Automatically Suggesting Topics for Augmenting Text Documents

Robert West
School of Computer Science
McGill University
Montréal, Québec, Canada
rwest@cs.mcgill.ca

Doina Precup
School of Computer Science
McGill University
Montréal, Québec, Canada
dprecup@cs.mcgill.ca

Joelle Pineau
School of Computer Science
McGill University
Montréal, Québec, Canada
jpineau@cs.mcgill.ca

ABSTRACT

We present a method for automated topic suggestion. Given a plain-text input document, our algorithm produces a ranking of novel topics that could enrich the input document in a meaningful way. It can thus be used to assist human authors, who often fail to identify important topics relevant to the context of the documents they are writing. Our approach marries two algorithms originally designed for linking documents to Wikipedia articles, proposed by Milne and Witten [15] and West *et al.* [22]. While neither of them can suggest novel topics by itself, their combination does have this capability. The key step towards finding missing topics consists in generalizing from a large background corpus using principal component analysis. In a quantitative evaluation we conclude that our method achieves the precision of human editors when input documents are Wikipedia articles, and we complement this result with a qualitative analysis showing that the approach also works well on other types of input documents.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*linguistic processing*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*; I.7.1 [**Document and Text Processing**]: Document and Text Editing—*document management*

General Terms

Algorithms, Experimentation

Keywords

Topic Suggestion, Principal Component Analysis, Eigenarticles, Data Mining, Wikipedia

1. INTRODUCTION

As of 2010, machines cannot think. They cannot understand natural language, and they cannot produce it in creative ways. These are still human prerogatives. However, while humans are intelligent and creative, they are sometimes forgetful, or not thorough enough: when writing text documents, we often fail to mention all relevant topics, be it that we are unaware of the missing topics being relevant, or be it that we simply forget to include them. Computers, on the other hand, excel at large-scale book-keeping and fast retrieval, as long as no deep understanding is required. Hence, although not creative *per se,* computers can still support the creativity of humans.

The main contribution of this paper is a method that can assist human creativity by automatically suggesting topics to authors of text documents. To the best of our knowledge, we are the first to address this task. Given a plain-text document, our algorithm outputs a ranked list of novel topics that could enrich the input document in a meaningful way. The author can then inspect the suggestions and decide which of them should be incorporated into the document.

Our approach couples two algorithms originally designed for linking documents to Wikipedia articles, proposed by Milne and Witten [15] and West *et al.* [22]. While neither of them can suggest novel topics by itself, their combination does have this capability. The key step towards finding missing topics consists in generalizing from a large background corpus, such as Wikipedia. Intuitively, a missing topic is suggested if it appears in many documents of the corpus that are similar to the input document. Technically, the generalization is performed using principal component analysis.

An automated topic suggestion system could be widely applicable. For instance, it is often impossible for journalists to be experts in all the fields about which they write. In this scenario, the journalist would first write a draft of the article, feed it to our system, and use some of the resulting topic suggestions to make the article more complete. There are many other user groups that could also profit from automated topic suggestion, e.g., lawmakers trying to avoid loopholes in a legal text, or Wikipedia contributors working to make an article as comprehensive as possible, to name but a few.

The remainder of this paper is structured as follows: Section 2 summarizes related work. In Section 3 we provide a high-level overview of our method, while Section 4 describes the technical details. Section 5 contains a quantitative evaluation showing that our method achieves the precision of human editors when input documents are Wikipedia articles. In Section 6 we complement this result with a qualitative analysis showing that the approach also works well on other types of input documents. Finally, Section 7 contains conclusions and avenues for future research.

2. RELATED WORK

To the best of our knowledge, the task of topic suggestion has not been studied extensively yet. While we are not aware of a topic suggestion algorithm for plain-text documents, Maguitman *et al.* [10] developed such a method for 'concept maps', i.e., semantic network–like graphical representations that can 'facilitate knowl-

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

edge capture for human examination and sharing' [10]. Their system can assist people during the process of drawing a concept map, by proposing topics that are novel yet related to the concept map produced so far.

Fortuna *et al.* [7] propose a solution for the similar task of constructing ontologies, i.e., networks of topics interconnected by relations. In Fortuna *et al.*'s setting, the ontology is built by a human knowledge engineer with the help of their software. Whereas Maguitman *et al.*'s focus is on generating topics that are both related and novel to the current context, novelty does not play a central role in Fortuna *et al.*'s system. Instead, it suggests potential subtopics for the topic node on which the knowledge engineer is currently working.

Wang *et al.* [20] deal with visual topic suggestion for group brainstorming situations. Their system analyzes the utterances of the participants of a brainstorming session and chooses and displays images that are related to the current context, with the goal of enhancing creativity by triggering novel ideas in the humans perceiving these visual stimuli.

Closely related to topic suggestion is keyphrase extraction: before one can find novel topics to add to a document, one should know which ones it already contains. Consequently, all of the aforementioned systems have a component that identifies the keyphrases, or main topics, of the current context. Note that we use the terms 'topic' and 'keyphrase' interchangeably, which differs from some other authors' nomenclature; e.g., in latent Dirichlet allocation [3] topics are defined as probability distributions over words.

Keyphrase extraction is usually distinguished from keyphrase assignment [18]. In keyphrase extraction any n-gram of the input document can potentially be returned as a keyphrase, whereas in keyphrase assignment a predetermined set of keyphrase candidates is assumed. Both types of applications are usually cast as supervised classification tasks [18].

Wikipedia link prediction is similar to keyphrase assignment. Given a plain-text document, the task is to (1) find the n-grams that should serve as anchors for links to Wikipedia articles, and (2) for each anchor, identify the correct target article. This problem has also been tackled with supervised machine learning techniques, e.g., by Mihalcea and Csomai [12] and by Milne and Witten [15].

Another class of algorithms for Wikipedia link prediction leverages unsupervised machine learning techniques. Fissaha Adafre and de Rijke [6] use clustering, while West *et al.* [22] use principal component analysis. Unlike the supervised algorithms [12, 15], these methods do not operate on plain text but augment articles that already contain a number of Wikipedia links, by adding new links that are justified by the pre-existing hyperlink structure.

We leverage some of these techniques [15, 22] as part of our approach to topic suggestion.

3. OUTLINE OF OUR METHOD

Figure 1 sketches the components of our system and the information flow between them. In this section, we will provide a high-level overview, while Section 4 contains technical descriptions of all components.

Keyphrase assignment. The input to our method is a plain-text document, i.e., a sequence of words $\mathbf{d}$. It is fed to the first component of our processing cascade, a keyphrase assignment algorithm. This module identifies the main topics of the input document and produces a high-dimensional, binary vector representation $\mathbf{v}$ of it. In this *topic vector,* each entry corresponds to a candidate topic (recall that in keyphrase assignment, the set of candidate topics is predetermined and static); the topics appearing in the input document have a value of 1, all others are 0. Since only a small fraction of all candidate topics appear in a document, $\mathbf{v}$ is extremely sparse.

Generalization. The centerpiece of our system is the generalization module. While the left and right boxes of Figure 1 may be thought of as pre- and postprocessing steps, respectively, the actual topic suggestion algorithm resides here. It takes as input the topic vector $\mathbf{v}$ that results from keyphrase assignment, and produces as output a vector $\tilde{\mathbf{v}}$, called *generalized topic vector,* of the same dimensionality as $\mathbf{v}$. While each entry of $\tilde{\mathbf{v}}$ still refers to the same candidate topic as the corresponding entry in $\mathbf{v}$, the values differ, and unlike the sparse and binary $\mathbf{v}$, vector $\tilde{\mathbf{v}}$ is dense and real-valued. Entries that are zero in $\mathbf{v}$ but much greater in $\tilde{\mathbf{v}}$ correspond to topics that are not keyphrases of the input document but that are suggested as such by our algorithm. We construct $\tilde{\mathbf{v}}$ by generalizing from a large background corpus using principal component analysis (PCA), a mathematical technique that is commonly used for reducing noise in data. Its usage within our algorithm may be understood intuitively in terms of noise reduction, too: if the absence of a topic j from the input document $\mathbf{d}$ (i.e., the entry $v_j = 0$ in the topic vector $\mathbf{v}$) is caused by noise, then this noise can be eliminated by adding j to $\mathbf{d}$ (i.e., by giving the entry $\tilde{v}_j$ a positive value, rather than zero). We say that the absence of topic j is due to noise if it constitutes a significant deviation from the overall patterns present in the background corpus, i.e., if many documents similar to $\mathbf{d}$ contain topic j.

Filtering and ranking. After generalization, the topic suggestions are implicitly given in the vector $\tilde{\mathbf{v}}$, as the entries that are much greater than they were in $\mathbf{v}$. However, we want to exclude those topics that are not novel, i.e., that are already contained in the input document $\mathbf{d}$ as n-grams (not necessarily as keyphrases). This constitutes the filtering step. The remaining topic suggestions are ranked in order of decreasing quality, as is common practice in information retrieval systems. Fortunately, the generalization step generates meaningful numerical values that can directly serve as indicators of suggestion quality, so we simply rank the suggested topics according to their values in the generalized topic vector $\tilde{\mathbf{v}}$.

4. DESCRIPTION OF THE COMPONENTS

After the high-level view of the previous section, we will now go into more detail regarding each component of our topic suggestion system.

4.1 Keyphrase assignment

The task of the keyphrase assignment component is to identify the important topics of a given plain-text document. In principle, any keyphrase assignment algorithm can be used in this step. As mentioned in Section 2, all such algorithms work with a predetermined set of candidate topics (let us call it $\mathfrak{T}$). This set defines the dimensions of the topic vectors $\mathbf{v}$ and $\tilde{\mathbf{v}}$ of Figure 1. As we want our method to be domain-independent, $\mathfrak{T}$ should be as general as possible. We define $\mathfrak{T}$ as the set of topics for which a Wikipedia article exists, since Wikipedia's coverage is so vast that nearly any conceivable topic has a corresponding Wikipedia article. If for any reason only domain-specific candidate topics should be considered, $\mathfrak{T}$ can be restricted accordingly.

According to the Wikipedia linking guidelines, the keyphrases of an article should serve as anchors for hyperlinks to other articles: links should represent 'relevant connections to the subject of another article that will help readers to understand the current article more fully' [25]. Consequently, a program that can successfully identify the Wikipedia anchors of an input document (i.e., the n-grams that should serve as link anchors to Wikipedia articles) can also be used for the task of domain-independent keyphrase assignment. In Section 2 we have mentioned two methods [12, 15] that

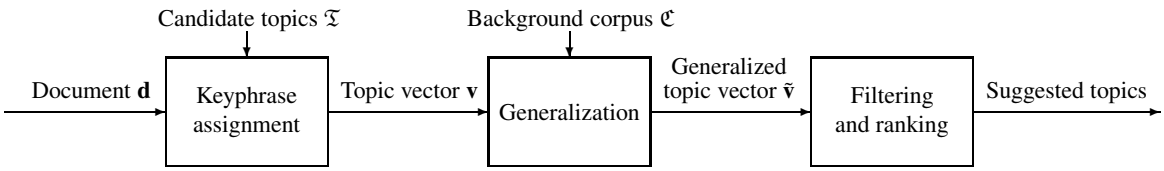

Figure 1: High-level diagram describing our method.

can augment plain text with Wikipedia links and that can therefore be plugged into our system as keyphrase assignment modules. We use Milne and Witten's method [15] because it outperforms Mihalcea and Csomai's [12] and because it is publicly available as part of the WikipediaMiner toolkit [13].

In a nutshell, Milne and Witten's algorithm is a machine learning classifier that decides for each n-gram of a plain-text input document to which Wikipedia article (if any) it refers and with what probability it should serve as an anchor to that article. It is trained in a supervised manner on Wikipedia articles, which can serve as labeled examples, since each article contains numerous link anchors. Although it is trained entirely on Wikipedia, the classifier performs as well on newswire stories as it does on Wikipedia articles, as shown in a human user evaluation. This is important because we want our method to work on arbitrary text documents, not only on Wikipedia articles.

Milne and Witten's link prediction algorithm offers several parameters that can be configured, but we use the default settings for all of them. Since the classifier outputs a probability for each phrase of the input text, we need to define a threshold above which we accept proposed keyphrases. We set this parameter to 0.5, i.e., we take a phrase to be a keyphrase if the classifier considers this more likely than not.

Note that if the input document already contains links to Wikipedia articles, keyphrase assignment is trivial. In this case we simply use the targets of the existing links as the document's keyphrases.

4.2 Generalization

The generalization component can be understood as reducing the noise in the topic vector **v** in the context of a large background corpus. We do not use this corpus in its plain-text form; rather, we preprocess it using PCA. In this section we explain the use of PCA for generalization, in both intuitive and technical terms.

4.2.1 The background corpus

The background corpus can be any sufficiently large collection of text documents, e.g., Wikipedia articles, newswire stories, scientific papers, parliament debates, or law texts, to name but a few. Let $\mathfrak{C} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_M\}$ be the set of documents in the corpus, where M is the number of documents. The same keyphrase assignment algorithm we use to find the topics of the input document (cf. Section 4.1) is also run on each document $\mathbf{c}_i$ of the corpus, in order to obtain its topic vector $\mathbf{t}_i$. If we take topic vectors as row vectors, then we can combine them in a matrix **T** whose i-th row is $\mathbf{t}_i$. We call **T** the *document–topic matrix*. Recall that the entries of a topic vector correspond to the candidate topics $\mathfrak{T}$. Thus, if there are N topics, **T** is of size $M \times N$. When we define $\mathfrak{T}$ as the set of Wikipedia articles—as we do throughout this paper—, N will be very large, and it follows that $M \ll N$.

4.2.2 Principal component analysis

Geometrically, **T** can be interpreted as a cloud of M points in an N-dimensional Euclidean space (called *document space*), where each topic vector (row of **T**) is represented by one point. This cloud will in general not extend equally in all directions but rather appear squished along certain axes and elongated along others, due to correlations in the data. PCA finds a set of orthogonal axes along which the data cloud is maximally spread out. More specifically, the first axis found by PCA, the so-called first principal component, is the vector in the N-dimensional space along which the variance of the data is maximized (when the data is projected onto it); the second principal component is constrained to be orthogonal to the first and chosen such that the variance is again maximal when the data is projected onto it; and so on up to M. For technical reasons (cf. Section 4.2.3), the principal components are called *eigenarticles,*[1] and the space spanned by them *eigenspace.* The less important eigenarticles merely account for minor variations in the data, variance being small along their directions. Such variations can be considered noise, and by eliminating them, the noise in the data is reduced. By convention, all eigenarticles are normalized to a length of 1. We write eigenarticles as row vectors and stack them on top of each other, thus obtaining the *eigenarticle matrix* **E** of size $M \times N$.

Now assume we want to reduce the noise in a new topic vector **v**. We can achieve this in three steps. First, project **v** onto the basis spanned by the eigenarticles; this way we obtain **v**'s eigenspace representation

$$\mathbf{p} = (p_1, p_2, ..., p_M) = \mathbf{v}\mathbf{E}^\top. \tag{1}$$

Second, eliminate the noise that manifests itself as variation along minor eigenarticles by setting the respective co-ordinates to zero, keeping only the entries $(p_1, p_2, ..., p_K)$ as non-zero, for some fixed K (the so-called *eigenspace dimensionality*). Call the resulting vector $\tilde{\mathbf{p}}$. Third, project $\tilde{\mathbf{p}}$ from eigenspace back into the original document space spanned by the canonical basis vectors. This yields the generalized topic vector

$$\tilde{\mathbf{v}} = \tilde{\mathbf{p}}\mathbf{E}. \tag{2}$$

In the above explanation, we have included the second step just to be conceptually clear. In practice, it suffices to store only the first K eigenarticles. (We use $\tilde{\mathbf{E}}$ to denote the resulting eigenarticle matrix of reduced size $K \times N$.) Equations (1) and (2) may then be combined, and the generalized topic vector computed as

$$\tilde{\mathbf{v}} = (\mathbf{v}\tilde{\mathbf{E}}^\top)\tilde{\mathbf{E}}. \tag{3}$$

After finding the eigenarticles, the background corpus $\mathfrak{C}$ need not be stored explicitly. We only need its reduced eigenarticle matrix $\tilde{\mathbf{E}}$. It is important to note that $\tilde{\mathbf{E}}$ has to be computed only once, in an offline preprocessing step. During normal operation of our algorithm, only single projections into eigenspace and back are performed, according to (3).

4.2.3 Technical issues

For clarity's sake, we have so far glanced over several technical issues. In this section we provide some details that are important for

[1]The term 'eigendocument' would be more appropriate, but we stick to the nomenclature of the paper in which the concept was first introduced [22]. There, the authors use the term 'eigenarticle' because all their documents are Wikipedia articles.

making our algorithm mathematically sound and computationally efficient.

Topic candidate selection. Above, we have defined the set $\mathfrak{T}$ of candidate topics as the set of all Wikipedia articles. However, the Wikipedia snapshot we use [24] has about 2.7 million articles, i.e., the number of topics $N \approx 2.7 \times 10^6$, which would make the reduced eigenarticle matrix $\tilde{\mathbf{E}}$ too large to fit into memory. We therefore follow the approach of West *et al.* [22] and include in $\mathfrak{T}$ only those Wikipedia articles with at least 15 incoming and 15 outgoing links. This way many unimportant articles are discarded and N is reduced to 468,510, or 17% of the original size.

Topic weighting. As defined in Section 3, topic vectors $\mathbf{v}$ are binary. However, not all topics present in a document are equally informative; e.g., the fact that a document mentions a rare concept such as CN TOWER is much more salient than the fact that it talks about something more common such as CANADA. We therefore apply the IDF weighting scheme also used by Milne and Witten [14] and West *et al.* [22], according to which a topic gets more weight if it appears in fewer documents of the background corpus $\mathfrak{C}$: before feeding topic vector $\mathbf{v}$ to the generalization component, we weight its j-th entry by a factor of $\log(M/M_j)$, where $M_j = \sum_{i=1}^{M} t_{ij}$ counts the documents containing the j-th topic. The weighting is also performed on each row of the document–topic matrix $\mathbf{T}$ before we find its eigenarticles.

Computing PCA efficiently. We call the principal components of $\mathbf{T}$ eigenarticles because mathematically they are the eigenvectors of the covariance matrix (and thus the scatter matrix) of $\mathbf{T}$. Let $\mathbf{m} = \frac{1}{M}\sum_{i=1}^{M} \mathbf{t}_i$ be the *mean topic vector,* and $\mathbf{M}$ a matrix of M rows, all of which are equal to $\mathbf{m}$. Then the scatter matrix is $(\mathbf{T}-\mathbf{M})^\top(\mathbf{T}-\mathbf{M})$, which has dimensionality $N \times N$. To speed up computation and reduce memory requirements, one may compute the eigenvectors of another matrix $\mathbf{L} = (\mathbf{T}-\mathbf{M})(\mathbf{T}-\mathbf{M})^\top$ and obtain the eigenarticles by pre-multiplying these eigenvectors by $(\mathbf{T}-\mathbf{M})^\top$ (cf. West *et al.* [22] for a derivation). Recall that $M \ll N$, so the $M \times M$ matrix $\mathbf{L}$ is much smaller than the $N \times N$ scatter matrix.

Since each document contains only a small fraction of all candidate topics, $\mathbf{T}$ is extremely sparse. But subtracting the mean maps most zeros to negative numbers. So $\mathbf{T}-\mathbf{M}$ is dense, and the naïve approach of first computing $\mathbf{T}-\mathbf{M}$ and then multiplying it with its transpose to obtain $\mathbf{L} = (\mathbf{T}-\mathbf{M})(\mathbf{T}-\mathbf{M})^\top$ takes very long.[2] However, calculating $\mathbf{L}$ as

$$(\mathbf{T}-\mathbf{M})(\mathbf{T}-\mathbf{M})^\top = \mathbf{T}\mathbf{T}^\top + \mathbf{M}\mathbf{M}^\top - \mathbf{T}\mathbf{M}^\top - (\mathbf{T}\mathbf{M}^\top)^\top \qquad (4)$$

takes only a few seconds (as opposed to several hours), since $\mathbf{T}\mathbf{T}^\top$ can be computed fast due to $\mathbf{T}$'s sparsity, $\mathbf{M}\mathbf{M}^\top$ has the constant $\mathbf{m}\mathbf{m}^\top$ everywhere, and all columns of $\mathbf{T}\mathbf{M}^\top$ are equal to $\mathbf{T}\mathbf{m}^\top$.

Projecting into eigenspace efficiently. As outlined above, the task of the generalization component is to project a topic vector $\mathbf{v}$ into reduced eigenspace and then back into the original document space. Since we mean-center the topic vectors of the background corpus before finding the eigenarticles, we must also center $\mathbf{v}$. Projecting into eigenspace in fact amounts to computing $(\mathbf{v}-\mathbf{m})\tilde{\mathbf{E}}^\top$. Here, too, we may decrease running time (by up to four orders of magnitude) by reformulating to $\mathbf{v}\tilde{\mathbf{E}}^\top - \mathbf{m}\tilde{\mathbf{E}}^\top$, where the first term can be computed efficiently because $\mathbf{v}$ is sparse and the second term does not depend on $\mathbf{v}$, i.e., needs to be calculated only once, offline.

[2] If principal components are computed without mean-centering, as the eigenvectors of $\mathbf{T}^\top\mathbf{T}$, the first principal component will approximate the data mean rather than the direction of maximum variance. While this is acceptable in certain applications of PCA, it affects performance negatively in our case, since it results in $\tilde{\mathbf{v}}$ (cf. (3)) being skewed towards the mean, which in turn makes topic suggestions less meaningful.

4.3 Filtering and ranking

The goal of the filtering and ranking step is to prepare the topic suggestions for human inspection.

If the input document $\mathbf{d}$ does not contain the j-th topic of the candidate set $\mathfrak{T}$ and $\tilde{v}_j \gg v_j$ after generalizing, then this suggests that j should be considered for inclusion in $\mathbf{d}$. The difference $\tilde{\mathbf{v}} - \mathbf{v}$, which we call *reconstruction gain* vector, attributes to each topic in $\mathfrak{T}$ a real number quantifying how good a suggestion it would be. Therefore, our algorithm—in the manner of typical information retrieval systems—does not select a subset of topics that are suggested for inclusion in the input document, but rather ranks all potential candidate topics, by sorting them in order of decreasing reconstruction gain.

As our goal is to suggest only *novel* topics, a topic should only be considered if the input document $\mathbf{d}$ does not already mention it. So certainly a topic should be filtered from the list of suggestions if it is a keyphrase of $\mathbf{d}$. But even if it is not a keyphrase, it might still appear in the plain text as an ordinary phrase and should thus be filtered, too. There are several ways of determining whether $\mathbf{d}$ mentions a topic j. Recall that we represent topics as Wikipedia articles. The simplest solution would be to check if $\mathbf{d}$ contains the title of j's Wikipedia article. This is, however, overly restrictive and results in low recall; e.g., we could not tell that a document that has the n-gram 'Maple Leaf Flag' implicitly contains the concept FLAG OF CANADA.

We therefore adopt a more robust approach, by leveraging Wikipedia links. The key observation is that the anchor texts used in Wikipedia to link to the article about j are in many cases synonyms of j, e.g., the anchor 'Maple Leaf Flag' links to the article about FLAG OF CANADA. However, the word 'flag', too, is used to link to this article, so simply accepting all anchors of an article j as names for it would result in low precision (since any flag might be referred to as 'flag'). To trade off precision against recall, we consider as names of j only anchors that link to j with high probability. Specifically, we accept a phrase a as a name for j only if its anchor likelihood $\Pr(\text{Target} = j | \text{Anchor} = a)$ is at least 30% (we chose this value by hand). Then, if the plain text of $\mathbf{d}$ contains phrase a we say that $\mathbf{d}$ talks about topic j and exclude j from our suggestion list. Note that, while this approach draws on Wikipedia, $\mathbf{d}$ need not necessarily be a Wikipedia article itself.

Let $\mathfrak{X}$ be the set of topic suggestions, i.e., the set of topics we do not exclude in the filtering step. It is interesting to note that its complement, $\mathfrak{T} \setminus \mathfrak{X}$, can be interpreted as the set of Wikipedia link suggestions. Recall that during keyphrase assignment we in fact augment $\mathbf{d}$ with links to Wikipedia (cf. Section 4.1). That is, a high-ranking suggestion $j \in \mathfrak{T} \setminus \mathfrak{X}$, which does have an anchor in $\mathbf{d}$, corresponds to a Wikipedia link that could be contained in $\mathbf{d}$ and whose absence is caused by noise. Consequently, with a modified filtering step, our method can be used for finding missing Wikipedia links, as done by West *et al.* [22].

5. TOPIC SUGGESTION FOR WIKIPEDIA ARTICLES

In this section we evaluate our topic suggestion algorithm quantitatively. Without querying human raters, this is rather difficult to achieve for general input documents, since judging the quality of topic suggestions is a highly subjective task. Instead, we attempt to get some insight into the performance of our method by constraining input documents to be Wikipedia articles and defining an automated evaluation heuristic based on the articles' edit history, such that the precision and recall of our method can be gauged on a large number of test documents. The results are presented in Sec-

tion 5.1. Since this automated evaluation has several shortcomings, we re-evaluate precision on a smaller test set by querying human raters. Those results are presented in Section 5.2.

In this first set of experiments, both the background corpus $\mathfrak{C}$ and all input documents $\mathbf{d}$ are Wikipedia articles. As a typical Wikipedia article $\mathbf{d}$ already contains links to other articles (which normally represent the keyphrases of $\mathbf{d}$; cf. Section 4.1), we need not run the keyphrase assignment step to obtain the document–term matrix $\mathbf{T}$ and the input document's topic vector $\mathbf{v}$. In principle, we may simply define $\mathbf{T}$ as Wikipedia's adjacency matrix. In practice, we compress its size by keeping as columns $\mathfrak{T}$ only the candidate topics as defined in Section 4.2.3, and as rows $\mathfrak{C}$ only a sample of 5,503 important articles (we follow West *et al.* [22] and choose the articles that are also included in the 2008/9 Wikipedia Selection for schools [23]).

We use the Wikipedia snapshot of March 6, 2009 [24]. In Section 4.2.3 we refer to the sparsity of $\mathbf{T}$. To express it in numbers, we note that, using this snapshot, only 2.7% of $\mathbf{T}$'s roughly 30 million entries are non-zero.

We set the eigenspace dimensionality to $K = 1{,}000$ in all experiments that use Wikipedia as a background corpus. This parameter has been hand-picked based on computational constraints and was not optimized via learning.

5.1 Automatically evaluating precision and recall

Given an input document $\mathbf{d}$, let the set $\mathfrak{X}_k$ contain the top k suggestions returned by our algorithm for $\mathbf{d}$, and let $\mathfrak{R}$ be the ground-truth set of relevant novel topics that are missing from $\mathbf{d}$. Then, precision at k is $|\mathfrak{X}_k \cap \mathfrak{R}|/|\mathfrak{X}_k|$, or the number of relevant suggestions divided by the number of suggestions, while recall at k is $|\mathfrak{X}_k \cap \mathfrak{R}|/|\mathfrak{R}|$, or the number of relevant suggestions divided by the number of relevant novel topics.

As mentioned above, the notion of relevance is highly subjective and cannot be defined in absolute terms [11], so precisely defining the set of relevant novel topics $\mathfrak{R}$ is not possible. Instead, we have to recur to a reasonable heuristic definition of $\mathfrak{R}$. Consider two versions of an input article $\mathbf{d}$, one from March 2009 (the time of the Wikipedia snapshot we are using), the other from April 2010 (i.e., about one year later). More often than not, human editors will have added several novel topics to the article during this one-year period. We make the assumption that editors mark important new topics by linking them to the respective Wikipedia articles, in accordance with Wikipedia's linking guidelines (cf. Section 4.1), and define the set of relevant novel topics $\mathfrak{R}$ as the set of links contained in the new but not in the old version of $\mathbf{d}$. Measuring precision and recall with respect to this ground truth, we can then estimate how well our algorithm matches the heuristically defined editing capabilities of human experts.

Since we are interested in novel *topics* (rather than merely novel links), we include in $\mathfrak{R}$ only links that correspond to topics which do not appear in the plain text of the old version of the input article. To determine whether this is the case, we take the link anchor–based approach described in Section 4.3.

In order to avoid the most obscure articles, we consider as test documents only articles with at least 100 incoming and 100 outgoing links. Also, to bound the size of $\mathfrak{R}$ and thus make precision and recall comparable across input documents, we consider only test documents to which editors added between 10 and 20 new links in the one-year period.

We use two evaluation sets. One comprises 100 'commonsense' articles. We chose this set with the human user evaluation which we present later in mind (cf. Section 5.2). The rationale is to facilitate the evaluation process by making sure raters have a basic understanding of the article for which they judge topic suggestions without having to read the article. To find these 100 articles, we presented lists of random article titles to 10 members of our group not involved in this research and asked each of them to select about 20 titles with the following property: 'Each selected title should represent a topic of which you have at least some basic knowledge. You don't have to be an expert in the topic, but you should have a rough idea what it is about.' This way, 213 titles were identified. Some of them were non-obvious, and manually sifting these, we kept 100 commonsense topics.

The other evaluation set consists of 1,000 articles. Unlike the set of 100 commonsense articles, these were randomly selected and can therefore serve for measuring the performance of our algorithm on typical Wikipedia articles.

The results are plotted in Figure 2, as functions of k, the number of top suggestions we return. First of all, note that the curves look rather similar, with the algorithm performing slightly better on the 100 commonsense articles than the 1,000 randomly selected articles. Recall increases superlinearly, which implies that at the top of our ranking, relevant suggestions are denser than further down, as desired. Recall at 1,000 is 26% for commonsense articles and 20% for arbitrary articles, i.e., the top 0.21% of the full ranking of all $N = 468{,}510$ candidate topics contain 26% or 20%, respectively, of the novel topics that were added as new links by human editors during the period of one year.

While these recall values seem satisfying, precision is rather low. Figure 2 plots two precision curves. For calculating *strict precision,* we count a suggestion j as relevant only if it is in $\mathfrak{R}$, i.e., if a link to the article about j was added to the input article in the one-year period. In practice, this is often too restrictive, since an editor might have added a mention of j to the article without also adding a link to the article about j. This is accounted for by *soft precision,* which counts the suggestion j as relevant if a mention of it was first added to the plain text of the input article within the one-year period. To decide if a document mentions a topic, we again use the link anchor–based method of Section 4.3. Figure 2 shows that the soft precision attained by our algorithm is somewhat higher than strict precision, as expected. (Both strict and soft precision are reported in their interpolated form [11].)

Although soft precision is clearly a more realistic metric than strict precision, it still does not appropriately capture the performance of our method. Consider, e.g., the top suggestions of our algorithm for the article about COMPUTER PROGRAMMING, listed in Table 1. Among the top five are topics such as TURING COMPLETENESS, COMPUTER SCIENCE, and CONTROL FLOW. While these topics are doubtless relevant, they are not counted as such because even the newer version of the article (after the one-year period) does not mention them yet. This is an inherent limitation of the evaluation paradigm that compares an old and a new version of an article: by defining ground truth based on the current version of an article, it assumes the latter to be perfect. Not only would this make suggesting further topics—and hence our work—pointless, it is also simply not the case.

Recall is affected in a similar way: often, human editors add novel links that do not correspond to relevant topics, or only marginally so. For instance, a link to the article about 1947 was added to the article about COMPUTER PROGRAMMING (cf. Table 1). Our method rightfully does not rank it amongst the top suggestions, yet this results in a decreased recall value.

5.2 Evaluating precision with human raters

Because of the shortcomings of the automated evaluation meth-

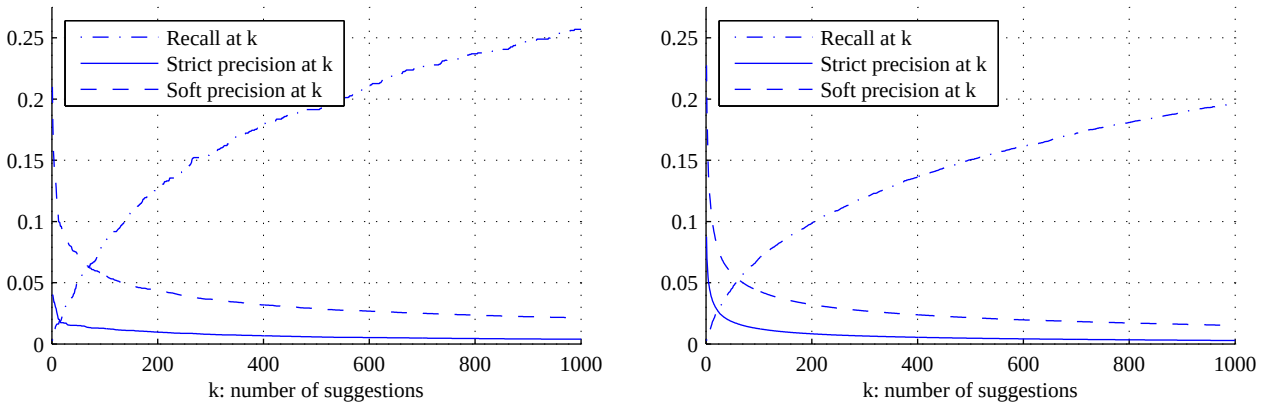

Figure 2: Performance of our topic suggestion algorithm on Wikipedia articles, where the ground-truth set of relevant novel topics is defined as the set of novel links added to the respective article by human editors within the period of one year. *Left:* Evaluation set of 100 commonsense articles. *Right:* Evaluation set of 1,000 randomly selected articles.

Human-added topics:	Automatic suggestions:
TROUBLESHOOTING	TURING COMPLETENESS
U.S. AIR FORCE	COMPUTER SCIENCE
ADACORE	* HIGH-LEVEL PROG. LANG.
PRINCIPLE OF LINGUISTIC RELATIVITY	JAVA (PROG. LANG.)
CARD STOCK	CONTROL FLOW
* HIGH-LEVEL PROG. LANG.	HASKELL (PROG. LANG.)
TEMPORARY FILE	UNIX
MEMORY LEAK	LISP (PROG. LANG.)
RACE CONDITION	RUBY (PROG. LANG.)
ERGONOMICS	TYPE SYSTEM
MAINTAINABILITY	SUBROUTINE
* SOFTWARE BUG	COMPARISON OF PROGRAMMING LANGUAGES
VULNERABILITY (COMPUTING)	DIFFERENCE ENGINE
* SCRIPTING LANGUAGE	PYTHON (PROG. LANG.)
MEASURING PROGRAMMING LANGUAGE POPULARITY	Z3 (COMPUTER)
1947	HALTING PROBLEM
THE ART OF COMPUTER PROG.	ABACUS
GERALD WEINBERG	UNIX-LIKE

Table 1: *Left:* The 18 novel links (i.e., topics) human editors added to the Wikipedia article about COMPUTER PROGRAMMING between March 2009 and April 2010. An asterisk signifies that the respective topic is among our algorithm's top 1,000 suggestions. *Right:* The 18 top suggestions of our algorithm.

odology described in the previous section, we found it necessary to complement these results with a human evaluation. Measuring recall remains elusive, since it would require defining $\mathfrak{R}$, the ground-truth set of relevant novel topics, for each single test document, which is practically prohibitive, and theoretically impossible due to the extremely subjective nature of the task. Precision, on the contrary, can be readily estimated by querying humans, since it does not require the *a-priori* definition of all of $\mathfrak{R}$; rather, we can have human raters assess relevance *ad hoc,* for each single suggestion.

In the experiments reported below, our goal is to compare the precision of our algorithm to that achieved by human Wikipedia editors. We define the set of human topic suggestions for an article **d** as the $\mathfrak{R}$ of Section 5.1, i.e., as the set of hyperlinks that were introduced to **d** during the period of one year and that do not correspond to topics already mentioned in the plain text of the old version of **d**. Suppose there are k such new links. In the set $\mathfrak{X}_k$

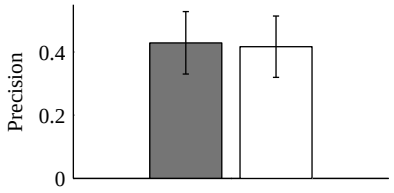

Figure 3: Average precision on the set of 100 commonsense articles. *Left:* Our topic suggestion algorithm. *Right:* Human Wikipedia editors. Error bars show 95% confidence intervals.

we pool the same number of top suggestions from the ranking produced by our algorithm. We then ask human raters to decide, for each suggestion in $\mathfrak{R}$ and $\mathfrak{X}_k$, whether it is relevant, and compute precision values for both sets of suggestions.

As our evaluation platform, we used Amazon Mechanical Turk [2]. Each evaluation task dealt with one input article **d** about a topic τ. The rater was asked to assess the relevance of all suggestions for **d**, human and automated, following these instructions:

> 'Below, you are given a list of topics. Your task is to decide, for each topic in the list, whether the Wikipedia article about τ should talk about it. If you are not familiar with what or who τ is, you can learn about it here.' [i.e., by following a link to τ's Wikipedia page]

This task description was followed by the topics from $\mathfrak{R} \cup \mathfrak{X}_k$, in randomized order to avoid any bias. As test documents we used the 100 commonsense articles introduced in Section 5.1, which contain each between 10 and 20 new human-added links, so the combined list of topic suggestions had at most 40 entries. To make the collected data more reliable, we protected the forms against Web bots using reCAPTCHA [19] and had the suggestions for every input article evaluated by five different raters. When interpreting the data, we say that a suggestion is relevant if a majority of the five raters (i.e., at least three) said so.

Figure 3 shows the average precision attained by our algorithm and by human editors, where the average is taken over the 100 test articles. The method we propose has an average precision of 43% (i.e., on average, 43% of the suggestions we make are assessed as relevant by a majority of raters), while human editors achieve 42%. The difference is not statistically significant at the $p < 0.05$ level

(determined via bootstrap resampling), which suggests that human Wikipedia editors and our method perform equally well at this task.

The precision of 43% attributed to our method under this evaluation methodology is an order of magnitude higher than the strict precision (at $k = 15$) of 3.5% shown in Figure 2 (we consider $k = 15$ because $k \in \{10, ..., 20\}$ in the human evaluation). Even soft precision as calculated there (10%) underestimates considerably.

These results confirm the observation of Section 5.1 that using the links, or topics, added by human editors as the ground truth of relevance is not a good heuristic. By definition, the ground truth should have perfect precision and recall. However, we show that precision is only 42%; also, the fact that many of our relevant suggestions lie outside the 'ground truth' implies that the latter has far from perfect recall.

The precision values reported in this section were obtained using the set of 100 commonsense articles. Referring back to the automated evaluation summarized in Figure 2, note that the strict precision curves for the 100 commonsense articles and the 1,000 random articles look virtually identical, while soft precision is slightly higher for commonsense articles. We conjecture that the actual precision (as measured in the human evaluation) scales analogously, being only slightly higher for commonsense than for random articles.

6. TOPIC SUGGESTION FOR PLAIN-TEXT DOCUMENTS

In the previous section we have demonstrated through a quantitative evaluation that our algorithm can match the performance of human editors when the input documents are Wikipedia articles. However, it is important to note that our algorithm is designed to work not only on Wikipedia articles but on arbitrary plain-text documents. We illustrate this in the present section.

A quantitative evaluation is considerably harder for arbitrary input documents than for Wikipedia articles: in order to be able to assess the relevance of link suggestions, human raters would have to read the entire input document first. In the case of Wikipedia articles, this is usually not necessary when testing on commonsense articles. For this reason, we provide a more qualitative analysis here, as follows: In Section 6.1, we show the output of our algorithm for a variety of input documents and comment on how the 'genre' of a document affects the quality of topic suggestions. While in all these examples we use Wikipedia as a background corpus, Section 6.2 demonstrates that the quality of suggestions can be improved by using a domain-specific background corpus. We use the U.S. Congressional Record to illustrate this point.

6.1 Using the Wikipedia corpus for different genres of input documents

Table 2 lists the top 10 topic suggestions for four input documents, which have been chosen because they belong to rather different genres. In the following, we comment briefly on each of these, in order to characterize qualitatively how our method can be expected to work on different types of documents.

Newspaper articles. The topic suggestions our algorithm makes for newspaper articles tend to be of high quality. In Table 2, we show the novel topics found for a *Time* magazine article about a currency crisis Europe is facing at the time of writing [4]. Our top suggestion, e.g., is EUROPEAN CENTRAL BANK, which is doubtless relevant, given that it is the institution in control of Europe's currency. The reason why newspaper articles are well suited as input documents is their factual style and high content in named entities, which makes them similar to Wikipedia articles. This has two consequences: First, Milne and Witten's method (cf. Section 4.1), since it is trained on Wikipedia, works as well on newswire stories as on Wikipedia articles [15], so the topic vector will be rich in meaningful keyphrases. Second, the generalization component can then augment this topic vector with high-quality suggestions because the input document lives in a part of document space that is densely populated with Wikipedia articles from the background corpus.

Frequently asked questions. Many websites contain sections with frequently asked questions. To identify the issues to be discussed there can be difficult, since recall and precision should be balanced: an FAQ section should answer all relevant questions, while it should also be short enough to not overwhelm readers. Automated topic suggestion can be helpful in this scenario, by offering the FAQ editor a ranked list in which he can find relevant topics through manual filtering. As a specific example, Table 2 shows the suggestions for the *W3C Semantic Web FAQ* [8]. While probably not all suggested topics should be mentioned in the FAQ section (in order to keep it concise), human editors might find some 'food for thought'; e.g., it might be useful to mention whether different OPERATING SYSTEMS offer different means to interact with the Semantic Web, or what impact the Semantic Web could have on GRAPHICAL USER INTERFACE design.

Philosophical essays. As mentioned above, an important criterion influencing the quality of topic suggestions is whether the character of the input document resembles that of Wikipedia articles. For many essays, especially those containing numerous technical terms, this is the case. Table 2 provides the example of Alan Turing's seminal 1950 paper *Computing Machinery and Intelligence* [17], in which he introduces what was to become later known as the Turing test. We conjecture that Turing would have appreciated our suggestions to contrast METAPHYSICS with the SCIENTIFIC METHOD in his musings about whether machines can think, and to complement his mention of Charles Babbage's invention of the digital computer with GOTTFRIED LEIBNIZ's discovery of the binary system. Of course, this should be taken with a grain of salt, since our Wikipedia background corpus contains many concepts that exist only *because of* Turing's influence and as such could not have been suggested 60 years ago, such as COMPUTER SCIENCE and ARTIFICIAL INTELLIGENCE. However, this is not the case for the other suggestions shown, which are due to the large number of scientific and philosophical references contained in Turing's paper and which could well have been made in 1950, had Wikipedia existed then.

Fictional texts. To characterize our method fully, it is important to also identify scenarios in which it fails. We found that this is frequently the case with fictional texts. More often than not, their purpose is to tell a story rather than to explain concepts, which sets them apart from Wikipedia articles and the aforementioned genres, on which our algorithm tends to work well. As an example, consider *Alice's Adventures in Wonderland* [5], for which we list the top 10 suggestions in Table 2. Keyphrase assignment results in a topic vector containing only seven non-zero entries (as opposed to 38 for the *Time* article and 50 for Turing's essay, in spite of these documents being significantly shorter), which do not summarize the story well: RABBIT HOLE, WILLIAM THE CONQUEROR, SOUP, TEA, RABBIT, PIG, IF I FELL (a Beatles song). The fact that most of our suggestions refer to British history is due to WILLIAM THE CONQUEROR, while the suggestion GESTATION (the carrying of babies in the female uterus) is caused by RABBIT and PIG.

In summary, our topic suggestion method works better the more the 'character' of the input document resembles that of Wikipedia articles. The most striking characteristic of Wikipedia articles,

Wikipedia corpus:	Congressional Record corpus:
LIST OF PRESIDENTS OF THE U.S.	PLAINTIFF
JURY	CLASS ACTION FAIRNESS ACT OF 2005
U.S. BILL OF RIGHTS	JUDICIARY
U.S. COURT OF APPEALS	U.S. DISTRICT COURT FOR THE MIDDLE DISTRICT OF FLORIDA
14TH AMENDMENT TO THE U.S. CONSTITUTION	JURY
LEGAL PERSON	WILL (LAW)
JURISDICTION	DUE PROCESS
SHARE (FINANCE)	TRIAL DE NOVO
ANDREW JOHNSON	TORT
U.S. SENATE COMMITTEE ON THE JUDICIARY	ADVANCE HEALTH CARE DIRECTIVE
	ATTORNEY GENERAL
LIMITED LIABILITY	JUDGE
LOUISIANA LAW	SUPREME COURT OF THE U.S.
LICENSE	STATE LAW
CUSTOM (LAW)	LIABILITY
FEDERALIST PAPERS	JURISDICTION
QIYAS	DAMAGES
GRISWOLD V. CONNECTICUT	FORUM SHOPPING
INNS OF COURT	U.S. COURT OF APPEALS FOR THE SECOND CIRCUIT
U.S. CAPITOL	
DAMAGES	PRODUCT LIABILITY

Table 3: The top 20 suggestions for a U.S. Congress speech during the debate on the Lawsuit Abuse Reduction Act of 2005 [1]. *Left:* Using Wikipedia as background corpus. *Right:* Using the Congressional Record as background corpus.

in turn, is their explanatory nature, which implies numerous references to other concepts and named entities. Keyphrase assignment works best in this setting, since the algorithm we use is trained on Wikipedia articles. The same holds for the generalization component if eigenarticles are computed using a background corpus of Wikipedia articles.

6.2 The Congressional Record corpus

In the previous section, we argued that our PCA-based generalization component works best for documents that resemble Wikipedia articles. It is important to note that this is only the case if we use Wikipedia, which constitutes a general-purpose background corpus, for computing eigenarticles. However, as stated in Section 4.2.1, any sufficiently large document collection can be employed for that purpose. If we use a domain-specific corpus, the eigenarticles found by PCA will be fine-tuned to the respective type of input documents, which will result in more meaningful topic suggestions than if we were to use Wikipedia as a generic background corpus. The goal of the current section is to illustrate this effect.

The background corpus we consider now is based on the U.S. Congressional Record and consists of all debates from the House of Representatives of 2005, compiled by Thomas *et al.* [16, 9]. We compute eigenarticles based on the 2,046 speeches contained in Thomas *et al.*'s test and training sets, where a speech is defined as the concatenation of all utterances a single speaker made in a single debate. We refer the reader to the original paper [16] for details concerning this corpus.

Table 3 lists the top suggestions for a speech given by Representative Mark Udall in October 2005 (available online [1]) during the debate on the Lawsuit Abuse Reduction Act, which was meant to prevent frivolous lawsuits. While the topics our algorithm suggests when using the general-purpose Wikipedia background corpus are generally from the realm of U.S. politics, with a bias towards jurisdiction, none of them is fully relevant. On the contrary, when the domain-specific Congressional Record corpus is used, suggestions are much more focused, and we find highly pertinent topics such as CLASS ACTION FAIRNESS ACT OF 2005 (another act aiming at reducing lawsuit abuse), FORUM SHOPPING (a common practice in lawsuit abuse), and PRODUCT LIABILITY (a common pretense for lawsuit abuse). Note that we did not include the debate containing the example speech in the eigenarticle computation. Rather, these useful suggestions are the result of the algorithm automatically generalizing from another debate, on the Class Action Fairness Act, that took place in Congress in February 2005, some months before Udall's speech.

The Congressional Record is better suited as a background corpus because the eigenarticles we compute from it are fine-tuned to the domain of the input document. To illustrate this, let us take a look at the eigenspace. Table 4 visualizes the mean topic vector and the first four eigenarticles of the Wikipedia corpus, while Table 5 does the same for the Congressional Record corpus. Each vector has as many entries as there are candidate topics, i.e., $N = 468{,}510$ in our case (cf. Section 4.2.3), but for clarity's sake we show only the 20 with the highest values. The mean topic vectors (at the far left of each table) give an impression of the most common themes across the entirety of the respective corpus: in Wikipedia these are mainly geographical regions and the two world wars, whereas in the Congressional Record the most common topics are the main U.S. political institutions and the more ubiquitous issues such as TERRORISM, TAXES, and HEALTH CARE.

The eigenarticles indicate directions in document space along which there is much deviation from the mean topic vector of the corpus. For instance, in the first eigenarticle of Wikipedia, dates prevail (most of them are not shown in Table 4), which is due to the fact that each year has a Wikipedia article listing important events and containing links to their dates, such that on the one hand, there are many articles containing many dates, but on the other hand, many other articles do not contain any dates at all, resulting in high variance with respect to the date content of articles. The subsequent eigenarticles are more interesting: the strongest entries of the second eigenarticle are from the realms of chemistry and atomic physics, while in the third, it is British and in the fourth, economic and political topics that prevail. By finding the major variations in the corpus, PCA effectively identifies its dominant semantic clusters.

The same effect can be observed in the Congressional Record corpus. Here, the first eigenarticle summarizes the stem cell controversy, the second and fourth are mostly about terrorism and anti-terror legislation, while in the third, budgetary topics weigh the heaviest.

Not only do the eigenarticles correspond to intuitive semantic classes, they also differ considerably between the corpora, which is reasonable and to be expected: on the one hand, articles about terrorism constitute only a small fraction of Wikipedia, on the other hand, debates about atomic physics are rare in Congress. Using the Wikipedia corpus amounts to injecting Wikipedia topics into the input document. While we have shown in Section 6.1 that this works in many cases, it might not always be appropriate. By using a domain-specific corpus, eigenarticles will be fine-tuned to the idiosyncrasies of the input genre, which in turn results in more precise topic suggestions. Therefore, the Congressional Record corpus would be most appropriate for a Congressman who wants to be sure to cover all relevant topics previous speakers have mentioned in similar contexts.

It should, however, be noted that in certain situations the less focused suggestions obtained when using a general-purpose corpus such as Wikipedia might in fact be desirable, e.g., if our Congressman intends to give his speech a fresh twist by introducing a topic that is related to the context but which previous speakers have not mentioned yet.

7. CONCLUSIONS AND FUTURE WORK

In this paper we propose an algorithm for suggesting novel topics to human authors. Given a plain-text document, our method leverages principal component analysis in order to find relevant new topics by generalizing from a large background corpus.

While we demonstrate the quality of topic suggestions for Wikipedia articles quantitatively, such an evaluation is inherently difficult for general input documents, due to the highly subjective nature of the task. Using examples from different genres of input documents, we therefore illustrate in a more qualitative way that many user groups beyond Wikipedia editors could profit from our system. We observe that our method works better on documents with factual contents than on fictional texts. However, a generalization-based algorithm would not be an appropriate tool for inspiring authors of fictional texts to begin with, since their objective is typically to surprise readers with entirely novel stories, not to write texts that are coherent with a background corpus.

In a typical use-case scenario, the human author would inspect the ranked output list of missing topics and decide which of these are actually worthwhile incorporating into the document. In principle, the algorithm can then be run again, on the modified input document. The effects of such a feedback loop remain to be investigated as part of our future work.

Beyond topic suggestion, eigenarticles have so far been used for finding missing Wikipedia links [22] and computing semantic relatedness [21]. In this paper, we demonstrate that the eigenarticles also cluster the topics of the background corpus semantically, thereby summarizing its most important contents. An interesting avenue for future research could therefore utilize eigenarticles for yet another purpose, as a tool for the exploratory analysis of text corpora.

8. ACKNOWLEDGMENTS

We acknowledge financial support by the Natural Sciences and Engineering Research Council of Canada (NSERC). We also thank David Milne for making WikipediaMiner public and Cosmin Păduraru for many fruitful discussions about this research.

9. REFERENCES

[1] *Congressional Record*, 151(1):H9318, 2005. http://frwebgate.access.gpo.gov/cgi-bin/getpage.cgi?dbname=2005_record&page=H9318&position=all (accessed May 26, 2010).

[2] Amazon. Mechanical Turk. Website, 2009. http://www.mturk.com (accessed May 20, 2010).

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[4] W. Boston. Germany tries to save the euro—all by itself. *Time*, May 21, 2010. http://www.time.com/time/world/article/0,8599,1991154,00.html (accessed May 26, 2010).

[5] L. Carroll. *Alice's Adventures in Wonderland*. Project Gutenberg, 2008.

[6] S. Fissaha Adafre and M. de Rijke. Discovering missing links in Wikipedia. In *Proc. 3rd International Workshop on Link Discovery (LinkKDD-05)*, 2005.

[7] B. Fortuna, D. Mladenič, and M. Grobelnik. Semi-automatic construction of topic ontologies. In *Semantics, Web and Mining: Joint Internat. Workshops EWMF/KDO-05*, 2006.

[8] I. Herman. W3C Semantic Web FAQ. Website, 2009. http://www.w3.org/2001/sw/SW-FAQ (accessed May 26, 2010).

[9] L. Lee. Convote dataset v1.1. Website, 2008. http://www.cs.cornell.edu/home/llee/data/convote.html (accessed May 13, 2010).

[10] A. Maguitman, D. Leake, and T. Reichherzer. Suggesting novel but related topics: Towards context-based support for knowledge model extension. In *Proc. 2005 International Conference on Intelligent User Interfaces (IUI-05)*, 2005.

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

[12] R. Mihalcea and A. Csomai. Wikify! Linking documents to encyclopedic knowledge. In *Proc. 16th ACM Conference on Information and Knowledge Management (CIKM-07)*, 2007.

[13] D. Milne. WikipediaMiner toolkit. Website, 2009. http://wikipedia-miner.sourceforge.net (accessed June 6, 2009).

[14] D. Milne and I. H. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proc. 1st AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI-08)*, 2008.

[15] D. Milne and I. H. Witten. Learning to link with Wikipedia. In *Proc. 17th ACM Conference on Information and Knowledge Management (CIKM-08)*, 2008.

[16] M. Thomas, B. Pang, and L. Lee. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP-06)*, 2006.

[17] A. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

[18] P. D. Turney. Coherent keyphrase extraction via Web mining. In *Proc. 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, 2003.

[19] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. reCAPTCHA: Human-based character recognition via Web security measures. *Science*, 321(5895):1465–1468, 2008.

[20] H.-C. Wang, D. Cosley, and S. R. Fussell. Idea Expander: Supporting group brainstorming with conversationally triggered visual thinking stimuli. In *Proc. 2010 ACM Conference on Computer Supported Cooperative Work (CSCW-10)*, 2010.

[21] R. West. Extracting semantic information from Wikipedia using human computation and dimensionality reduction. Master's thesis, McGill University, 2010.

[22] R. West, D. Precup, and J. Pineau. Completing Wikipedia's hyperlink structure through dimensionality reduction. In *Proc. 18th ACM Conference on Information and Knowledge Management (CIKM-09)*, 2009.

[23] Wikipedia. 2008/9 Wikipedia Selection for schools. Website, 2008. http://schools-wikipedia.org (accessed June 3, 2009).

[24] Wikipedia. Data dump of March 6, 2009. Website, 2009. http://download.wikimedia.org/enwiki/20090306 (accessed June 3, 2009).

[25] Wikipedia. Wikipedia:Linking. Website, 2010. http://en.wikipedia.org/w/index.php?title=Wikipedia:Linking&oldid=342061829 (accessed Feb. 5, 2010).

Time article about euro crisis [4]:	*W3C Semantic Web FAQ* [8]:	Alan Turing's *Computing Machinery and Intelligence* [17]:	Lewis Carroll's *Alice's Adventures in Wonderland* [5]:
EUROPEAN CENTRAL BANK	USENET	COMPUTER SCIENCE	HENRY I OF ENGLAND
INFLATION	OPERATING SYSTEM	ISAAC NEWTON	EDWARD THE CONFESSOR
ORGANISATION FOR ECONOMIC CO-OPERATION & DEVELOP.	GOOGLE	ARISTOTLE	HAROLD GODWINSON
	MICROSOFT	GOTTFRIED LEIBNIZ	GESTATION
EUROBAROMETER	MICROSOFT WINDOWS	BERTRAND RUSSELL	ANGEVIN EMPIRE
SINGLE MARKET	INTERNET EXPLORER	SCIENTIFIC METHOD	EDMUND IRONSIDE
GROSS DOMESTIC PRODUCT	GRAPHICAL USER INTERFACE	QUANTUM MECHANICS	HENRY II OF ENGLAND
EUROPEAN COMMISSION	JAVASCRIPT	GOTTLOB FREGE	TOSTIG GODWINSON
MOTION OF NO CONFIDENCE	INTERNET	ARTIFICIAL INTELLIGENCE	HARALD III OF NORWAY
BANK	MOZILLA FIREFOX	METAPHYSICS	BATTLE OF HASTINGS
BOREAL KINGDOM			

Table 2: Top 10 topic suggestions for different types of input documents, using Wikipedia as a background corpus.

Mean topic vector:		1st eigenarticle:		2nd eigenarticle:		3rd eigenarticle:		4th eigenarticle:	
EUROPE	0.66406	ROMAN NUMERALS	98.4044	OXYGEN	20.2236	LONDON	9.0715	GROSS DOMESTIC PRODUCT	26.1622
UNITED KINGDOM	0.65311	GREGORIAN CALENDAR	89.9053	CHEMICAL ELEMENT	19.9675	JULIAN CALENDAR	8.8904	UNITED NATIONS	25.4508
UNITED STATES	0.61414	NOBEL PRIZE IN PHYSIOLOGY OR MEDICINE	77.1151	HYDROGEN	19.0041	BBC	8.4054	PETROLEUM	18.0982
WORLD WAR II	0.59878			ATOMIC NUMBER	16.8553	PRIME MINISTER OF THE UK	7.8	AGRICULTURE	16.6884
FRANCE	0.5491	NOBEL PRIZE IN LITERATURE	76.6046	ISOTOPE	16.553	CHURCH OF ENGLAND	7.5718	INTERNATIONAL MONETARY FUND	16.5953
LATIN	0.54532	NOBEL PRIZE IN CHEMISTRY	75.7546	ION	15.1002	VICTORIA OF THE UK	6.5653		
INDIA	0.50131	NOBEL PRIZE IN PHYSICS	75.4313	HALF-LIFE	14.6232	ELIZABETH II OF THE UK	6.3286	HEAD OF STATE	15.6899
LONDON	0.50057	NOBEL PEACE PRIZE	74.2943	REDOX	14.2344	WALES	6.284	UNITED STATES DOLLAR	15.4465
ROMAN CATHOLIC CHURCH	0.48414	MARCH 4	56.7351	CARBON	14.1193	ARCHBISHOP OF CANTERBURY	6.1882	EUROPEAN UNION	15.1074
CHINA	0.48149	FEBRUARY 11	54.2642	ATOM	13.6386	WESTMINSTER ABBEY	5.8816	ATLANTIC OCEAN	15.0615
GERMANY	0.45495	MARCH 1	53.9667	ROMAN NUMERALS	13.3465	HOUSE OF COMMONS OF THE UK	5.8789	EXECUTIVE (GOVERNMENT)	14.9905
UNITED NATIONS	0.45283	JANUARY 12	50.5206	ELECTRON	13.2997	WILLIAM SHAKESPEARE	5.6846	PEOPLE'S REPUBLIC OF CHINA	14.4143
ENGLAND	0.44843	FEBRUARY 20	50.488	RADIOACTIVE DECAY	12.8377	HENRY VIII OF ENGLAND	5.667	PURCHASING POWER PARITY	13.6758
AFRICA	0.44297	MARCH 20	50.3921	ULTRAVIOLET	12.0777	JAMES I OF ENGLAND	5.5965	HEAD OF GOVERNMENT	13.6625
RUSSIA	0.41727	JANUARY 1	49.6162	HELIUM	11.5755	PARLIAMENT OF THE UK	5.548	WORLD BANK	13.1801
WORLD WAR I	0.41406	NOBEL MEMORIAL PRIZE IN ECONOMIC SCIENCES	49.3786	NITROGEN	11.2547	HOUSE OF LORDS	5.5463	UNITED STATES	12.9573
AUSTRALIA	0.41216			PROTON	10.6843	EDINBURGH	5.3916	CARBON DIOXIDE	12.8043
CHRISTIANITY	0.41038	DECEMBER 12	49.1927	EARTH'S ATMOSPHERE	10.5762	GREGORIAN CALENDAR	5.1994	REDOX	12.5982
JAPAN	0.40279	MARCH 17	48.9415	KELVIN	10.5551	CHARLES II OF ENGLAND	5.1038	CANADA	12.2193
ROMAN NUMERALS	0.40037	MAY 19	48.8819	MOLECULE	10.4533	THE TIMES	5.0592	TEMPERATENESS	11.9671
		JANUARY 22	48.8599					MAMMAL	11.9313
		JANUARY 27	48.7879						

Table 4: The mean topic vector m and the first four eigenarticles of the Wikipedia corpus. Each vector has $N = 468{,}510$ entries, of which we show only the 20 with the highest values. Notice that eigenarticles tend to cluster topics into semantic classes.

Mean topic vector:		1st eigenarticle:		2nd eigenarticle:		3rd eigenarticle:		4th eigenarticle:	
UNITED STATES DOLLAR	0.36552	STEM CELL	20.3218	TERRORISM	12.2331	DEFICIT	3.8416	USA PATRIOT ACT	9.2632
REPUBLICAN PARTY (U.S.)	0.36125	STEM CELL CONTROVERSY	20.2578	JUDICIARY	11.6066	DEBT	3.8224	FBI	7.0531
UNITED STATES CONGRESS	0.36007	EMBRYO	17.6271	SEPTEMBER 11 ATTACKS	11.3887	PENSION	3.3429	TELEPHONE TAPPING	6.4234
SPEAKER OF THE U.S. HOUSE OF REPRESENTATIVES	0.35505	EMBRYONIC STEM CELL	17.5909	USA PATRIOT ACT	10.2485	ENDANGERED SPECIES	2.8039	CIVIL LIBERTIES	5.5687
		CELL (BIOLOGY)	16.9893	FEDERAL BUREAU OF INVESTIGATION	10.0368	TAX CUT	2.8029	U.S. DEPARTMENT OF JUSTICE	5.5165
PRESIDENT OF THE U.S.	0.33407	DIABETES MELLITUS	16.1223	CIVIL LIBERTIES	9.7826	SMALL BUSINESS	2.6906	CRIME	5.4314
U.S. HOUSE OF REPRESENTATIVES	0.33181	ALZHEIMER'S DISEASE	13.7533	CONSTITUTION	9.073	DIVIDEND	2.6161	NATIONAL SECURITY LETTER	5.4101
		ADULT STEM CELL	13.2466	U.S. SENATE COMM. ON THE JUDIC.	8.7288	HEALTH INSURANCE	2.5915	TERRORISM	5.0347
UNITED STATES	0.33129	SPINAL CORD	12.1779	U.S. HOUSE COMM. ON THE JUDIC.	8.0508	MIDDLE CLASS	2.4234	SURVEILLANCE	4.698
CALIFORNIA	0.32783	NIH	11.8282	9/11 COMMISSION	7.4646	LOONEY TUNES: BACK IN ACTION	2.3635	JUDICIARY	4.5579
DEMOCRATIC PARTY (U.S.)	0.30961	IN VITRO FERTILISATION	11.6173	UNITED STATES FEDERAL COURTS	7.2932			PRIVACY	4.2309
TAX	0.30573	FERTILISATION	10.7157	TELEPHONE TAPPING	7.2334	TAX	2.2933	U.S. SENATE COMMITTEE ON THE JUDICIARY	4.1788
TERRORISM	0.28849	MEDICINE	10.6972	LAWYER	6.9399	BANKRUPTCY	2.2515		
FEDERAL GOVERNMENT OF THE U.S.	0.26726	STEM CELL RESEARCH ENHANCEMENT ACT	10.6216	CRIME	6.822	POVERTY	2.2482	SEPTEMBER 11 ATTACKS	3.9196
				U.S. DEPARTMENT OF JUSTICE	6.6145	ENDANGERED SPECIES ACT	2.2476	CONSTITUTION	3.9158
TEXAS	0.26575	CANCER	10.1784	IMMIGRATION	6.5851			CAPITAL PUNISHMENT	3.9096
UNITED STATES SENATE	0.26006	IN VITRO	10.1764	UNITED STATES SENATE	6.5612	GOVERNMENT DEBT	2.1609	MURDER	3.7962
GEORGE W. BUSH	0.25953	BIOMEDICAL RESEARCH	10.0144	NATIONAL SECURITY LETTER	6.5099	MEDICAID	2.0413	JUDICIAL REVIEW	3.7399
HEALTH CARE	0.25026	CELL CULTURE	9.8672	JUDICIAL REVIEW	6.4624	PENSION BENEFIT GUARANTY CORP.	2.0331	U.S. HOUSE COMMITTEE ON THE JUDICIARY	3.6918
SEPTEMBER 11 ATTACKS	0.24217	CLONING	9.8331	U.S. DEPT. OF HOMELAND SECURITY	6.398				
IRAQ	0.24039	UMBILICAL CORD	9.5242			HEALTH CARE	2.0158	JUDGE	3.5834
OIL	0.23858					INSURANCE	1.9984	FOREIGN INTELLIGENCE SURVEILLANCE ACT	3.5527
BIPARTISANSHIP	0.23552					ECONOMIC GROWTH	1.9464		

Table 5: The mean topic vector m and the first four eigenarticles of the U.S. Congressional Record corpus. As in Table 4, we show only the 20 entries with the highest values for each vector. Again, eigenarticles cluster topics into semantic classes.