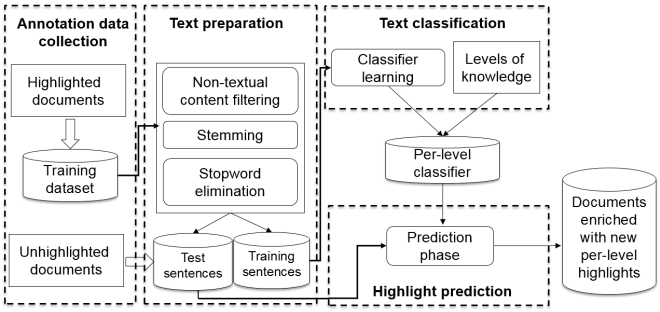


Fig. 1. The SKIll Predictor architecture.

data to the next classification process (see Section 4.2). Classification entails learning a model from the subset of document sentences that have been manually highlighted by human experts. The model is exploited to analyze new sentences of the collection and decide whether they are worth being highlighted or not based on their content and, possibly, based on the level of knowledge of the highlighting user (see Section 4.4). Finally, learners are provided with highlights corresponding to different levels of knowledge (see Section 4.5).

4.1 Data representation

For each sentence of the training and test document collections we consider the following attributes: (i) the textual content, (ii) the presence of highlights, and (iii) the level of knowledge of the user who highlighted the sentence (if any).

The training data consists of a set of records. Each record corresponds to a distinct pair $\langle s_j^i, l_k \rangle$, where l_k is the knowledge level of users who evaluated sentence $s_j^i \in \mathbf{D}$. For example, if the document collection contains 100 sentences, each one evaluated by users with three different levels, then the training dataset consists of 300 records. Note that some document sentences may be highlighted by users corresponding to a subset of the possible knowledge levels (i.e., some combinations of sentence and level may not occur). Each record stores the information about the highlights made by users with knowledge level l_k on sentence s_j^i. Specifically, we associate to each sentence s_j^i a class value $c(s_j^i, l_k)$ and the knowledge level l_k of user u_k. The class takes value *highlighted* if at least a portion of the sentence text is highlighted, *non-highlighted* otherwise. The level of knowledge indicates the proficiency level of the user who evaluated the sentence. The structure of the test dataset is analogous to those of the training dataset. However, the class value associated with each record is unknown. Table 1 shows some examples of training and test records associated with a text ranging over text classification.

4.2 Text preparation

To predict highlights from learning documents, the HIGHLIGHTER system considers the following features: (i) the occurrences of single terms (unigrams) in the sentence text, (ii) the occurrence of sequences of terms (n-grams), and (iii) the level of knowledge of the user who highlighted the sentence (if available). To properly handle textual features during sentence classification, few basic preparation steps are applied. First, non-textual content occurring in the text is automatically filtered out before running the learning process. Then, two established text processing steps are applied: (i) stemming and (ii) stopword elimination.

Stemming. Stemming entails reducing words to their base or root form (i.e., the stem) [35]. This step, which can be enabled or disabled according to the user's preferences, remaps the textual content to a reduced number of word roots. For example, nouns, verbs in gerund form, and past tenses (e.g. words *grants*, *granted*, *granting*) are re-conducted to a common root form (e.g. *grant*). This step is particularly useful for reducing the bias in classification when statistics-based text analyses are performed. For instance, different conjugations of the same verb, which rarely occur in the text, are transformed into their corresponding root form (i.e., the word stem), which is more likely to occur frequently.

Stopword elimination. Stopword elimination entails filtering out weakly informative words, i.e., the stopwords. Examples of stopwords are articles, prepositions, and conjunctions. In text analyses, these words are almost uninformative for predicting highlights and, thus, should be ignored. Furthermore, since they are likely to frequently occur in the analyzed data, their presence may bias the quality of statistics-based text analyses.

We apply the Wordnet stemming and stopwords algorithms for English-written documents. To cope with documents written in different languages, different stemming and stopword elimination algorithms can be straightforwardly integrated as well.

To analyze the occurrence of single terms in the sentence text, after stemming and stopword elimination the sentence text is transformed into a term frequency-inverse document frequency (tf-idf) matrix [36]. Tf-idf is an established term statistics, which is briefly introduced below.

The tf-idf matrix. The term frequency-inverse document frequency (tf-idf) evaluator [36] is commonly used to measure how important a word stem is in a text [35]. In our context, an arbitrary element ti_{qi} of the tf-idf matrix TI is defined as follows: $ti_{qi} = \frac{n_{qi}}{|s_i|} \cdot \log \frac{|\mathbf{S}|}{|\{s_i \in \mathbf{S} : w_q \in s_i\}|}$, where $\mathbf{S}$ is the set of sentences occurring in the training document collection $\mathbf{D}_{tr}$, n_{qi} is the number of occurrences of the q-th stem w_q in the i-th sentence $s_i \in S$, $|s_i|$ is the number of word stems that are contained in s_i, and $\frac{|\mathbf{S}|}{|\{s_i \in \mathbf{S} : w_q \in s_i\}|}$ represents the inverse document frequency of the stem w_q in the whole collection. The logarithm of the inverse document frequency is minimal when the inverse document frequency is equal to 1 (i.e., a term occurs in every sentence of the dataset) and thus the corresponding tf-idf value reduces to zero.

The key idea behind the tf-idf statistics is that word stems appearing frequently in few sentences (i.e., high local term frequency), but rarely in the whole collection (i.e., low document frequency), are the most discriminative ones in text classification.

Table 2 reports the tf-idf values associated with each word stem in the example dataset in Table 1. For example, word stems occurring in most sentences (e.g. *data*) have relatively low tf-idf values, whereas word stems occurring in few sentences (e.g. word stem *algorithm*) have relatively

TABLE 1
Examples of training and test records.

Sentence text	Level of knowledge	Class
TRAINING		
Text mining is related to machine learning and, to some extent, to artificial intelligence as well.	Expert	Yes
Classification and Decision Trees are prediction algorithms	Beginner	No
Support Vector Machines are commonly used on textual data	Intermediate	No
Textual data are inherently sparse, because words are likely to occur only in few sentences	Expert	Yes
Support Vector Machines were designed for coping with textual data	Beginner	Yes
TEST		
Support Vector Machines are among the most accurate classifiers on textual data	Beginner	**?**

TABLE 2
tf-idf matrix (transposed).

Word stem	s_1	s_2	s_3	s_4	s_5
Algorithm	0	0.447	0	0	0
Artifici	0.375	0	0	0	0
Classif	0	0.447	0	0	0
Common	0	0	0.716	0	0
Cope	0	0	0	0	0.582
Data	0	0	0.227	0.127	0.185
Decis	0	0.447	0	0	0
Design	0	0	0	0	0.582
Extent	0.375	0	0	0	0
Inher	0	0	0	0.402	0
Intellig	0.375	0	0	0	0
Learn	0.375	0	0	0	0
Like	0	0	0	0.402	0
Machin	0.119	0	0.227	0	0.185
Mine	0.375	0	0	0	0
Occur	0	0	0	0.402	0
Predict	0	0.447	0	0	0
Relat	0.375	0	0	0	0
Sentenc	0	0	0	0.402	0
Spars	0	0	0	0.402	0
Support	0	0	0.408	0	0.331
Text	0.375	0	0	0	0
Tree	0	0.447	0	0	0
Vector	0	0	0.408	0	0.331
Word	0	0	0	0.402	0

TABLE 3
Training dataset (extract).

Record	Algorithm	Artifici	Classif	...	Level	Class
s_1	0	0.375	0	...	Expert	Yes
s_2	0.447	0	0.447	...	Beginner	No
s_3	0	0	0	...	Intermediate	No
s_4	0	0	0	...	Expert	Yes
s_5	0	0	0	...	Beginner	Yes

high tf-idf values.

Starting from the tf-idf matrix associated with the document sentences, a training and a test datasets are generated. Both datasets have the same structure: a tf-idf matrix enriched with two new columns *level of knowledge* and *class* indicating, respectively, the level of the knowledge of the highlighting user and the information about the presence/absence of an highlight (highlighted/non-highlighted). For example, Table 3 reports an extract of the training dataset generated from the training collection and tf-idf matrix in Tables 1 and 2. A similar procedure is applied to the test records. In the latter case, the value of the class is unknown and represents the target of the prediction.

Combinations of terms, such as *data mining*, can provide additional information with respect to single terms *data* and *mining*. For this reason, we generate also an extended version of the tf-idf matrix in Table 2 including both unigrams and n-grams (n>1). In our context, n-grams are sequences of n word stems. As discussed in Section 6, using the extended tf-idf matrix allows achieving better performance than using only the standard one.

For example, since sentence with id 1 in Table 1 contains the sequence of words *text mining*, in the tf-idf matrix in the extended matrix version we added a column labeled as *text mine*, which indicates the occurrence/absence of the corresponding sequence of word stems within each document sentence. Note that, since the formula of tf-idf simply relies on frequency counts, its extension to the case of n-grams is straightforward [36].

4.3 Feature selection

To predict the class value of the test records, features in the training dataset may have different importance. Some of them are strongly correlated with the class and, thus, their presence is crucial to perform accurate predictions. Others are uncorrelated with the class. Hence, their presence could be harmful, in terms of both accuracy and efficiency of the classification process.

Feature selection is the task of selecting a smaller subset of dataset features that are worth considering in place of the whole feature set [35]. It is often applied prior to text classification to filter out the n-grams that have least importance or relatively weak correlation with the class [5]. To perform accurate highlight predictions we filter unigrams/n-grams based on their average tf-idf value in the training dataset. Tf-idf values are commonly used as quality measures of the importance of the terms in the document collections in both text categorization [5] and summarization [30]. To decide whether a sentence should be highlighted or not, we apply the classification algorithms to a filtered version of the dataset including the top-K unigrams/n-grams in order of decreasing average tf-idf value. The impact of parameter K on classifier performance is discussed in Section 6.2.

4.4 Text classification

Classification is a two-step process which entails: (i) Learning a model from the training dataset, called classifier, which considers the most significant correlations between the class and the other data features, and (ii) assigning a class value to each record in the test dataset, based on the previously generated model.

To investigate the use of text classification algorithms in highlight prediction, we learn multiple benchmark classifiers relying on different techniques.

4.5 Per-level document highlighting

If in the training dataset there is no information about the level of knowledge of the users, one single classification model is generated and used to predict new highlights. Otherwise, the knowledge level of the highlighting users is considered because it is deemed as relevant to perform accurate highlight predictions. The use of the knowledge level in the prediction process will yield the following advantages.

Tailoring highlights to users with different knowledge levels. Since users with the same level of knowledge are likely to highlight the same parts of the text [12], we partition training records according to the knowledge level of the highlighting user and we generate one classification model per level. Each model captures the underlying correlations hidden in the text highlighted by users with the same level. Hence, per-level models produce highlights tailored to different levels.

Enabling per-level highlight exploration. The reliability and usability of text highlights strongly depend on the level of expertise of the highlighting users [12]. For example, thanks to their proficiency on the covered topic, expert users can produce more reliable highlights than beginners. However, in some cases, the highlights made by users with lower levels of knowledge can be useful for supporting learning activities as well. For example, they may cover background knowledge commonly disregarded by domain experts. Our aim is to exploit the information about the level of knowledge of the highlighting users during highlight exploration. Specifically, to improve the quality of the learning experience, learners are provided with highlights corresponding to different levels of knowledge and may perform a per-level exploration of the newly generated highlights by adapting the level of exploration to their needs.

In Section 6 we validated the performance of both the unified and the per-level models generated from benchmark document collections.

5 Analysis of Document Highlights at Different Granularity Levels

The approach described in Section 4 relies on the assumption that documents are enriched with highlights at the sentence level. However, learning documents can be highlighted at different granularity levels. For example, users can highlight separate words within the same sentence, short sequences of words, or entire paragraphs consisting of a few sentences.

Since the methodology proposed in this paper to predict document highlights is general, it can be easily extended to documents enriched with highlights at different granularity levels. To effectively cope with documents highlighted at the level of paragraphs, sentences within the same paragraph are first merged together and then inserted into the training dataset. Conversely, when highlights are mapped to n-grams (i.e., sequences of n terms) each training record is identified by a distinct pair $\langle$*user*, *n-gram*$\rangle$ and the level of abstraction of the selected features depends on the least granularity level of the highlighted text. For example, while considering highlights of 3-grams (i.e., sequences of three words in a row), the textual features characterizing the textual content can be unigrams or bigrams.

In Section 4 we labeled a sentence as highlighted if at least a portion of its textual content is highlighted. This assumption entails considering some extra contents during the classification process with respect to those manually highlighted by the users. Considering this additional content will result in generating some extra features in the training dataset, which are probably somehow related to but do not necessarily correspond to the actually highlighted content. To overcome this issue, as described in Section 4.3, we apply a feature selection step before running the classification algorithm which selects only the most significant data features, thus improving the effectiveness and efficiency of the classification process. An experimental analysis of the impact of the feature selection step on benchmark data is reported in Section 6.

6 Experimental Results

We performed a large suite of experiments to analyze the accuracy of the highlight predictions (see Section 6.1), the impact of considering n-grams and of the feature selection step on the accuracy of the generated predictions (see Section 6.2), the impact of false positive/negative outcomes on classifier performance (see Section 6.3), the applicability of the proposed method on real teaching documents (see Section 6.4), the use of summarization techniques to address highlight prediction (see Section 6.5), and the training and prediction times (see Section 6.6).

All the experiments were performed on a quad-core 3.30 GHz Intel Xeon workstation with 16 GB of RAM, running Ubuntu Linux 12.04 LTS. To test the performance of different state-of-the-art text classifiers, we used the implementations available in the RapidMiner data mining and machine learning suite (version 5.3.015). Specifically, we considered the following classifiers belonging to different categories (indicated in italic face below).

- *Bayesian classifiers*: Naive Bayes [6]
- *Support Vector Machines*: The LibSVM classifier [8]
- *Decision trees*: The ID3 classifier [7]
- *Rule-based classifiers*: Ripper [9]
- *Neural Networks*: Auto Multi-Layer Perceptron (AutoMLP) [37]
- *Instance-based classifiers*: The K-Nearest Neighbor Classifier [38]
- *Associative classifiers*: The Live and Let Live (L^3) Classifier [10]
- *Logistic regression*: SparseLOGREG (SLOGREG) [39]

More details on the classification models are given in [40]. Since no associative classifier is integrated in RapidMiner, we separately tested the state-of-the-art L^3 classifier [10] by using the Weka plugin provided by the respective authors. For each classifier we tested several configuration settings, among those recommended by the respective authors, and we selected the best performing one over all the tested datasets. In the feature selection step, we set the top-K number of word stems to 250 as standard configuration. To compare the performance of text classifiers with that of text summarizers, we re-implemented the Association Mixture

Text Summarization (AMTS) algorithm [41] to the best of our understanding, and we used the implementation of the Integer Linear Programming-based ICSI summarization system (ICSIsumm) [42] provided by the respective authors. All the experiments were performed on a quad-core 3.30 GHz Intel Xeon workstation with 16 GB of RAM, running Ubuntu Linux 12.04 LTS.

6.1 Accuracy of the highlight predictions

To study the accuracy of the highlight predictions, we performed experiments on the Document Understanding Conference 2005 (DUC'05) SCU-marked collections [11].

Analyzed documents. The collection is a benchmark for textual document summarization, which consists of a set of news documents ranging over different topics. Based on the covered topics, documents are organized in collections. A group of manually written summaries of the document collections are edited by hand by real end users to produce a set of simple declarative phrases, hereafter denoted as Summary Content Units (SCUs), for each topic. Thanks to the effort of the university of Ottawa, SCUs were mapped to the original document sentences through the Pyramid evaluation system [43] to generate a set of SCU-enriched document collections, i.e., 27 document collections highlighted at the sentence level. SCUs correspond to sentence highlights, possibly enriched with textual comments and a weight, which indicates the pertinence of the highlight to the main topic of the document. For our purposes, we ignored the textual content associated with the SCUs and we considered the weight of a SCU as a measure of the level of knowledge of the user who generated the SCU. SCU weights are assumed to be a good indicator of the level of proficiency of the highlighting user, because expert users are more likely to highlight sentences pertinent to the topic under analysis than non-expert ones.

In the DUC'05 documents, more than 10% of the document sentences were enriched with SCUs. SCU weights range from 1 to 7. Approximately 35% have minimal weight (1), 30% of the SCUs have maximal weight (7), whereas the intermediate weights (from 2 to 6) have similar frequency counts and cover altogether approximately 35% of the SCUs. SCU weights are roughly distributed equally across all documents.

Accuracy definition. We evaluated classifier performance in terms of classification accuracy. Accuracy measures the ability of the classifier to correctly classify unlabeled records [35]. It is the ratio between the number of correctly classified data records and the number of tested records. In our context, it indicates the percentage of sentences that were correctly labeled as *Highlighted* or *Not highlighted* according to the model generated on the training data.

Validation procedure. Accuracy has been computed by using a 10-fold stratified cross validation test. Specifically, the training dataset is first randomly partitioned into 10 folds, each one including approximately the same number of sentences and having the same class label distribution (i.e., having approximately the same percentage of highlighted sentences). Then, one fold is considered as test dataset, whereas the remaining ones are used as training set. The classification process is validated by computing the per-fold accuracy value (i.e., the percentage of correctly classified records in the test dataset). Then, the test dataset is changed and the same procedure is iterated until all the ten per-fold accuracy values are computed. The resulting accuracy estimate is given by the sum of the per-fold accuracy values.

6.1.1 Results

Table 4 summarizes the accuracy results achieved by all the classifiers on the benchmark datasets without considering the levels of knowledge (i.e., by generating a unified model from all the manually highlighted sentences). We recall that approximately 10% of the sentences in the document collection were highlighted by at least one user. Table 5 reports the average accuracy results achieved by the per-level classification models (i.e., one classification model per level of knowledge). In the latter case, the detailed results per dataset were omitted for the sake of brevity. For each classifier we selected unigrams (single terms) and bigrams (sequences of 2 terms) as discriminative features and applied the feature selection step with K=250. In the tables we reported the results achieved by a reference configuration chosen by performing several tests with different settings and by choosing the one that achieved the best average performance over all datasets. The best parameter settings are specified below each classifier name. For each collection, the best classifier performance is written in boldface.

The Neural Network Auto Multi-Layer Perceptron (AutoMLP) performed best by considering both the unified model (approximately 87%) and six out of seven per-level models (between 88.5% and 90.7%). Hence, approximately 9 out of 10 highlight predictions were correct. Despite its simplicity, the Naive Bayes classifier achieved accurate values fairly close to AutoMLP. Similarly, The L^3 associative classifier and the SparseLOGREG algorithm achieved averagely high accuracy values, whereas the average accuracy of the predictions made by LibSVM, K-NN, ID3, and Ripper are fairly lower. Due to the inherently higher complexity of the problem, the accuracies achieved by the unified models are, on average, lower than those achieved by the per-level models. Among the per-level predictions, model predictions corresponding to higher levels of knowledge are slightly more accurate than lower-level ones. This is probably due to the higher sparsity of the textual data distribution at lower levels of knowledge. Anyway, the average accuracy performance is encouraging: over all the per-level models, the worst performing classifier correctly predicted, on average, more than 85 document highlights out of 100.

We also tuned the performance of the classifiers separately for each dataset. Specifically, we considered a potentially different configuration setting for each dataset. Based on the achieved results (not reported here for the sake of brevity) AutoMLP and L^3 have proven themselves to be again the most accurate classifiers (e.g. average accuracies for the unified model AutoMLP 88.58%, L^3 88.10%).

To validate the statistical significance of the accuracy improvements achieved by the AutoMLP classifier, we used the 10-fold cross-validated paired t-test. All tests were applied at significance level p = 0.05 (95%) on all the evaluated datasets. The performance of the AutoMLP classifier was compared with that of all the other classifiers, by considering both the reference and the tuned configuration

TABLE 4
Unified model. Comparison between state-of-the-art classifiers on average percentage accuracy. For each dataset, the best accuracy value is written in boldface and all significant worsening w.r.t. AutoMLP reference config. (t-test value above 0.05) are starred.

Collection (Num. sentences)	**Naive Bayes**	**L^3** s=1% c=50% Compact rules	**SLOGREG** kernel=polyn. ϵ=0.001	**AutoMLP** cycles=500 num. gener.=10	**LibSVM** kernel=sigmoid type=C-SVC ϵ=0.001	**k-NN** k=5 dist=mixed-Eucl.	**ID3** Crit.=Gain R. min gain=0.05	**Ripper** Crit.=I.G. ratio=0.9
d311i (1152)	87.52	**89.35**	87.87	88.65	88.22	79.84	88.66	89.00
d324e2 (168)	84.12	**85.11**	84.36	84.37	81.39*	63.04*	77.20*	80.63
d324e3 (256)	82.08*	**86.94**	83.92*	86.11	85.55	60.56*	82.70	83.71
d324e (331)	83.12	**84.41**	83.39	81.59	78.79*	63.69*	74.19*	79.56
d345j (463)	83.11	85.74	**85.90**	85.25	84.10	79.51	81.81	80.00*
d366i (220)	90.88*	92.20	92.09	**92.25**	92.03	92.14	90.88	90.66
d376e (356)	78.35	79.68*	81.06	81.06	77.37*	77.96	**81.81**	77.58*
d391h (508)	90.16	90.49	89.52*	**90.70**	85.03*	89.04	85.84*	89.04
d393f (822)	88.52	**88.93**	88.20	88.52	79.99*	86.20	86.93	85.99
d400b2 (303)	88.04	**88.48**	88.66	87.59	88.30	82.14	84.55	83.66
d400b (637)	88.44	87.89	**88.72**	88.17	89.27	84.77	85.87	85.05*
d407b2 (347)	85.95	86.22	**87.82**	87.44	86.88	83.04	85.84	84.63
d407b (633)	87.52	87.33	**88.67**	87.24	87.62	85.24	86.57	85.24
d413a (776)	80.97	**82.45**	82.16	81.71	79.02	76.17*	83.21	81.72
d422c (580)	81.33	78.40*	81.54*	**82.80**	80.71	76.15*	79.87*	76.15*
d426a2 (769)	90.12	91.31	**92.12**	90.04	89.60	84.10*	91.60	88.56
d426a (595)	90.20*	91.00*	**92.52**	**92.52**	90.13	86.87	91.51	91.43
d431h (556)	88.45	**89.71**	88.72	89.44	86.73	87.00	87.01	87.00
d435f (756)	87.64	88.42	**88.85**	87.82	88.16	85.65	85.04	87.81
d632i (926)	83.90*	86.39	85.80*	**88.04**	85.08*	81.54	78.70*	82.37
d633g2 (439)	85.65*	85.20*	85.65*	**86.21**	82.02*	83.17	80.00*	83.62
d633g (715)	87.14	87.15	**87.37**	87.15	84.45*	83.88	82.86*	84.33
d654f (547)	86.10	87.00	86.42*	**87.66**	84.12*	84.36	82.22*	85.683
d671g (701)	86.19	**86.51**	86.51	85.87	85.24	80.00*	85.87	85.87
d683j (682)	84.37	84.24	82.75*	**84.89**	82.75	72.83*	82.74	80.44*
d695c2 (742)	88.09	88.56	88.56	**88.75**	86.77	84.79	86.39	86.01
d695c (731)	87.68	88.24	**88.71**	88.25	86.94	84.98	87.13	86.10
Average and std dev.	86.13 $\pm$ 2.98	86.94 $\pm$ 1.92	86.96 $\pm$ 2.88	**86.99 $\pm$ 2.86**	85.05 $\pm$ 3.30	80.69 $\pm$ 1.72	84.33 $\pm$ 2.79	84.51 $\pm$ 1.79

TABLE 5
Per-level models. Comparison between state-of-the-art classifiers on average percentage accuracy and standard deviation.

Level of knowledge	**Naive Bayes**	**L^3** s=1% c=50% Compact rules	**SLOGREG** kernel=polyn. ϵ=0.001	**AutoMLP** cycles=500 num. gener.=10	**LibSVM** kernel=sigmoid type=C-SVC ϵ=0.001	**k-NN** k=5 dist=mixed-Eucl.	**ID3** Crit.=Gain Ratio min gain=0.05	**Ripper** Crit.=I.G. ratio=0.9
1 (min)	88.03 $\pm$ 2.90	88.23 $\pm$ 1.70	88.35 $\pm$ 2.83	**88.51 $\pm$ 2.76**	86.37 $\pm$ 0.88	85.82 $\pm$ 3.06	87.08 $\pm$ 2.23	86.37 $\pm$ 0.88
2	**89.96 $\pm$ 3.04**	89.08 $\pm$ 1.58	87.09 $\pm$ 3.18	88.43 $\pm$ 2.23	87.57 $\pm$ 1.12	85.06 $\pm$ 3.97	86.11 $\pm$ 2.35	87.57 $\pm$ 1.12
3	89.96 $\pm$ 2.44	**90.19 $\pm$ 1.29**	89.91 $\pm$ 2.25	**90.19 $\pm$ 1.94**	88.70 $\pm$ 0.78	88.12 $\pm$ 2.81	89.01 $\pm$ 2.02	88.70 $\pm$ 0.78
4	89.91 $\pm$ 2.29	90.17 $\pm$ 1.32	89.92 $\pm$ 2.35	**90.36 $\pm$ 1.84**	89.06 $\pm$ 0.85	88.41 $\pm$ 2.54	89.08 $\pm$ 1.93	89.06 $\pm$ 0.85
5	90.30 $\pm$ 2.12	90.28 $\pm$ 1.30	90.33 $\pm$ 2.06	**90.61 $\pm$ 1.58**	89.04 $\pm$ 2.63	89.97 $\pm$ 0.79	89.56 $\pm$ 1.71	89.97 $\pm$ 0.79
6	90.38 $\pm$ 2.28	90.53 $\pm$ 1.14	90.41 $\pm$ 2.22	**90.72 $\pm$ 1.84**	89.90 $\pm$ 0.80	88.71 $\pm$ 2.79	89.85 $\pm$ 1.84	89.90 $\pm$ 0.80
7 (max)	90.23 $\pm$ 2.23	**90.69 $\pm$ 1.26**	90.24 $\pm$ 2.27	**90.69 $\pm$ 1.85**	89.91 $\pm$ 0.86	88.64 $\pm$ 2.88	89.87 $\pm$ 1.82	89.91 $\pm$ 0.86

settings. For each comparison, Table 6 reports the number of collections on which the AutoMLP classifier performs statistically better/worse than the classifier reported in the corresponding column. More details on the per-dataset comparisons are given in Table 4. Specifically, for each dataset every significant worsening w.r.t. AutoMLP (with reference configuration) is starred. The AutoMLP classifier confirmed to be, on average, the most accurate classifiers (e.g., 5 statistically significant accuracy improvements against Naive Bayes using the reference configuration settings against only one significant accuracy worsening).

6.2 Impact of n-grams and feature selection

We separately analyzed the impact of (i) the presence of n-grams and (ii) the use of feature selection on the accuracy performance of the proposed approach. To address issue (i) we repeated all the experiments described in Section 6.1 on a simplified version of the datasets including only unigrams. To address issue (ii) we repeated all the experiments by enabling and disabling the feature selection step.

The achieved results are quite interesting: due to the combinatorial growth in the number of generated combinations, before feature selection 3 features out of 4 are bigrams (approximately 9000 bigrams against 2000 unigrams). Conversely, after feature selection only 1 feature out of 4 is a bigram. Hence, the feature selection step pruned a quite large number of bigrams which were deemed as not discriminative for classification purposes. The presence of bigrams improved the accuracy of the predictions (e.g. AutoMLP +1.2% by using the unified model). Therefore, although many combinations of terms are not significant thus they are early pruned, a subset of them is very informative and strongly correlated with the class label.

We analyzed also the impact of parameter K on the accuracy results. Figure 2 plots the average accuracy values achieved by AutoMLP on three representative DUC'05 collections with different values of K by considering also bigrams. The three collections are characterized by different number of sentences, average sentence length, and word stem frequencies. For example, collection d366i is rather

TABLE 6
Statistical tests of significance. x/y indicates that the AutoMLP classifier with standard/tuned configuration is significantly better x times (worse y times) than the classifier in column

	Unified model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	5/1	3/1	5/1	9/0	8/0	8/0	6/0
AutoMLP tuned config.	7/0	3/1	6/1	11/0	9/0	9/0	6/1
	Level-1 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	3/1	4/2	6/4	7/2	6/0	4/0	6/2
AutoMLP tuned config.	4/1	5/3	6/3	9/1	7/0	6/1	8/0
	Level-2 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	2/2	1/1	3/1	4/2	6/1	4/1	3/1
AutoMLP tuned config.	1/1	1/1	4/1	4/0	9/0	5/1	4/0
	Level-3 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	3/2	2/1	3/0	4/1	5/1	3/1	6/2
AutoMLP tuned config.	4/1	2/0	3/0	6/1	6/2	4/2	8/1
	Level-4 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	5/3	3/2	2/1	3/0	6/2	4/2	3/1
AutoMLP tuned config.	6/3	3/2	3/1	4/0	7/0	5/1	4/1
	Level-5 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	4/2	2/1	2/0	3/1	3/1	3/2	2/0
AutoMLP tuned config.	4/1	2/1	1/0	5/3	4/2	5/3	3/0
	Level-6 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	3/1	3/2	2/0	4/2	7/3	5/1	4/1
AutoMLP tuned config.	5/2	3/1	1/0	6/3	8/2	6/0	5/0
	Level-7 model. Paired t-test, p=0.05						
	Naive Bayes	**L^3**	**SLOGREG**	**LibSVM**	**K-NN**	**ID3**	**Ripper**
AutoMLP reference config.	2/2	1/1	2/1	3/2	4/2	3/1	3/1
AutoMLP tuned config.	1/1	1/1	1/0	3/1	5/1	3/0	3/0

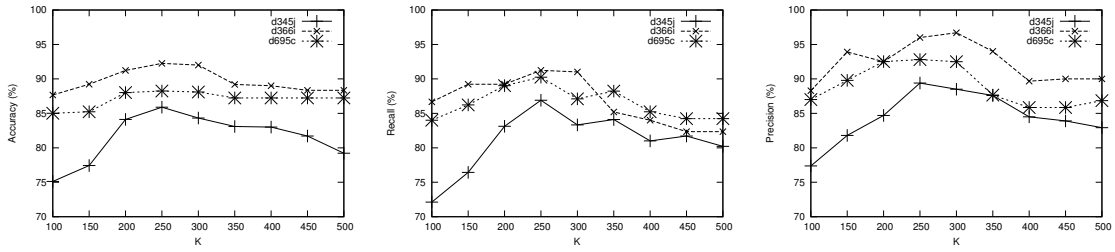
small and contains a limited number of word stems (approximately 30% less than both d345j and d695c) but term frequencies are on average higher (approximately 10% higher than the others). Collection d345j contains approximately twice the number of sentences of d366i, but sentence length is rather limited, whereas collection d695c is large and consists of relatively long sentences. According to the achieved results, by varying the value of K between 200 and 350 the quality of the predictions remains roughly stable. Similar results were achieved on the other document collections. The best performance was achieved on small- and medium-size collections, because they include a larger number of frequently occurring stems. However, the variance of the accuracy values achieved on all the documents in the DUC'05 collection is relatively low (below 4%) by varying K in this value range, whereas it significantly increases by considering a larger set of K values (approximately 12% with K between 100 and 500). Hence, in the selected value range, the sensitivity of this parameter on classifier performance is relatively low. On the other hand, the number and size of the documents within each collection may affect the quality of the prediction while setting K values out of the suggested value range. For example, when coping with very small document collections, setting very large values of K may yield data overfitting. Conversely, while setting K in the range [200, 350], the classifier performance appeared to be roughly stable for all the tested document collections. Based on the achieved results, we recommend to set K to 250 as default setting.

6.3 Recall and precision of the predictions

The accuracy measure does not consider the presence of false positives/negatives in the prediction outcomes. In our context, false positives are particularly harmful, because they imply an excessive use of text highlights.

To gain insights into the quality of the highlight predictions, we computed also precision and recall of class *highlighted*. Precision is the ratio between the number of sentences correctly assigned to class *highlighted* and the total number of sentences assigned to class *highlighted*. It measures to what extent highlight predictions are reliable (i.e., the percentage of true positives over all positive predictions). Recall is the ratio between the number of sentences correctly assigned to class *highlighted* and the total number of sentences actually belonging to class *highlighted*. It measures the completeness of the prediction set with respect to the actual set of highlighted sentences (i.e., the percentage of true positives over the number of expected positives).

According to the achieved results, the presence of false positives is quite limited both on benchmark and on real data. All classifiers are fairly precise, i.e., precision of class *highlighted* above 83%. The most precise classifier is AutoMLP (87.92%). L^3, AutoMLP, LibSVM, SLOGREG achieved good recall values as well (all above 80%), whereas the recall values of Naive Bayes, Ripper, and k-NN are a bit lower (between 74% and 79%). Figures 2(b) and 2(c) plot the precision and recall values of class *highlighted* achieved on three representative DUC'05 collections with different values of K.



(a) Accuracy. (b) Recall of class *Highlighted*. (c) Precision of class *Highlighted*.

Fig. 2. Impact of parameter K on classification performance.

TABLE 7
Real teaching document. Top-10 highlighted sentences.

Highlighted sentence	Rank	Num. of highlights
It takes time to bring about large-scale change, but in the future I think we will be living in cities that fundamentally operate differently.	1st	12
Overall, the changes to these smart cities will be incremental, whether they are driven by researchers like Beckman and Catlett, urban planners, private innovators, or some combination.	1st	12
Experts say cities that capitalize on all the new urban data could become more efficient and more enjoyable places to live.	3rd	11
City planners will be able to make more informed decisions about where to place new bus stops or how much road salt to apply to certain areas after a heavy snow.	3rd	11
Now you have city governments regrouping and developing comprehensive long-range visions of the role information technology will play in making their city better.	5th	10
The leading urban centers are not placing their technological futures in the hands of a company or a single university research group.	6th	9
The challenges are diverse and demanding, but a handful of new projects around the world are providing a glimpse of what a truly smart city could offer.	7th	8
As urban populations increase, the number of data-generating sensors and Internet-connected devices will grow even faster.	8th	5
This public utility approach is one of the common threads linking the different smart city projects.	9th	5
As an example, Birchenall cites Glasgow's new cycling app, which helps riders plan routes, easily locate bicycle racks, and more;	10th	3

6.4 Validation on real learning documents

We studied the applicability of the proposed approach in a real learning scenario. We collected highlights on a teaching document by means of a crowd-sourcing experience with students of a university-level Computer Science course. The experience, carried out on a voluntary basis, lasted one month and was conducted by involving 52 students of a Database course (a 2nd year B.S. course) given by the Politecnico di Torino, an Italian technical university. Students were invited to participate to the crowd-sourcing experience by filling an anonymous questionnaire available at the course website. The activity consisted in reading a scientific article [44], which presents the key aspects behind the diffusion of Smart Cities and Open Data, and highlighting sentences from the given article based on their personal judgment and experience. The article was given to the students as additional teaching material and was deemed as an example of technical document whose topic can be of interest for most of the course participants. The document is characterized by 52 sentences, 22 of which were highlighted by at least one student. Table 7 reports the top-10 highlighted sentences in the teaching document. For each sentence, the textual content, the number of user highlights, and the corresponding ranking are reported.

The goal of this experimental study was to apply the HIGHLIGHTER approach to real teaching documents enriched with manual highlights to automatically predict new highlights. Since we have no a priori knowledge about the proficiency of individual students, we assumed that all the students have a basic knowledge on the topic. To this purpose, we applied a train-test leave-one-out validation strategy, which is appropriate for small/medium-size datasets [35]. Specifically, we randomly partitioned the dataset in two parts, hereafter denoted as *train* and *test*. The *train* consists of all the document sentences, except for one. Sentences in the train are enriched with the corresponding highlights and used to generate the classification models. The *test* comprises the remaining sentence and it is exploited to perform highlight predictions by temporarily ignoring the actual class value. The test sentence is labeled as *highlighted* or *non-highlighted* according to the prediction made by the classification model. Then, to evaluate classifier performance prediction values are compared with the expected values. The procedure is iterated by considering all the possible combinations of train and test sets.

Before running the classifiers, the textual content is prepared to the next classification process by applying the preprocessing and feature selection steps, executed with K=250. For example, according to tf-idf ranking, unigrams *public*, *future*, *technology* and bigram *public utility* are selected because they all placed in the top-10 tf-idf ranking.

The classification results, which were achieved by using the reference configurations reported in Section 6.1 for all the tested algorithms, can be summarized as follows: L^3, AutoMLP, and LibSVM correctly classified 45 sentences out of 52 (average accuracy 86.54%), SLOGREG 43 (82.69%), Naive Bayes, and Ripper 42 (80.77%), and k-NN 40 (76.92%). The fairly high accuracy values confirm that the classification models are able to capture the most significant correlations between the text features and the class.

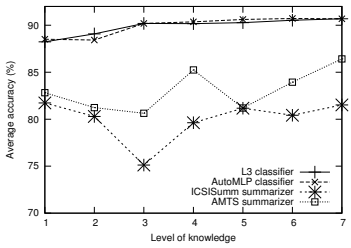


Fig. 3. Accuracy comparison between best performing classifiers and text summarizers by varying the level of knowledge.

6.5 Comparison with summarization approaches

Most general-purpose document summarizers are unsuitable for addressing the problem under analysis, because they select a significant subset of document sentences without considering any prior knowledge on the document collection (i.e., past highlights are ignored). Furthermore, sentences are not only evaluated based on their individual interest, but also according to their pertinence and level of redundancy with respect to the already selected sentences.

To investigate the use of document summarizers to address highlight prediction we identified a summarization contest that is somehow related to the problem under analysis. Specifically, in the Update Summarization Task of the Text Analysis Conference (TAC) 2008, the contest was to generate the summaries of two document sets A and B. To summarize set B we assume to know the content of the documents in A. To tailor the document summarization process to the problem under analysis, we considered the set of highlights of the sentences in A as the summary of A and we summarized the documents in B given the summary of A. A more thorough description of the addressed task is given in [42]. To perform the experimental evaluation, we considered the following two summarizers: (i) a recently proposed summarizer relying on word association discovery, i.e., Association Mixture Text Summarization (AMTS) [41], (ii) a widely used open source text summarizer relying on the Integer Linear Programming, i.e., Integer Linear Programming-based ICSI multi-document summarization system (ICSIsumm) [42]. ICSIsumm achieved excellent performance on the TAC'08 datasets, while AMTS is a recently proposed and quite effective document summarizer that considers also the user-specified background knowledge beyond the original document content.

Similar to what previously described in Section 6.1, to estimate the average accuracy of each summarizer we applied a 10-fold stratified cross validation test on the benchmark DUC'05 document collections. Specifically, for each fold we ran the summarizer on the textual content of the test set using, as background knowledge, the document highlights in the training set. The sentences included in the output summary, which represent the result of the prediction, are then compared with the actual highlights in the test set to count the number of matchings. Figure 3 compares the average accuracy values achieved by the ICSIsumm and AMTS summarizers with different levels of knowledge with those achieved by the two best performing classifiers AutoMLP and L^3 on the DUC'05 collections. By using unified models the accuracy values are 86.99% for AutoMLP, 86.94% for the L^3 classifier, 78.95% for ICSISumm, and 79.15% for AMTS. The two classifiers performed significantly better than the document summarizers (approximately +10%) with both per-level and unified models. According to the 10-fold cross validated paired t-test, the improvements achieved by AutoMLP with respect to both summarizers are statistically significant on 9 datasets out of 27.

6.6 Execution time

Learning classification models is the most computationally intensive task of the presented approach. For this reason, we analyzed the time spent in learning classification models with different techniques on the benchmark datasets.

Depending on the analyzed data distribution, Support Vector Machines and Neural Networks took between few minutes to 1 hour (approximately) to learn the classifier. Conversely, the training times of L^3, Decision Trees, Logistic Regression, Ripper, and Naive Bayes were at least one order of magnitude lower. The k-NN classifier does not generate any classification model. Thus, most of the computational time (typically between few seconds and few minutes) is spent in class value prediction. For all the other classifiers, the prediction time is negligible.

7 Conclusions and Future Work

This paper proposes HIGHLIGHTER, a new approach to automatically generating highlights of learning documents. It generates classification models tailored to different levels of knowledge from a set of highlighted documents to predict new highlights, which are provided to learners to improve the quality of their learning experience. A performance comparison between various classifiers on benchmark data and an analysis of the usability of the proposed approach on real document collections have been performed. In the current version of the system, highlights are not personalized. Specifically, the same highlights are deemed as appropriate for all the users having the same level of knowledge. As future work, we aim at tailoring the automatically generated highlights to specific users. Therefore, we would like to generate not only unified and per-level models, but also user-centric models. Furthermore, we currently ignore the presence of textual annotations, which could enrich the document content with additional notes or rephrases. We plan to analyze such automatically generated content to gain insights into the level of knowledge of learners.

References

[1] J. L. Moore, C. Dickson-Deane, and K. Galyen, "E-learning, online learning, and distance learning environments: Are they the same?" *The Internet and Higher Education*, vol. 14, no. 2, pp. 129 – 135, 2011.

[2] F. Grünewald and C. Meinel, "Implementation and evaluation of digital e-lecture annotation in learning groups to foster active learning," *TLT*, vol. 8, no. 3, pp. 286–298, 2015.

[3] S. Elliott, *Educational Psychology: Effective Teaching, Effective Learning*. McGraw-Hill, 2000.

[4] A. B. Alencar, M. C. F. de Oliveira, and F. V. Paulovich, "Seeing beyond reading: A survey on visual text analytics," *Wiley Int. Rev. Data Min. and Knowl. Disc.*, vol. 2, no. 6, pp. 476–492, Nov. 2012.

[5] C. Aggarwal and C. Zhai, "A survey of text classification algorithms," in *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds. Springer US, 2012, pp. 163–222.

[6] D. D. Lewis, "Naive (bayes) at forty: The independence assumption in information retrieval," in *ECML*, 1998, pp. 4–15.
[7] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.
[8] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.
[9] W. W. Cohen, "Fast effective rule induction," in *In Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995, pp. 115–123.
[10] E. Baralis, S. Chiusano, and P. Garza, "A lazy approach to associative classification," *IEEE TKDE*, vol. 20, no. 2, pp. 156–171, 2008.
[11] Document Understanding Conference, "HTL/NAACL workshop on text summarization," 2004.
[12] Y. H. Lee, G. D. Chen, L. Y. Li, Nurkhamid, C. Y. Fan, and K. H. Chiang, "The effect of utilizing the learning skill of highlighting and constructing a map in a networked hyperlink condition on learning performance," in *2012 IEEE 12th International Conference on Advanced Learning Technologies*, July 2012, pp. 546–548.
[13] D. Suthers and K. Verbert, "Learning analytics as a "middle space"," in *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, ser. LAK '13. New York, NY, USA: ACM, 2013, pp. 1–4.
[14] N. Milic-Frayling and R. Sommerer, "Facility for highlighting documents accessed through search or browsing," Feb. 9 2010, uS Patent 7,660,813.
[15] A. Patel and D. desJardins, "Highlighting occurrences of terms in documents or search results," Dec. 14 2010, uS Patent 7,853,586.
[16] B. Cragun and P. Day, "Apparatus and method for automatically highlighting text in an electronic document," Mar. 20 2007, uS Patent 7,194,693.
[17] J. J. Hull and D.-S. Lee, "Simultaneous highlighting of paper and electronic documents," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 4, 2000, pp. 401–404 vol.4.
[18] R. Ferguson, "Learning analytics: Drivers, developments and challenges," *Int. J. Technol. Enhanc. Learn.*, vol. 4, no. 5/6, pp. 304–317, Jan. 2012.
[19] G. Siemens, "Learning analytics: Envisioning a research discipline and a domain of practice," in *Proceedings of the 2Nd International Conference on Learning Analytics and Knowledge*, ser. LAK '12. New York, NY, USA: ACM, 2012, pp. 4–8.
[20] A. M. Shahiri, W. Husain, and N. A. Rashid, "A review on predicting student's performance using data mining techniques," *Procedia Computer Science*, vol. 72, pp. 414 – 422, 2015.
[21] A. Wolff, Z. Zdrahal, A. Nikolov, and M. Pantucek, "Improving retention: Predicting at-risk students by analysing clicking behaviour in a virtual learning environment," in *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, ser. LAK '13. New York, NY, USA: ACM, 2013, pp. 145–149.
[22] X. Zhou, B. Wu, and Q. Jin, "Open learning platform based on personal and social analytics for individualized learning support," in *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing*, Aug 2015, pp. 1741–1745.
[23] A. Ezen-Can, K. E. Boyer, S. Kellogg, and S. Booth, "Unsupervised modeling for understanding mooc discussion forums: A learning analytics approach," in *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, ser. LAK '15. New York, NY, USA: ACM, 2015, pp. 146–150.
[24] G. Yang, D. Wen, Kinshuk, N.-S. Chen, and E. Sutinen, "Personalized text content summarizer for mobile learning: An automatic text summarization system with relevance based language model," in *Technology for Education (T4E), 2012 IEEE Fourth International Conference on*, July 2012, pp. 90–97.
[25] S. Saraswathi, M. Hemamalini, S. Janani, and V. Priyadharshini, ser. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2011, vol. 193.
[26] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85 – 117, 2015.
[27] P. Frasconi, G. Soda, and A. Vullo, "Text categorization for multi-page documents: A hybrid naive bayes hmm approach," in *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '01. New York, NY, USA: ACM, 2001, pp. 11–20.
[28] C. Apté, F. Damerau, and S. M. Weiss, "Automated learning of decision rules for text categorization," *ACM Trans. Inf. Syst.*, vol. 12, no. 3, pp. 233–251, Jul. 1994.
[29] Z. A. Pardos, R. S. J. D. Baker, M. O. C. Z. San Pedro, S. M. Gowda, and S. M. Gowda, "Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes," in *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, ser. LAK '13. New York, NY, USA: ACM, 2013, pp. 117–124.
[30] D. Wang, S. Zhu, T. Li, Y. Chi, and Y. Gong, "Integrating document clustering and multidocument summarization," *ACM Trans. Knowl. Discov. Data*, vol. 5, pp. 14:1–14:26, August 2011.
[31] E. Baralis, L. Cagliero, N. A. Mahoto, and A. Fiori, "Graphsum: Discovering correlations among multiple terms for graph-based summarization," *Inf. Sci.*, vol. 249, pp. 96–109, 2013.
[32] J. Steinberger, M. Kabadjov, R. Steinberger, H. Tanev, M. Turchi, and V. Zavarella, "Jrc's participation at tac 2011: Guided and multilingual summarization tasks," in *TAC'11: Proceedings of the The 2011 Text Analysis Conference*, 2011.
[33] D. Gillick, B. Favre, D. Hakkani-Tur, B. Bohnet, Y. Liu, and S. Xie, "The ICSI/TUD summarization system at TAC 2009," in *Proceedings of the Text Analysis Conference*, ser. TAC '09, Gaithersburg, MD (USA), 2009.
[34] E. Baralis, L. Cagliero, A. Fiori, and P. Garza, "Mwi-sum: A multilingual summarizer based on frequent weighted itemsets," *ACM Trans. Inf. Syst.*, vol. 34, no. 1, p. 5, 2015.
[35] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining, (First Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005.
[36] C.-Y. Lin and E. Hovy, "Automatic evaluation of summaries using n-gram co-occurrence statistics," in *Proceedings of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, 2003, pp. 71–78.
[37] M. I. Jordan and C. M. Bishop, "Neural networks," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 73–75, Mar. 1996.
[38] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theor.*, vol. 13, no. 1, pp. 21–27, Sep. 2006.
[39] S. K. Shevade and S. S. Keerthi, "A simple and efficient algorithm for gene selection using sparse logistic regression." *Bioinformatics*, vol. 19, no. 17, pp. 2246–2253, 2003.
[40] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*. New York, NY, USA: Cambridge University Press, 2011.
[41] O. Gross, A. Doucet, and H. Toivonen, "Document summarization based on word associations," in *Proceedings of the 37th ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '14. New York, NY, USA: ACM, 2014, pp. 1023–1026.
[42] D. Gillick, B. Favre, and D. Hakkani-Tur, "The ICSI summarization system at TAC 2008," in *Proceedings of the Text Analysis Conference*, ser. TAC '08, Gaithersburg, MD (USA), 2008.
[43] T. Copeck and S. Szpakowicz, "Leveraging pyramids," in *DUC'05: Proceedings of the 2005 Document Understanding Conference*, 2005.
[44] G. Mone, "The new smart cities," *Commun. ACM*, vol. 58, no. 7, pp. 20–21, Jun. 2015.

Elena Baralis is full professor at the Dipartimento di Automatica e Informatica of the Politecnico di Torino since January 2005. She holds a a PhD in Computer Engineering from Politecnico di Torino. Her current research interests are in the field of database systems and data mining, specifically on mining algorithms for very large databases and sensor/stream data analysis. She has published over 80 papers in international journals and proceedings.

Luca Cagliero has been an assistant professor at the Dipartimento di Automatica e Informatica of the Politecnico di Torino since January 2016. He holds a M.Sc. degree in Computer and Communication Networks and a PhD in Computer Engineering both from Politecnico di Torino. His current research interests are in the fields of Data Mining and Database Systems and, specifically, on classification, summarization, and pattern mining from structured and semi-structured data.