Universidade Federal do Amazonas
Instituto de Computação
Programa de Pós-Graduação em Informática

# Métodos para Seleção de Palavras-Chave em Sistemas de Publicidade Contextual

Klessius Renato Berlt

Manaus – Amazonas
Dezembro de 2012

Klessius Renato Berlt

# Métodos para Seleção de Palavras-Chave em Sistemas de Publicidade Contextual

Tese apresentada ao Programa de Pós-Graduação em Informática do Departamento de Ciência da Computação da Universidade Federal do Amazonas, como requisito parcial para obtenção do Título de Doutor em Informática. Área de concentração: Recuperação de Informação e Banco de Dados.

Advisor: Prof. Edleno Silva de Moura, Doutor
Co-Advisor: Prof. Marco Antonio Cristo, Doutor

Klessius Renato Berlt

# Métodos para Seleção de Palavras-Chave em Sistemas de Publicidade Contextual

Tese apresentada ao Programa de Pós-Graduação em Informática do Departamento de Ciência da Computação da Universidade Federal do Amazonas, como requisito parcial para obtenção do Título de Doutor em Informática. Área de concentração: Recuperação de Informação e Banco de Dados.

Banca Examinadora

Prof. Dr. Edleno Silva de Moura – Orientador
Instituto de Computação – UFAM/PPGI

Prof. Dr. Marco Antonio Cristo – Co-Orientador
Instituto de Computação – UFAM/PPGI

Prof. Dr. Altigran Soares da Silva
Instituto de Computação – UFAM/PPGI

Prof. Dr. Marcus Fontoura
Research Scientist at Google Inc.

Prof. Dr. Pavel Calado
Instituto Superior Técnico – INESC-ID

Manaus – Amazonas
Dezembro de 2012

*"What is now proved was once only imagined."*

*William Blake.*

# Agradecimentos

Agradeço aos meus pais, base de toda a minha educação. Sem os quais eu jamais teria iniciado esta jornada e que, mesmo nos momentos de dificuldade me apoiaram irrestritamente. Essa tese é dedicada a eles.

À minha irmã que sempre foi companheira e não só compartilhou os momentos de alegria como me ajudou a superar os momentos de tristeza pelos quais passamos juntos nas nossas vidas.

Agradeço à Rebeca, meu amor, que esteve ao meu lado sempre com muita compreensão, ajudando a adoçar os momentos mais difíceis com suas palavras de conforto.

Ao meu orientador, Edleno Silva de Moura, por tantos anos de parceria, que foi sem dúvida alguma o maior responsável por ter alimentado esse meu interesse pela ciência e me encorajado a embarcar nessa aventura que é a vida acadêmica.

Ao meu co-orientador Marco Cristo e aos professores Altigran Soares da Silva e João Marcos Cavalcanti, sempre presentes nas várias etapas deste processo me auxiliando com sua experiência. Também a Maisa Leão, Alessandro Sena, Mauro Rojas, Marcela Pessoa, Eli Cortez, David Fernandes, Thierson Couto, Nivio Ziviani e todas as outras pessoas que participaram de forma direta ou indireta dos trabalhos que me ajudaram na conclusão desta tese.

Aos meus companheiros de BDRI e de UFAM em geral, que muito colaboraram criando um ambiente muito amigável e propício para que o trabalho fosse desenvolvido.

Aos meus amigos, Glad, Perna, Serjão, Beto, George, Júlio, Karane, Leandrinho, PP, Papa, Sassa, Theo, Bobinho e Biro. Sempre presentes nos momentos de descontração muito necessários para tornar este processo mais ameno e saudável.

Ao Universon Online, Neemu, Suframa e CAPES, instituições comprometidas com a pesquisa e que muito fizeram para viabilizar a conclusão deste trabalho.

E a todas as outras pessoas que de alguma maneira participaram deste processo e me ajudaram a atingir esse objetivo.

# Resumo

Neste trabalho, nós estudamos o problema de seleção de palavras-chave para sistemas de publicidade contextualizada em dois diferentes cenários: páginas web e textos curtos.

Nós lidamos com o problema de seleção de palavras-chave em páginas web utilizando aprendizado de máquina. Abordagens tradicionais baseadas em aprendizado de máquina geralmente possuem como objetivo selecionar palavras-chave consideradas como relevantes por um conjunto de usuários. Entretanto, a nova estratégia proposta nesse trabalho objetiva selecionar palavras-chave que gerem o melhor resultado na qualidade final do sistema de seleção de publicidade. A esta estratégia, nós demos o nome de *ad collection aware* keyword selection (também chamada de *ACAKS*).

Esta nova abordagem baseia-se no julgamento dos usuário em relação às propagandas com as quais cada palavra-chave é relacionada pelo sistema de seleção de publicidade. Apesar desta estratégia demandar um alto esforço para rotular o conjunto de treino em relação às abordagens tradicionais, nós acreditamos que o ganho obtido em revocação é suficiente para fazer com que o *ACAKS* seja uma melhor alternativa.

Nos experimentos que nós realizamos com uma coleção de anúncios e considerando as características propostas em um trabalho anterior, nós descobrimos que a nova abordagem proposta levou a um ganho de 62% em revocação em relação ao baseline utilizado sem perder precisão. Além desta nova alternativa para selecionar palavras-chave, nós estudamos ainda a utilização do conjunto de características estraída da coleção de anúncios para selecionar palavras-chave.

Nós também apresentamos três novos métodos para extrair palavras-chave de páginas web que não necessitam de treino e usam a Wikipedia como fonte externa de informação. A informação usada da Wikipedia inclui os títulos dos artigos, co-ocorrência de palavras-chave e categorias associadas com cada artigo da Wikipedia.

Resultados experimentais mostram que nossos métodos são soluções competitivas para selecionar boas palavras-chave que representem bem o conteúdo de páginas web, enquanto se mantém simples e eficientes.

Além da seleção de palavras-chave de páginas web nós também estudamos métodos para selecionar palavras-chave em textos curtos. Textos curtos tem se tornado uma maneira muito popular que os usuários encontraram para publicar conteúdo na web. Todos os dias, milhões de usuários postam seus pensamentos, necessidades e sentimentos na web através de sistemas de redes sociais, como Facebook e Twitter, ou espaços para comentários em sites de notícias. Grande parte da renda destes sistemas é proveniente de publicidade contextualizada, desta forma selecionar palavras-chave neste novo cenário surge como um novo desafio.

Nós propomos e estudamos uma nova família de métodos que utiliza a informação de conectividade presente na Wikipedia para descobrir os conceitos mais relacionados em cada texto curto. Utilizamos também os métodos propostos como um novo conjunto de características em um Framework de aprendizado de máquina para melhorar a qualidade dos resultados obtidos. Nós mostramos que esta abordagem apresenta um bom desempenho e supera o melhor baseline em cerca de 35%.

Finalmente, nós aplicamos a abordagem *ACAKS* em textos curtos e ele gerou bons resultados, superando uma abordagem tradicional baseada em aprendizado de máquina em cerca de 80% tanto em termos de precisão quanto revocação.

Palavras Chaves: Seleção de Palavras-Chave, Aprendizado de Máquina, Publicidade Contextualizada.

# Abstract

In this work we address the problem of selecting keywords for contextual advertising systems in two different scenarios: web pages and short texts.

We deal with the problem of selecting keywords from web pages using machine learning. While traditional machine learning approaches usually have the goal of selecting keywords considered as good by humans. The new machine learning strategy proposed drives the selection by the expected impact of the keyword in the final quality of the ad placement system, which we name here as *ad collection aware* keyword selection (also referred in this work as *ACAKS*).

This new approach relies on the judgement of the users about the ads each keyword can retrieve. Although this strategy requires a higher effort to build the training set than previous approaches, we believe the gain obtained in recall is worth enough to make the *ad collection aware* approach a better choice.

In experiments we performed with an ad collection and considering features proposed in a previous work, we found that the new *ad collection aware* approach led to a gain of 62% in recall over the baseline without dropping the precision values. Besides the new alternative to select keywords, we also study the use of features extracted from the ad collection in the task of selecting keywords.

We also present three new methods to extract keywords from web pages which require no learning process and use Wikipedia as an external source of information to support the keyword selection. The information used from Wikipedia includes the titles of articles, co-occurrence of keywords and categories associated with each Wikipedia definition.

Experimental results show that our methods are quite competitive solutions for the task of selecting good keywords to represent target web pages, albeit being simple, effective and time efficient.

Besides selecting keywords from web pages we also study methods for selecting keywords from short texts. Short texts have became a very popular way users adopt for publishing content on the web. Every day, millions of users post their thoughts, needs and feelings on the Web through systems, such as social networks like Facebook and Twitter, or spaces for comments on news web sites. Much of these systems' revenue is from contextual advertising systems, thus selecting keywords in this new scenario raise as a new challenge.

We propose and study a novel family of methods which uses the connectivity information present on Wikipedia to discover the most related concepts on each short textual unit. We also used the proposed methods as a new set of features on a Machine Learning Framework to boost the quality of the results obtained. We show that this approach presents a good performance and outperforms the best baselines by more than 35%.

Finally, we apply the *ACAKS* approach on short texts and it yielded good results, outperforming a traditional machine learning approach by more than 80% in precision and 80% in recall.

Keywords: Keyword Selection, Contextual Advertising.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The Internet has become a very effective medium for advertising by allowing the development and application of new advertising options which support interactive, targeted and persuasive communication. In fact, the entire market has changed driven by new behaviours from consumers whom, in their quest for novelties, information and entertainment, are quickly moving from traditional media, such as TV and newspaper, to digital alternatives, all converging to the Internet.

While for the publishers these advertisements represent a way to monetize their web services, for the advertisers, it represents the possibility of global exposition at low cost, with large potential for direct measurement of results and interaction with consumers. Such characteristics, along with a growing audience, motivates us to study the widespread adoption of web advertising.

One of the most successful model of advertising on the web is Contextual Advertising, widespread in the recent years due to its success in generating revenue to a large variety of web services, ranging from microblogs to large web portals. This form of advertising works by automatically associating ads with the content the user is currently viewing. The general assumption is that ads that are semantically correlated to the content of the document the user is browsing might have a higher chance to grab his attention, since he is probably interested in the subject of the document in the first place. The main challenge in the task of retrieving contextual ads is to accurately and quickly find ads that are more semantically related to web pages in a set of thousands (or even millions) ads from a big variety of products and services.

Traditional contextual advertising systems usually are divided in two phases. The first phase consists of extracting the contextual information from the web resource which will be associated with an ad, for instance a post in a microblog or a web page. It can be done, for example, by selecting the keywords that best summarize its content. On the second phase, this contextual information is used to obtain a ranking of the ads that will be displayed to users. This thesis focus on studying techniques to improve the quality of the results of the first phase.

A naive approach to determine the topic of a web resource would be using its whole textual content. However, the entire textual content can be a very noisy source of information, since there are several terms that do not have a direct relation with its main subject and using these terms could decrease the quality of the advertising systems while increasing the communication and latency costs [1]. An interesting alternative to solve this problem is the adoption of algorithms to estimate the importance of the keywords in the text to be processed. Such estimations are then used as input for other algorithms used to rank the ads.

The task of determining the keywords of a web resource may vary according to its characteristics. We here divide the study of selecting keywords into two main classes. First we present studies about the selection of keywords from web pages, presenting two alternative methods for this task. Then, we consider the problem of selecting keywords from web resources where users post short texts, such as microblogs, and spaces for user comments on web portals.

A simple strategy to determine the most important keywords of a given web page could be to use classical information retrieval statistics such as term frequency and inverse of document frequency [2, 32]. However, there are many other characteristics, such as the position of the term on the web page and the occurrence of HTML tags, which could help on this task. Previous work have used these features to describe the terms of a web page based on a machine learning framework [40]. We propose here a novel approach called *ACAKS*, which guides the training phase of a machine learning classifier on selecting the keywords that can retrieve the best ads on an advertising system. Our intuition is that the keywords selected by the users as the best for advertising are not always the most appropriate for contextual advertising systems. We show that this approach achieves a recall more than 60% better than the method proposed on [40] with a similar precision. We present *ACAKS* in Chapter 4.

The usage of a learning method presents the drawback of requiring an extra training effort to fit the method to the ad collection. Further, in practical applications the ad collections change fast, which thus would require the learning process to be continually applied. Given these restrictions, we also investigated alternative for selecting keywords without using a learning method. When looking to the literature, a successful alternative adopted by several previous research efforts [25, 23, 38, 20, 15, 7, 16, 17] is to use external sources of knowledge to predict the importance of each keyword.

In this thesis, we propose and study methods for using Wikipedia as this extra source of information. Wikipedia is a free online encyclopedia created collectively by volunteers considered as one of the largest knowledge bases available online. The open nature of the editing process in Wikipedia makes it a very dynamic and fresh source of information. As an example, Wikipedia is likely to have complete information on new products that draw public attention, such as laptop models, game consoles or recently published books. This is an important characteristic in contextual advertising systems, since the user context and their interests also have a dynamic nature, that is, they change over months, weeks, and even during the same day. We show that a simple approach using the textual content of the Wikipedia articles and the categories they belong to may do very well on selecting keywords from web pages. We present the results obtained with this strategy in Chapter 4.2

Besides traditional web pages, short texts have become a very popular alternative adopted by users for publishing their thoughts, needs, opinions and feeling on the web. Social networks like Facebook and Twitter have millions of new posts every day. This type of information can also be used to determine the context the user is inserted into and then to boost the performance of the advertising systems. However, as occurs on web pages, the complete content of a short text may carry noisy terms which can deteriorate the performance of the whole advertisement system. Selecting the best keywords on such scenario is a crucial step on a good advertising system.

To reach this objective we also studied methods for selecting keywords from short texts. Previous work have shown that in this scenario the usage of an external source of contextual information, such as the Wikipedia, is again a promising alternative for selecting keywords [25, 38]. We here follow this strategy and propose a graph-based approach to find keywords using connectivity information from Wikipedia. The proposed methods are based on the assumption that if a set of keywords appears together in a text and their correspondent Wikipedia articles are linked to each other, these keywords must be related to the main topic of the text.

We compare our method to the best alternative we have found in literature to find keywords on short texts, presenting variations using a machine learning technique and also options that do not adopt a learning process. The experiments presented indicate our approach is a competitive alternative for finding keywords on short texts. Further, we also investigate the performance of our best method to find keywords on short text when applied to a contextual advertising application. In the experiments we include an alternative which combines our graph approach with the *ACAKS* keyword selection strategy. These experiments are useful to check whether the advantages of applying the *ACAKS* strategy also holds in the scenario of short texts.The methods for selecting keywords from short texts and the results obtained with them are presented in Chapter 5.

## 1.1 Contributions

The main contributions of this thesis are the proposal and study of new alternative methods for selecting keywords from web pages and short texts. Among the specific contributions we list:

- A new graph-based approach for taking advantage of information from Wikipedia to select keywords from short texts. We present several methods for finding keywords on short texts based on the new approach presented. The proposed methods can be applied using machine learning and also without requiring a learning process;
- A new machine learning oriented method for selecting good advertising keywords from texts, named as *ACAKS*, which is able to improve the recall of results while maintaining the precision levels;
- A new method that adopts Wikipedia as an external source of information to select keywords from web pages;
- A new collection for evaluating methods for selecting keywords from short texts. The collection contains posts extracted from the microblog Twitter and a list of possible keywords that can be found from them.

## 1.2 Organization

The remaining context of this thesis is organized as follows: Chapter 2 presents the related work. Chapter 3 presents basic concepts necessary to better understand the study presented in this thesis. Chapter 4 presents the alternatives we have proposed and studied to select keywords from web pages. Chapter 5 presents the methods for selecting keywords from short texts. Finally, Chapter 6 presents the conclusions and future work.

# Chapter 2

# Related Work

In this Chapter we revise the related work associated to the task of selecting keywords from web resources, focusing on previous work related to the specific problems of selecting keywords from web pages and also from short texts.

## 2.1 Selecting Keywords in Web Pages

One of the most common strategies adopted for selecting keywords from texts is to apply a selection based on the frequency of terms in the text (TF) and on how rare those terms are in a corpus, which is expressed by the Inverse Document Frequency (IDF) [31]. The idea is to select the keywords by ranking the terms found in a text according to the product of TF by IDF. We name this strategy as the TFIDF selection method. While this approach seems to be naive, it produces quite good results in practice, and its simplicity turns it into a reasonable strategy even nowadays. TFIDF approach is effective in case of texts extracted from web pages, but it does not work well when selecting keywords from short texts, where the TF values are usually one for all terms and thus measuring the frequency of terms does not make much sense. Also, choosing the keywords by their IDF would be the same as selecting only the most rare expressions without taking into account the context and not providing good results.

Although the TFIDF strategy achieves good results when selecting terms from web pages, with the popularization of machine learning methods, other more sophisticated and effective solutions were proposed. Among them, we can highlight Kea [38] and Genex [22]. In Kea, the product of the TF and the IDF of a term was adopted as a feature along with the position where it appears first in the text. Then, a Naive Bayes Classifier was trained to predict which terms better describe the content of a text. In Genex, a more descriptive set of features, including syntactic and statistical features, was adopted as input to a genetic algorithm framework to find keywords.

Other strategies using machine learning algorithms were proposed by Goodman and Carvalho [14] and Yih et al [40]. They developed a method for determining keywords to be used to place ads in emails and web pages. In these two research efforts, authors adopted logistic regression to learn good keywords for advertising. They studied a large number of features to determine the importance of each term present in a text, and thus selected the most important ones as the keywords to represent the text.

Among the features considered by them, we cite the frequency of each keyword candidate, its rareness in the collection, the content section where it occurs (for instance, metadata section, title, etc) and its presence in search query logs. From their empirical study, they found that the presence of keywords in the query log was the most important feature to determine the importance of a keyword. They also concluded that other features, when taken into combination, were also useful to determine the best keywords. While the focus was to select keywords for advertising, both articles did not present experiments to evaluate the impact of the proposed methods in terms of precision and recall when retrieving ads.

The idea presented by Wu and Bolivar [39] is very similar to the one proposed by [40]. However, their focus was to select good keywords for displaying ads from ebay[1], and they take advantage of a set of features derived from proprietary data obtained from this web site.

[1] http://www.ebay.com

In another research line, several authors have exploited the idea of taking advantage of external sources of knowledge to boost the performance of keyword selection methods. With the dissemination of Wikipedia as an external source of information for several tasks, the idea that such kind of resource could improve the quality of the keywords chosen and even, in some cases, provide a link to a more detailed description about their meaning, became very attractive. Therefore a large number of studies on how to carry this out have appeared.

A first example of method that exploits the Wikipedia to detect keywords was proposed by Mihalcea et. al [25], where authors presented a method named as *keyphraseness*. Their method achieved good results estimating the probability that a phrase, sequence of consecutive terms, is a keyword by calculating how many times the phrase was linked to other Wikipedia articles. This estimative was then used to link keywords to Wikipedia articles. This method used the probability of a term to be selected as a keyword in a new document. The method considered as keywords the terms that are linked to other Wikipedia articles, i.e., it considered link identification and keyword extraction as the same problem.

One of the most successful approaches that follows the research line of using Wikipedia as an external source of information to detect keywords is presented by Maui [23]. Authors presented a two-stage approach for automatic tagging documents that enhanced the Kea's [38] machine learning framework by including semantic knowledge retrieved from Wikipedia. It first selected a list of candidate keywords and then filtered them using bagged trees on which *keyphraseness* was adopted as a feature. Also, the work presented by Kondo et. al [20] used the Wikipedia to discover which keywords better matched the interests of a user by analysing the user's web browsing history.

As proposed in the method *keyphraseness*, several other authors have also adopted strategies that use Wikipedia information to build a graph to represent the keywords of a given text. Previous research, such as the ones presented by Grineva et. al [15] and Course and Mihalcea [7] indicated this strategy may be used to derive high quality keyword selection methods. The article by Grineva et. al [15] presented a method to extract keywords from multi theme documents. This method uses a graph where the keywords from the text are vertexes and a semantic relatedness measure was used to weight the edge between them. Thus, a sophisticated algorithm [37] was used to find communities in this graph.

The work by Course and Mihalcea [7], identified topics on documents using the whole Wikipedia graph. For this, they adopted a modified version of the Pagerank algorithm [16] biased towards the keywords present in the original document.

Authors in [17] devised another method of Wikipedia semantic relatedness that associate texts to Wikipedia categories. They experimented with two matching techniques: exact matching and relatedness matching. The first performed exact string matching, which is very efficient, while the second used the cosine measure to select keywords. Interestingly, exact matching has performed better in most cases, showing that it represented the best alternative to match textual documents and keywords.

As using Wikipedia as an external source of information has proven as a good strategy to select keywords from a given document and also the performance of the machine learning approaches also achieved good results, some works trying to take advantage of both the approaches have appeared. Among them, we call attention to these ones described below.

In the paper by Milne et. al [26] the authors used as features the number of times an article was linked from a term in Wikipedia and the similarity between Wikipedia articles. Recently, another work that used Wikipedia information to extract keywords is [20]. The authors used the Wikipedia link structure and a variation of the HITS algorithm to rank keywords in a Web page.

ESA (Explicit Semantic Analysis) [13] had the objective of semantically enriching texts in natural language. It represented texts as vectors of Wikipedia keywords, associating a weight to each keyword quantifying the relatedness between the text and the keyword. They experimented with word-level and text-level semantic relatedness, obtaining better results than other approaches in the literature. Furthermore, they also experimented the effectiveness of their semantic relatedness measure for generating features for classification.

## 2.2 Selecting Keywords from Short Texts

In spite of such a large number of researchers seeking to find the key terms of a document, there was little effort placed on mining the main keywords in short pieces of text. The raise of social networks has drawn some attention to this area, where we can cite [28] who proposed an approach to identify the main topic in social media posts combining NLP, tag-based and semantic-based techniques. Also, the work on [41] to select keywords in short web pages presented good results. However, this approach, besides dealing with short texts, relies on clues such as information about an advertisement set and HTML tags to predict the most important keywords on the text. Such types of clues are usually not available in most of the short texts sources we found nowadays on the Internet.

Some methods have risen with the purpose of automatic annotating the main concepts in fragments of text. Among them we highlight Spotlight[2], which automatically annotates mentions of DBpedia resources in fragments of text. Another interesting work, is Tagme [10], which use anchor texts from Wikipedia to annotate plain texts and create links to Wikipedia pages.

Recently, Meij [24] et al proposed an approach to add semantics to microblog posts. The authors studied methods to identify the keywords semantically related to the posts and link them to their corresponding Wikipedia articles. The best results were achieved using an approach referred to as *Commonness*, which ranks each keyword based on the relative frequency that the n-grams present in the post are used as an anchor text for that keyword. Then they improved the Commonness results by modelling each keyword as a set of features and applying a Random Forest Classifier to refine the results. This work is used as baseline for comparison with the methods we propose.

[2] http://spotlight.dbpedia.org/

# Chapter 3

# Basic Concepts

This Chapter introduces basic concepts required for a better understanding of our proposed methods. We start by describing what is an ad and its main parts in Section 3.1. In Section 3.2 we describe the Vector Space Model, a classical Information Retrieval model that is adopted as part or as a reference to methods studied here. The most important metrics adopted to evaluate the quality of the methods proposed in this work are presented on Section 3.4.

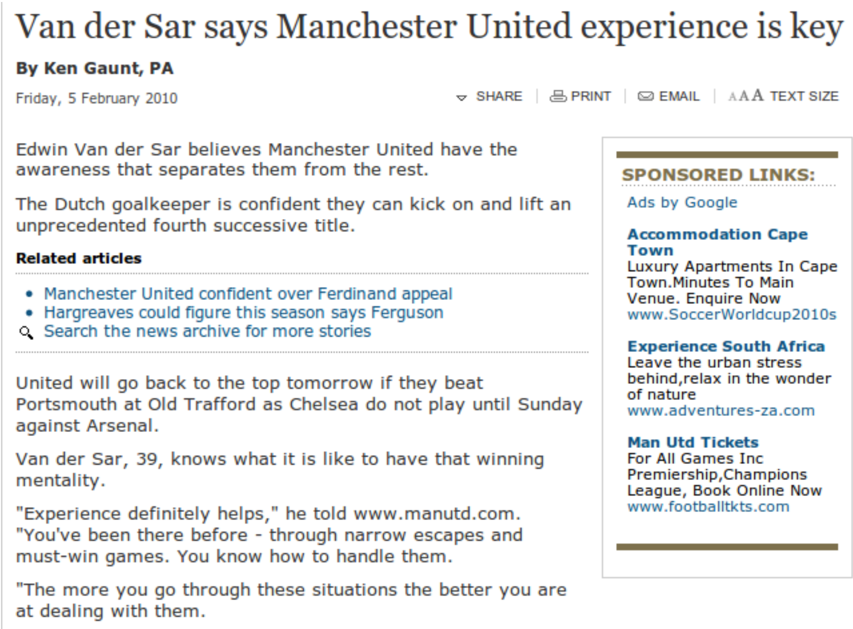
## 3.1 Contextual Advertising

Contextual Advertising is a form of online advertising where the ads displayed on a web service (like a web page) are related to its content (for example, displaying ads from a pet shop on a web page about dogs). The success of such kind of advertising can be attributed to assumption that if the user is interested in the content of the web service, he will be also interested in an ad on the same context.

In this work we consider that an ad is composed of three structural parts: a title, a textual description and a set of keywords. In fact, these are the usual components of an ad in online advertising systems and compose which is called the *ad creative*. Some works may also consider the content of the page the user is directed to when clicking on the ad (a.k.a. landing page) as another part of the ad, but due to the high costs of including such content and low benefit provided by it, in this work we do not consider it as a part of the ad creative.

Further, an advertiser can associate several ads with the same product or service. We refer to such group of ads as a *campaign*. Note that only an ad per campaign should be placed in a web page in order to ensure a fair use of the page advertising space and increase the likelihood that the user will find an interesting ad.

Figure 3.1 illustrates an example of three contextual ads on the right side of a web page. For the ad in the first ad slot, the title is “Accommodation Cape Town”, the description is “Luxury Apartments in Cape Town. Minutes To Main Venue. Enquire Now.”, and the hyperlink points to the site “www.SoccerWorldCup2010s.com”. Some example of keywords related to this ad are ”soccer” and ”world cup”.



Van der Sar says Manchester United experience is key

**By Ken Gaunt, PA**

Friday, 5 February 2010

SHARE | PRINT | EMAIL | A A A TEXT SIZE

Edwin Van der Sar believes Manchester United have the awareness that separates them from the rest.

The Dutch goalkeeper is confident they can kick on and lift an unprecedented fourth successive title.

**Related articles**

- Manchester United confident over Ferdinand appeal
- Hargreaves could figure this season says Ferguson
- Search the news archive for more stories

United will go back to the top tomorrow if they beat Portsmouth at Old Trafford as Chelsea do not play until Sunday against Arsenal.

Van der Sar, 39, knows what it is like to have that winning mentality.

"Experience definitely helps," he told www.manutd.com. "You've been there before - through narrow escapes and must-win games. You know how to handle them.

"The more you go through these situations the better you are at dealing with them.

SPONSORED LINKS:

Ads by Google

**Accommodation Cape Town**
Luxury Apartments In Cape Town.Minutes To Main Venue. Enquire Now
www.SoccerWorldcup2010s

**Experience South Africa**
Leave the urban stress behind,relax in the wonder of nature
www.adventures-za.com

**Man Utd Tickets**
For All Games Inc Premiership,Champions League, Book Online Now
www.footballtkts.com

Figure 3.1: Example of contextual advertising in the page of an England newspaper that offers tickets to soccer games, accommodations to the World Cup and tourism in South Africa, where the 2010 World Cup will take place. The content of the page is about soccer.

## 3.2 Vector Space Model

The Vector Space Model is an algebraic model for representing text documents as vectors of identifiers [33, 34, 35]. It represents documents and queries as vectors in a $T$-dimensional Euclidean space, where $T$ is the number of distinct index terms in the document collection.

$$\begin{aligned}\vec{d_j} &= (w_{1,j}, w_{2,j}, w_{3,j}, \ldots, w_{T,j}) \\ \vec{q} &= (w_{1,q}, w_{2,q}, w_{3,q}, \ldots, w_{T,q}) \qquad (3.1)\end{aligned}$$

where $w_{t,j}$ is the weight of the term $t$ in the document $d_j$, and $w_{t,q}$ the weight of $t$ in the query $q$.

The term weights in the document vectors are given by two parameters: (i) $tf(t, j)$, computed as the number of times that the term $t$ occurs in a document $d_j$, and (ii) $idf(t)$, that is a function of the number of documents where the term $t$ occurs. Thus, the weight of an index term $t$ in a document $d_j$ is given by:

$$w_{t,j} = tf(t, j) \times idf(t) = tf(t, j) \times \log \frac{N}{n_t} \qquad (3.2)$$

where $N$ is the total number of documents in the collection, and $n_t$ is the number of documents that contains the term $t$.

The ranking of a document with regard to query is defined as the vector distance measured between their respective vector representations. This ranking is assumed to be correlated with the probability of relevance of the document. In practice, the distance measure is defined as the cosine of the angle between the vectors:

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{t=1}^{t} w_{t,j} \times w_{t,q}}{\sqrt{\sum_{t=1}^{t} w_{t,j}^2} \times \sqrt{\sum_{t=1}^{t} w_{t,q}^2}} \qquad (3.3)$$

where $w_{t,q}$ corresponds to the weight of term $t$ in query $q$, whose definition is equivalent to the weight of a term in a document. The factors $|\vec{d_j}|$ and $|\vec{q}|$ correspond to the norm of document and query vectors, respectively. The ranking calculation is not affected by $\vec{q}$ because its value is the same for all documents.

Thus, we can say that the Vector Space Model relies on three basic components: the frequency of the term on the document ($tf(t, j)$), the inverse document frequency of the term ($idf(t)$) and the norm of the document ($|\vec{q}|$).

The Space Vector Model is usually applied in classical Information Retrieval problems such as search engines, where the documents are web pages and the queries are web queries specified by the users. In this work, we consider the textual content of the ad as the document and the web page (or the keywords that represents its content) as the queries. Thus, a list of ads sorted by similarity is returned when submitting a web page as a query.

## 3.3 Machine Learning

Machine Learning can be defined as a set of techniques that allows an algorithm to learn and improve its performance without being explicitly programmed. Usually, these techniques learn patterns from sets of examples given as input and apply this patterns on future predictions.

Many areas, like digital image processing, speech recognition, web search and also keyword detection already use Machine Learning algorithms to help solving their problems. Among the most popular problems where these algorithms present good results we highlight the Classification and Regression.

Classification is a problem where the objective is to classify some data into some specific groups according to some features. For example, selecting keywords from a text can be modelled as a classification problem where each word has to be classified as relevant or irrelevant to the main content of the text and the features could be the frequency of each word on the text.

In this example, the classification algorithm could analyse a set of examples of words labelled as keywords or not keywords and aims to find a pattern using these features provided to, given a new word, label it as relevant keyword or not.

On the Regression problem, the objective is to predict the value of some unknown variable $y$ using a set of other variables $x_1, x_2, x_3, ..., x_n$, which the value is already known. These variables are also called features. For example, finding keywords on HTML pages can be defined as a regression problem where each word is associated with three features: their $IDF$, their $TF$ and the fact the word occurs on the title of the page or not. The objective would be to predict the probability each word is a relevant keyword.

Thus, the Regression algorithm would aims to provide us some model where we can input the variables we already know (i.e. the set of features) and the output is the prediction of the probability of each word is a relevant keyword.

## 3.4 Metrics for Evaluation

The efficacy of each method can be measured in several distinct aspects. In this Section we present the main metrics adopted to evaluate the quality of the proposed methods.

### 3.4.1 Precision and Recall

Precision and Recall are well known metrics in the Information Retrieval area. We used them as our main quality measure in this work. The Precision of a method is the ratio between the number of relevant answers and total number of answers returned by the method. In this work, the Precision of each method is its average precision considering all the documents on the dataset used. The Recall is the ratio between the number of relevant answers returned by the method and the total amount of known relevant answers on the referred document. In this work, the Recall of each method stands for its average considering all the documents of the dataset used.

We also analyse the precision and recall of each method taking into account only the top n ranked answers. In this case, we use the *Precision at n* (also referred as $p@n$) and

*Recall at n* (also referred as $r@n$).

The *Precision at n* of a method is calculated as follows:

$$p@n = \frac{|rel \cap answersattopn|}{|answersattopn|} \tag{3.4}$$

where $rel$ is the set of relevant answers associated with the document in the pool and $answersattopn$ is the first $n$ ranked answers provided by the evaluated method. Note that, as we consider only the $topn$ answers, the maximum value of $|answers|$ is $n$. Indeed, in some cases, the system retrieves less then $n$ answers.

A problem found when computing $p@n$ in our experiments is that it is common to find cases where a method does not provide answers. This is a quite common problem in real case collections. In these cases, the precision for such specific document cannot be determined. To cope with that, we could define $p@n$ as 0 (or even 1) if no answer is provided, however, we think such a definition does not reflect the real precision. We then calculate $p@n$ as an average over only the pages where at least one ad was returned by the method. However, by adopting only that strategy we do not provide full information about quality. For instance, if method $A$ returns just one answer for one document (out of a set of 100 documents) and this answer is relevant, $p@n$ for A would be 100%. Retrieving only such result would hide the fact that the method could not retrieve any answer (relevant or not) with the other 99 documents.

Thus, to provide more insight about the performance of each approach, we also introduced the computation of *Recall at n* which is calculated taking the number of relevant answers found in the pool as the set of relevant answers, but limiting this number to the maximum of relevant answers considered by each method. For example, if we are evaluating only the top 10 results of each method, any method that returns 10 relevant results in the top 10 results provided is going to have a $recallat10$ of 100% nevertheless the size of the pool. The $Recallatn$ is calculated as follows:

The $r@n$ of a method given a document $p$ is:

$$r@n = \frac{|rel \cap answers|}{min(|rel|, n)} \tag{3.5}$$

where $rel$ and answer are defined as in the previous equation and $min$ is a function that returns the minimum value of two arguments.

### 3.4.2 F-Measure

As the precision and the recall of a method evaluate distinct aspects, we also decided to use a measure to unify these two values into one single score. This measure is the $F - Measure$ (also referred as $F1$). It is a weighted average between Precision and Recall, calculated as $F1 = \frac{2 \times Prec \times Rec}{Prec + Rec}$. Where $Prec$ and $Rec$ are calculated taking into account the complete set of answers returned by the evaluated method.

### 3.4.3 MRR

Also, a common way to evaluate rankings, is the MRR (Mean Reciprocal Rank). The Reciprocal Rank of a method is is obtained by dividing 1 by the position where the first right answer was found. Thus, the MRR of a method $A$ is the average value of the Reciprocal Ranks obtained for each document using the method $A$ as described on the following equation:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rankrel}_i}$$

where $Q$ is the set of queries evaluated with $|Q|$ being the number of queries and $rankrel_i$ is the position where the first relevant answer appears in the ranking.

# Chapter 4

# Selecting Keywords from Web Pages

A naive approach to determine the importance of a keyword from a web page could be using some basic information, like its frequency on the page, as a value to measure its importance. However, there are several other alternatives that can be useful when trying to predict the importance of a keyword on a web page, like position, length and HTML tags. Previous approaches have been proposed on the literature to take advantage of this kind of extra information [38, 40, 22, 14]. Most of them use machine learning algorithms, in which a term (or a sequence of them) can be modelled as an object with several features. The aim in these cases is to learn specific patterns which would clearly distinguish good ad keywords.

Machine learning methods for selecting keywords present the advantage of usually results in high quality keyword selection methods, but on the other hand, present as a drawback the necessity of continually update of the learning models. This requirement may be particularly problematic on ad selection methods, where both the web pages and the ad collection are continually updated. Another alternative considered by several authors is the development of methods that do not adopt machine learning.

We thus decided to divide our study about selection of keywords from web pages into two parts. First, we study alternatives that take advantage of machine learning to select keywords, studying the alternatives found in literature and proposing a new way of modelling the problem that is application driven. Second, we revise the methods that select keywords without using learning and present again an alternative method proposed by us.

## 4.1 Using Machine Learning to Select Keywords

The machine learning approaches proposed in literature for selecting keywords from texts are modelled with the goal of selecting keywords considered as good by humans. We here propose a new machine learning strategy where the selection of keywords is driven by the expected impact they have in the final quality of the ad placement system, which we name as *ad collection aware* also referred to as *ACAKS*. Our intuition is that the keywords selected by the users as the best for advertising are not always the most appropriate for contextual advertising systems.

More specifically, on this work we take advantage of the ad collection by changing the strategy used to compose the training collection which guides the learning process. Instead of asking users to directly label examples of what are the good keywords found in the training pages (which we call *traditional approach*), we gather the ads which have a match with any sequence of one or more word candidates to be a keyword (*keyword candidate*) found in the training pages and ask the users to evaluate their relevance.

As we show, this strategy provides competitive results. Further, while at a first glance it appears to be prohibitively expensive, we show that it is possible to perform training with the *ACAKS* method and that it produces, at least in our experiments, results superior to the baseline, being an interesting alternative to select keywords from web pages for ad placement purposes.

We used the features proposed by Yih et al [40], which was the best alternative we found in literature to select keywords from ads, being also a good candidate for a baseline. We also assess the impact of using a new set of features derived from the ad collection. While these new features are not studied in [40], previous research articles indicate that ad collection information is useful to improve the quality of results in ad placement systems [30, 6], which motivated us to investigate its use also in the keyword selection problem.

The inclusion of ad collection features also makes the comparison between the methods fairer, since we can say that the *ACAKS* approach indirectly uses information about the ad collection as part of its keyword selection method. The selection of keywords in the *ACAKS* is guided by the relevance of the ads they can bring, and we can say it implicitly uses ad collection information in its keyword selection process. When including ad features in the experiments, we also provide information about ad collection to the traditional approach, thus allowing both methods to take advantage of this information.

### 4.1.1 Keyword Selection as a Classification Problem

As Yih et al [40], we address the problem of determining the advertising relevance of a keyword (sequence of terms) as a classification problem. Thus, let $K = \{k_1, k_2, ..., k_n\}$ be a set of keyword candidates. Each keyword $k_i$ is represented by a set of $m$ features $F = \{F_1, F_2, ..., F_m\}$, such that $k_i = (f_{i1}, f_{i2}, ..., f_{im})$ is a vector representing $k_i$, where each $f_{ij}$ is the value of feature $F_j$ in keyword $k_i$. Note the term *feature* describes a statistic that represents a measurement of some advertising relevance indicator associated with a keyword candidate.

We assume that we have access to some *training data* of the form

$$\{(k_1, r_1), (k_2, r_2), ..., (k_n, r_n)\} \subset K \times \{0, 1\}$$

where each pair $(k_i, r_i)$ represents a keyword candidate and its corresponding relevance value, such that if $r_i = 1$, then the candidate $k_i$ is a keyword. Otherwise, it is not a keyword.

Using this learning approach, the solution to this problem consists in: (a) determining the set of features $\{F_1, F_2, ..., F_m\}$ used to represent the keyword candidate in $K$; and (b) applying a classification method to find the best combination of the features to predict the relevance value $r_i$ for any given keyword $k_i$.

To accomplish this, we use a logistic regression that, by means of the logistic function (see eq.(4.1)), computes the probability of a keyword being relevant for advertising as a function of the values of its relevance indicators.

$$f(z) = \frac{1}{1+e^{-z}} \tag{4.1}$$

Note that, in eq.(4.1), given a keyword $k_i$, $z = \beta_0 + \beta_1 f_{i1} + \beta_2 f_{i2} + ... + \beta_m f_{im}$. The values $\beta_0, \beta_1, ..., \beta_m$ are the regression coefficients which indicate how important are the relevance indicators $f_{ij}$ to the probability of $k_i$ being relevant.

We perform a regression for each class, setting the output equal to one for training instances that belong to the class and zero for those that do not. Then, given a candidate, we calculate the value of the logistic regression expression both for keywords and not keywords and choose the one that is largest. This value is also used to rank the keywords. We decided to use logistic regression in the classification task because it was also used by Yih et al [40], work that will be used as baseline for our experiments.

### 4.1.2 Definition of Keyword Candidates

In the method proposed in [40] and *ACAKS*, we use a keyword candidate definition that follows the same settings of the monolithic combined candidate selector described in [40] since this was found to be the best keyword selector in that work. More specifically, a keyword candidate is any word or phrase (consecutive words up to length 5) that appears in a page, in any of the sections: title, body, and meta-tag.

Phrases are not selected as candidates if they cross sentence or block boundaries. Further, phrases are taken as individual entities, such that features do not describe statistics about phrase constituent words but about the entire phrase. Note that no stemming normalization or stopword filtering was applied. In spite of that, phrases were not selected if they start or finish with stopwords.

### 4.1.3 Keyword Relevance

As previously mentioned, we assume that we have access to some training data of the form

$$\{(k_1, r_1), (k_2, r_2), ..., (k_n, r_n)\} \tag{4.2}$$

where each pair $(k_i, r_i)$ represents a keyword candidate and its corresponding relevance value. To obtain the relevance values $r$ we use two strategies. In the first one, human judges chose the keywords, as it is proposed in [40]. This first strategy is used here as a baseline. In second one, the candidates were taken as keywords according to their capability to trigger relevant ads. This second strategy is our new proposal to select relevant keywords.

In the first approach, we ask volunteers to label as keyword the words or phrases they judge relevant for advertising in a test collection. Volunteers are instructed to select keywords respecting the definition of keyword candidates presented in Section 4.1.2. Given a candidate $k_i$, it is considered as relevant ($r_i = 1$) if, at least, one user labels it as a keyword[1]. Otherwise, it is considered as irrelevant ($r_i = 0$). In this work, we referred to the keywords selected using this strategy as the baseline.

In the second approach, which is our proposal and we name as *ad-collection-aware* keyword selection (also referred as $ACAKS$) we retrieve the most similar ads for each keyword candidate in the training pages. We then ask users to evaluate the relevance of the ads for the page where the keyword candidate was extracted from. More specifically, for a given keyword $k_i$, five ads are retrieved according to their similarity to $k_i$. Candidate

[1] We also considered in preliminary experiments the possibility of using as thresholds the values 2 and 3, but these threshold resulted in worse quality when compared to threshold 1. While requiring more relevant judgements to consider an ad as relevant could improve the precision of the method, such constraint results on a small set of positive examples. With few examples, the learner is not able to build a good model which leads to low accuracy. On future work, we intend to increase the amount of pages used in the training to obtain a more accurate result.

$k_i$ is considered as relevant ($r_i = 1$) if at least one of these ads is considered as relevant for being presented in the page. Otherwise, $k_i$ is considered as irrelevant ($r_i = 0$). As in the baseline, we also experimented considering an keyword as relevant only when $n$ ads associated with it were considered relevant, for different values of $n$, but again the best results were achieved with threshold 1.

Note that this new method is also based on learning from human evaluation of relevance, but requires a different kind of information. Our assumption is that it is easier for a human to select relevant ads for being placed in a page than directly determine what is a relevant keyword.

Further, a natural advantage of this new approach is that reference collections adopted for evaluating the performance of ad systems already contain the training information we require, since to create such reference collections it is necessary to evaluate the relevance of ads given a web page. Reference collections are also available if the ad system uses any learn-to-advertise approach [21]. Thus, in practice, the change of focus in the selection of keywords may reduce the cost for training. Also, click-through information may be used as an approximation for human judgement relevance for ads since, for most of the keywords. This information is already available for companies that operate sponsored search systems.

One could argue that if no reference collection is available, the cost of training in *ACAKS* keyword selection is higher. While this is not a practical situation, still the cost is not so high to avoid the application of the method, since a large number of keyword candidates do not have a match to the ad collection in practical systems, a phenomenon that is referred to in the literature as *impedance* between the web page vocabularies and the terms founds in the ad inventories [30, 40].

To retrieve the most similar ads to a keyword candidate we used the ADKW method, described in [30]. This model was adopted in literature as a ranking method in contextual advertising [30]. This model considers an ad as the concatenation of all the terms on the fields title, description and keywords of the ad and applies the Vector Space Model to rank the ads.

We selected this simple algorithm because it has already been used to rank ads in literature and our main focus here is to validate our keyword selection method.

### 4.1.4 Keyword Representation

In this Section, we describe the features used to represent the keywords. These features are extracted from the textual content of the pages and query log. They were originally proposed and extensively studied in [40]. From the set of features used in that work, we have omitted the linguistic ones derived from the annotation obtained using a part-of-speech tagger. As observed by the authors in [40], linguistic features did not help in this domain, providing redundant information with other features, easier to calculate, such as capitalization and presence in query log.

The features are organized in several groups, as described as follows:

- **C**apitalization: whether the keyword is capitalized. The capitalization can indicate the keyword is part of a proper noun, or is an important word.
- **H**ypertext: whether a candidate phrase or word is part of an anchor text present on the page the candidate belongs to
- **T**itle: whether the candidate is part of the TITLE field.
- **M**eta features: whether the candidate is part of the meta description, meta-keywords or meta-title fields of the HTML document.
- **M**eta section features: whether the candidate is part of the metadata section present on the header of the HTML document.
- **U**RL: whether the candidate is part of the URL string
- **I**nformation retrieval features: the TF (term frequency) and DF (document frequency) values of the candidate. The document frequency is the number of documents in the web page collection that contains the candidate. In addition to the

original TF and DF, $log(TF + 1)$ and $log(DF + 1)$ are also used as features to provide this information in a different (logarithmic) scale.

- **R**elative location of the candidate: the beginning of a document often contains an introduction with important words and phrases. Therefore, the location of the occurrence of the candidate is extracted as a feature. Since the length of a document varies considerably, we use the relative location by considering a normalized document length equal to 1. When the candidate is a phrase, its first word is used as its location. There are three different relative locations used as features: (a) *wordRatio*: the relative location of the candidate in the sentence; (b) *sentRatio*: the location of the sentence where the candidate is in divided by the total number of sentences in the document; (c) *wordDocRatio*: the relative location of the candidate in the document. In addition to these 3 features, we also use their logarithms as features to provide this information in a different scale. Specifically, we used $log(1 + wordRatio)$, $log(1 + sentRatio)$, and $log(1 + wordDocRatio)$.

- **S**entence and document length: the length (in words) of the sentence (*sentLen*) where the candidate occurs, and the length of the whole document (*docLen*) (words in the header are not included) are used as features. Similarly, $log(1 + sentLen)$ and $log(1 + docLen)$ are also included.

- **L**ength of the candidate phrase: the length of the candidate phrase (*phLen*) in words and $log(1 + phLen)$ are included as features.

- **Q**uery log: the query log of a search engine reflects the distribution of the keywords people are most interested in. We use the information to define three features: whether the phrase appears in the query log, the frequency with which it appears and the log value, $log(1+frequency)$. In this work, we used the query log described in Section 4.1.5.

### 4.1.5 Experimental Evaluation

We here describe the datasets, the experimental methodology we used to conduct our empirical study and the results obtained.

**Environmental Setup**

To train and evaluate our ad placement framework, we used a test collection built from a set of 300 pages extracted from a Brazilian newspaper. As we have no preference for particular topics, these pages cover diverse subjects, such as culture, local news, international news, economy, sports, politics, agriculture, cars, children, computers and Internet, among others.

The IDF information we have used was obtained from a commercial search engine. We submitted each keyword candidate selected from the pages in the experiments (referred also as $kw_{candidate}$) as a query to the search engine and the number of documents retrieved was considered as the DF (Document Frequency). We then computed the IDF as $log(\frac{N}{DF(kw_{candidate})})$, where $N$ is the total number of documents found in the search engine collection. As no search engine provides this information explicitly we estimated it by searching for some stop words, like ”a” and ”the”, and then considering the highest number of results obtained as the value of $N$.

The ads used in our experiments were obtained from a real case ad collection composed of $93,972$ ads grouped in $2,029$ campaigns provided by $1,744$ advertisers. With these ads, advertisers associated a total of $68,238$ keywords[2]. In this collection, only one keyword is used by the advertiser to describe each ad.

We need to obtain reference sets containing information about what keywords are useful to represent web pages and also containing the ads that are relevant to be placed in each of the 300 web pages of the collection. These information will be used in the training and test phases of the experimented keyword selection approaches.

[2] Data in Portuguese language provided by an on-line ad company that operates in Brazil.

To train and test the baseline method, we need to construct a set of keywords that should be manually labelled by users. To obtain such training we present each of the 300 pages of the test collection to volunteers (60 volunteers contributed to all phases of our experiments), asking them to select keywords from these pages following the keyword candidate guidelines described in Section 4.1.2, and considering that the purpose of the keyword selection is to associate pages with relevant ads.

Note that the decision about the selection of keywords depends exclusively on the judgement of the volunteers. The result of this process is a set of keywords associated with each of the 300 web pages of the collection. Given a page $p$, we name this set as the *human tagged keywords* of $p$, denoted as $(HT_{Kw}(p))$. In the experiments, users tagged an average of 14.33 keywords per page. From these keywords, very few were labelled by more than one user. More specifically, 13.88% were labelled by two or more volunteers and 3.28% were labelled by the three volunteers.

The reference collection for $ACAKS$ method also requires a set of relevant ads related to each of the 300 pages. To obtain this set, we extracted keyword candidates. As the textual content of some web pages is very large (1000 or more words), we take into account only the first 400 words in each page. Such constraint does not affect most of web pages and reduce the number of keyword candidates considered for these very large pages. As we consider that eliminating or rising this threshold could improve even more the results obtained by our method, we intend to study the impact of using higher values as future work. The average number of $k$eyword candidates per page we found by using this approach was 279.76. The result of this process is a set of $k$eyword candidates associated with each of the 300 web pages of the collection. Given a page $p$, we name this set as the $k$eyword candidates set of $p$, denoted as $(C_{kw}(p))$.

We submit each $k$eyword candidate as a query to the indexed collection of ads and take the top five answer results, using the $ADKW$ [30] as the ranking method. For each page $p$, we create a set of ads $AD_P$ composed by the answer results obtained from all the $k$eyword candidates found in $C_{kw}(p)$. As a result of the above process, we selected a total of 95,327 distinct pairs of ads and pages corresponding to an average of roughly 317 ads per page.

Then, for each pair ($p$,$a$), $p$ being a page and $a \in AD_P$, three human volunteers judged whether $a$ is relevant to $p$ or not. Thus, note that we have $285,981$ evaluation of pairs. To reduce the costs of this phase, each volunteer evaluated a set of 15 pages and the ads associated with them. Using this strategy, each volunteer spent at most three hours labelling ads as relevant or not. Note that this effort would not be necessary if clickthrough information were available for the ad collection adopted. It could be also avoided if we had obtained a reference collection with the complete set of relevant ads for each page.

We consider as relevant to $p$ an ad labelled as relevant by at least one volunteer. Only 20.90% of the relevant ads were labelled as relevant by two or more volunteers and 6.88% of them, labelled as relevant by the three volunteers. Finally, a *k*eyword candidate is considered relevant to $p$ if at least one of the ads retrieved by it is relevant. The average number of relevant ads per page obtained with this process was 41.83, and the number of relevant keywords per page obtained in the reference set according to the *A*CAKS method was 21.35.

The result of this process is also a set of *k*eywords associated with each of the 300 web pages of the collection. Given a page $p$, we name this set as the *s*core tagged keywords of $p$, denoted as ($ST_{kw}(p)$). Table 4.1 summarizes the information about the training data used on *ACAKS* approach.

| ACAKS |
|---|
| 300 pages |
| 279.76 keyword candidates/page (average) |
| 317 ads/page (average) |
| Total of 95,327 pairs (ad,page) evaluated |
| 41.83 relevant ads/page (average) |
| 21.35 keywords/page (average) |

Table 4.1: Training details of ACAKS approach.

The query log features used on this work were derived from the query log of the

WBR03 collection, a database extracted from the Brazilian web which contains queries submitted to TodoBR[3], a real case search engine. The log consists of $12,795,101$ queries and $2,987,745$ distinct queries.

### 4.1.6 Evaluation Methodology

To perform the experiments, we used the 10-fold cross validation method [27]. All the results reported are average values of the 10-fold runs and for all comparisons reported in this work, we used the Student's t-test [11] for determining if the difference in performance was statistically meaningful. We consider statistically meaningful results with a $p$-value $\leq 0.01$. We assessed the performance of each keyword selection method proposed through four distinct experiments, described in the following paragraphs.

First, we measured the quality of our keyword classifier using the accuracy measure, which is defined as the proportion of correctly classified examples for this purpose. Although this experiment is interesting to measure the quality of the classifier on each approach, it is important to note that the main goal of $ACAKS$ is to select keywords to improve the quality of advertising systems.

In the second experiment, we evaluated the quality of ads retrieved by the keywords obtained by each approach. The set of keywords returned by each method was used as a query submitted to a system which returned a ranking of ads based on the $ADKW$ method [30]. This experiment aimed at asserting the impact of the keyword selection strategies studied, when used in an ad selection system. The metric adopted was precision and recall considering the top 3 ads retrieved by each method.

The third experiment shows the performance of each approach with training sets of different sizes. The objective of such experiment is to discover the behaviour of each method while increasing and decreasing the size of the training set.

Note that both the second and third experiments can bring new pairs of ad and pages

[3] TodoBR is a trademark of Akwan Information Technologies, which was acquired by Google in July 2005.

not evaluated in our initial pool. Thus, a second round of evaluation was required to complete the set of relevant ads associated with each page. In this second round we found an average number of 1.34 ads not evaluated per page. After evaluating then, we found an average of 0.65 extra relevant ads per page.

Finally, the fourth experiment includes ad collection features in the learning process in order to check if their inclusion changes the comparative results between the $A$CAKS and the baseline approaches.

### 4.1.7 Experimental Results

This Section presents the results of experiments we conducted to evaluate our proposed methods and compare them to the baselines.

In all tables of this Section, we present the results obtained while using *ACAKS* and baseline approaches and the performance of *IDEAL-baseline* and *IDEAL-ACAKS*. We refer as *IDEAL-baseline* the method that classifies as keywords exactly the ones (and only them) labelled as such by the human evaluators, i.e., the ones present in $HT_kw(p)$, for each $p$ in the test set. In the same way, we refer as *IDEAL-ACAKS* to the method which classifies as keywords exactly the ones (and only them) taken as relevant by the *ACAKS* approach, i.e., the ones found in the set $ST_{kw}(p)$, for each page $p$ in the test set. The IDEAL methods emulates the performance of an ideal classifier, which could classify all the keywords correctly according to the definition adopted by each approach.

Our first experiment aims at evaluating the quality of the keyword selection methods, taking the human judgements as our gold standard. Note that in this scenario, it is expected that the baseline approach reaches better accuracy than the *ACAKS* one, since in the first approach the keywords were learned taken as training set *exactly* the gold standard, whereas in the second approach, the keywords were selected based on their performance on triggering ads. In fact, the resulting training sets are quite different. Out from $10,444$ keywords provided to train the methods, only about 10% occurred on both training sets. Table 4.2 depicts the accuracy results of the studied keyword detection approaches. Note that the results presented in this Table consider as relevant keywords of a given page $p$ only the human tagged keywords, thus $HT_{kw}(p)$.

| Method | Accuracy |
|---|---|
| baseline | 29.16% |
| *ACAKS* | 29.16% |
| *IDEAL-baseline* | 100% |
| *IDEAL-ACAKS* | 22.41% |

Table 4.2: Accuracy of each method in the task of selecting good keywords, the keywords in the test sets were labelled by humans.

While there was little intersection between the keywords labelled as relevant in the training sets used by the *ACAKS* and the baseline approaches, the methods achieved the same performance. After a careful inspection of the keywords used for training and the ones selected by the methods in the test, we noticed twice as many keywords in common in the test than in the training. However, the difference in the output of the methods is still large. For instance, from the total of keywords selected by both the baseline and *ACAKS* methods, about 80% were different. Among the examples of keywords selected only by the baseline approach, we cite “rig” and “whiz kid”. These keywords, in general, are not found at all or have little importance in the ad collection. On the other hand, examples of keywords selected only by the *ad-collection-aware* approach are “advising”, “team”, and “work of art”. These keywords normally have peripheral importance in the pages. As we show in the next experiment, many of these candidates not selected by baseline, but caught by *ACAKS*, have triggered interesting ads.

The performance presented by the approaches indicates that the *ACAKS* approach may be used as a more general annotation method to find keywords of web pages. While this is not the focus here, we plan to study this possibility as a future work.

A second important point to observe from Table 4.2 is that *IDEAL-ACAKS*, a classification method that would take exactly the correct keywords according to the *ACAKS* approach, would in fact result in a classifier with less relevant keywords according to the human evaluation performed. However, as we show in the next experiment, such result does not necessarily imply a worse keyword selection for the ad placement system. This result reinforces our initial intuition that a *ACAKS* keyword selection approach may be better than the baseline approach in ad placement systems.

Although the results presented in Table 4.2 are important to reinforce our initial intuition and better understand the behaviour of both methods, the main objective of the proposed approach is to select keywords that are useful on the task of retrieving relevant ads. The objective of the following experiment is to assess the performance of each method on such scenario. Table 4.3 shows the average precision and recall at the top three results retrieved by each method (*ACAKS*, the baseline and both their IDEAL versions as described above). Note we consider as relevant ads chosen by, at least, one user.

| Method | p@3 | r@3 |
|---|---|---|
| baseline | 0.4478 | 0.1933 |
| *ACAKS* | 0.4774 | 0.3133 |
| *IDEAL-baseline* | 0.5872 | 0.5833 |
| *IDEAL-ACAKS* | 0.7678 | 0.7678 |

Table 4.3: p@3 and r@3 for each method. An ad is considered as relevant to a page if at least one user label it as relevant.

We first note in Table 4.3 that the results obtained by the *ACAKS* approach were better than those achieved by the baseline, confirming our assumption that our approach is quite better on retrieving relevant ads. The precision achieved by both methods was quite close (0.4478 for the baseline and 0.4774 for *ACAKS*) and the difference between them in terms of $p$@3 were not statistically meaningful. Thus, we can conclude that in terms of $p$@3, both methods are equivalent.

When considering the recall, the *ACAKS* approach improves the result obtained by the baseline by more than 62%. It indicates that the proposed approach is able to select a higher number of ads when compared with the baseline, achieving this improvement without dropping precision. These results show the importance of choosing keywords considering the quality of the ads they will retrieve and not what humans believe to be good keywords directly.

By using the *ACAKS* approach, 593 ads were displayed, for a total of 267 relevant ads.

From the total of pages, 133 have received at least one relevant ad. By using the baseline, 339 ads were displayed for a total of 157 relevant ads. Only 86 pages have received at least one relevant ad.

We can also observe in Table 4.3 the precision results obtained by the perfect versions of the baseline and *ACAKS*. If such classifiers could be used, we would obtain a *ACAKS* result more than 60% better in terms of $p$@3 and 145% better in terms of $r$@3. Similarly, we would improve the baseline results by more than 30% in $p$@3 and more than 200% in $r$@3. Such findings indicate we have much room for improvements by enhancing the accuracy of our automatic classifiers. Further, the gain obtained by an *IDEAL-ACAKS* when compared to an *IDEAL-baseline* would be 31.73%. Such results indicate that the *ACAKS* approach is a fair better alternative strategy for extracting keywords in ad selection systems.

A possible reason for the best performance of the *ACAKS* method is the fact that the intersection between the keywords selected by this method and the ads vocabulary is high. While only 67.82% of the keywords selected by the baseline approach were found on at least one advertisement, this number rises to 97.10% while considering the *ACAKS* approach.

Also, based on an anecdotal analysis of our data, some factors which contribute for human judges selection errors are (a) their tendency to avoid keyword candidates in text fragments of peripheral importance and (b) their general lack of knowledge about the ad database, in particular, regarding its vocabulary and advertising opportunities. We believe these problems are smoothed by the *A*CAKS approach.

Another important aspect to be considered is the impact of the size of the training sets on the results obtained by each approach. The *A*CAKS approach relies on judgements about the ads related to each *k*eyword candidate, so the number of ads associated with a page to be evaluated is quite high, while the effort to label keywords in the baseline approach tends to be smaller, since users need only to label the keywords in the training pages. As the effort to produce both training collections is quite different, one could argue that this is the cause of the difference in precision of the results obtained by the *A*CAKS and baseline approaches.

As the effort required to create a training set for *ACAKS* is high, we evaluated the quality of the results obtained by each approach with training sets of different sizes. Our goal with this final experiment is to measure the importance of the size of training set on the final results obtained by each method.

Figure 4.1 shows the *p*@3 value of each approach with training sets of different sizes. As it can be seen, the performance of both approaches did not increase in the experiments for training sets with more than 150 pages. These results indicate that extra efforts to increase the training set might not be worth.

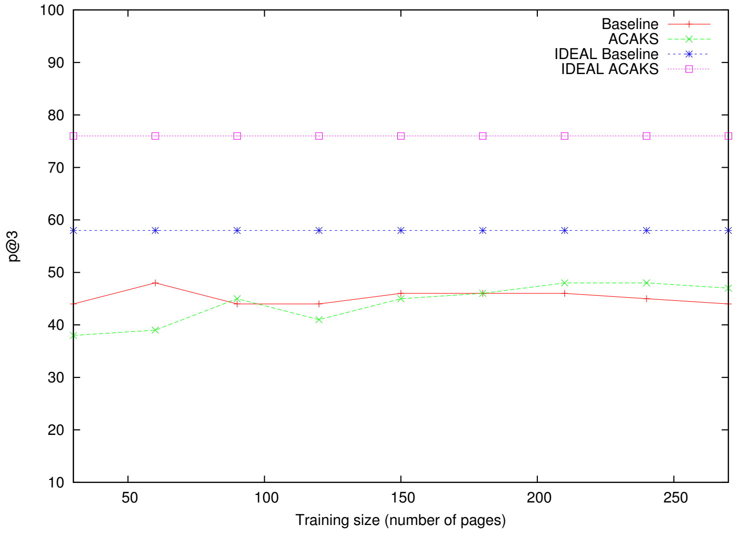


Figure 4.1: p@3 values for training sets of different sizes.

Figure 4.2 shows the *r*@3 value of each approach with training sets of different sizes. Both methods do not show improvements on *r*@3 value using training sets with more than 150 pages. Which leads us to conclude that for this collection, a training set of more than 150 pages cannot provide extra information enough to improve the results.

In this case, the performance of the baseline is quite worse than the performance obtained by the *ACAKS* approach. As a conclusion, we can say that the adoption of *A*CAKS represents an important practical advantage to an ad selection system, since

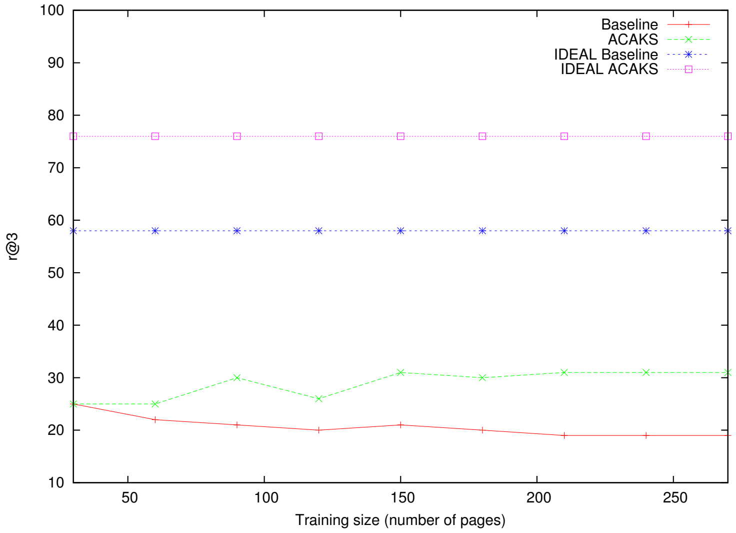


Figure 4.2: r@3 values for training sets of different sizes.

gains in recall may also represent an increasing in revenue that certainly justifies the extra effort required to train.

Both the methods did not take any advantage of using more than 150 pages on the training set in any of the metrics adopted. Also, besides having a higher cost of training, caused by the high number of ads related to each page, *ACAKS* approach outperformed the baseline score in terms of $r$@3. As the $r$@3 seems to stabilize with training sets with more than 150 pages, the results indicate that even increasing the training set to more than 270 pages, the baseline approach would not be able to outperform *ACAKS*.

### 4.1.8 Including Ad Collection Features

Previous work showed that using statistics about the document collection could improve the quality of the ranking [36] produced while using machine learning strategies. Also, the work on [21] presented good results with a set of features obtained using data from the advertising collection. As the *ACAKS* method proposed here obtained good results on the task of selecting keywords using information about the ad collection, we decided to study whether the inclusion of an extra set of features derived from the Ad Collection would improve the results of both *ACAKS* and the baseline or not.

We adopted the following features to be extracted from the ad collection:

- **A**d Section TF: candidate frequency in each of the structural sections of an ad creative. Since an ad has three sections, we use three features to represent them: Ad title TF, Ad description TF, and Ad keyword TF.

- **A**d Section Max-TF: maximum candidate frequency in each of the structural sections of an ad creative. As for *Ad Section TF*, we then have: Ad title Max-TF, Ad description Max-TF, and Ad keyword Max-TF.

- **A**d Section Avg-TF: average candidate frequency in the three sections of an ad creative: Ad title Avg-TF, Ad description Avg-TF, and Ad keyword Avg-TF.

- **A**d Section DF: number of ads in which candidate occurs in a certain section of an ad creative. The three features are in this case: Ad title DF, Ad description DF, and Ad keyword DF.

- **C**ampaign Section Max-TF: maximum candidate frequency in the structural sections of all the ads of a campaign. The three features in this case are: Campaign title Max-Tf, Campaign description Max-FT, and Campaign keyword Max-TF.

- **C**ampaign Section Avg-TF: average candidate frequency in each of the structural sections of all the ads of a campaign. The three features in this case are: Campaign title Avg-TF, Campaign description Avg-TF, and Campaign keyword Avg-TF.

- **C**ampaign Section DF: number of campaigns in which candidate occurs in a certain section of an ad creative. The three features in this case are: Campaign title DF , Campaign description DF, and Campaign keyword DF.

These features were chosen given the success of frequency of terms in objects (**TF**) and of the number of objects where a term occurs (**DF**) as features in previous work related to information retrieval tasks [21, 36].

| Method | p@3 | r@3 |
|---|---|---|
| baseline P.L. | 0.4478 | 0.1933 |
| *ACAKS* P.L. | 0.4774 | 0.3133 |
| baseline P.L.+A.C. | 0.4463 | 0.1967 |
| *ACAKS* P.L.+A.C. | 0.4791 | 0.3144 |

Table 4.4: p@3 and r@3 for each method. An ad is considered as relevant to a page if at least one user labels it as relevant.

Table 4.4 presents a comparison between the results obtained by the *ACAKS* and baseline methods using only the Page and Log features proposed in [40] and described in Section 4.1.4 (referred on this table as baseline P.L. and *ACAKS* P.L.) and the results obtained by the methods using the Page and Log features and the Ad Collection features described above (referred on this table as baseline P.L.+A.C. and *ACAKS* P.L.+A.C.). The usage of Ad Collection features resulted in no gain for both methods. The difference in the results is not statistically significant. As a conclusion of this final experiment, we can say that the *ACAKS* approach is superior to the baseline even when the ad collection features are taken into account. It reinforce the initial intuition that the way *ACAKS* use information about the ad collection is more effective than simply using them as features on the baseline method.

## 4.2 Keyword Selection without Machine Learning

In this Section we study methods for selecting keywords from web pages without using machine learning techniques. When considering this scenario, previous research efforts have adopted methods that adopt external sources of information, particularly Wikipedia, in order to better identify keywords.

We here propose and study new methods that use external information available in the Wikipedia [4] in an attempt to semantically enrich the information extracted from web pages. Wikipedia is a free online encyclopedia created collectively by volunteers. It is composed of articles, and each article is produced collaboratively by a group of editors,

[4] http://www.wikipedia.org

which agree on its content by consensus. At the time of writing, Wikipedia had 3.3 million articles in English, representing one of the largest knowledge bases available online.

We propose and study four alternative ways of using information from Wikipedia to represent web pages. Our methods were engineered to be simple and present light weight computational costs. Thus, the proposed methods are suitable for industrial contextual advertising systems that are required to process millions of requests every day. In these systems, every page view in the web site generates a request, and any method that needs too much computational effort would not be acceptable.

We evaluate the effectiveness of our proposed methods by comparing them to a popular Wikipedia-based keyword extraction algorithm described in [25]. Experimental results show that the proposed methods are competitive in practice, outperforming or achieving results compatible to the baselines in all studied scenarios. Our methods are specially better than the baselines when using small number of keywords to represent each web page. For instance, when selecting 10 keywords to represent a web page, our best method achieves an improvement of 17% in $p$@3 when compared to the keyword extraction method proposed in [25].

### 4.2.1 Wikipedia-Based Keyword Extraction

The first keyword selection method we propose to improve the match between ads and web pages, which we refer to as Wiki-TF-IDF, is a naive method that uses Wikipedia article titles as a controlled vocabulary to extract keywords. The proposed methods called Wiki-Categories-1 and Wiki-Categories-2 are proposal to expand the Wiki-TF-IDF in order to obtain better results.

#### Wiki-TF-IDF

In this method, Wikipedia article titles are used to obtain statistics from the target page (term frequency) and from Wikipedia (inverse document frequency), and these statistics are used to rank the candidate keywords and to select the top $k$ keywords. The intuition behind the method is to extract semantically richer units of information, with the objective of reducing noise and ambiguity in the extracted set of keywords. By extracting more accurately the keywords, Wiki-TF-IDF improves the matching to external datasets.

For example, consider that the phrase “New York” is present in a web page. If we use words as semantic units of information, the keywords “New” and “York” would be extracted. The first word (“New”) carries very low information value, and would certainly be considered a stopword and removed by some methods. The second word (“York”) is ambiguous, since it may refer to the city “York”, or to the city “New York”. Our method considers the phrase “New York” as a single semantic unit, avoiding the negative effect of the noisy keywords.

The Wiki-TF-IDF algorithm works by first extracting candidate keywords from the web page. By using a hash structure to store the Wikipedia titles it is possible to extract all candidates in linear time. Wiki-TF-IDF matches only the largest possible phrase present in a text portion, e.g., in the phrase ”World Wide Web” we only consider the full phrase as a candidate and do not consider the sub-phrases ”World Wide”, ”Wide Web”, ”World”, ”Wide” and ”Web”. In our method we excluded all keywords that occurred less then 3 times in all Wikipedia titles. Moreover, we only considered keywords composed of less than 9 words, so that we could limit the time taken to extract them.

After extraction, the candidates are ranked using the TF-IDF scheme on [42]:

$$w_{ij} = (1 + log(f_{ij})) * log(\frac{N}{n_i}) \tag{4.3}$$

where $f_{ij}$ is the frequency of the candidate keyword $c_i$ in the document $d_j$, $N$ is the number of documents in Wikipedia, and $n_i$ is the number of Wikipedia documents in which the candidate $c_i$ occurs at least once. We rank all the candidates in the document and retrieve the top $k$ keywords.

Wiki-TF-IDF is expected to execute efficiently even in large-scale systems. The main reason is the simplicity of the method.

### Wiki-Categories-1

In this method, firstly, we select the top 10 keywords ranked with Wiki-TF-IDF. Each keyword, which is a title of an article from Wikipedia, belongs to one or more categories in the Wikipedia. We use the categories' names of these top keywords to generate a new pseudo-document. Finally we used Wiki-TF-IDF to rank and the keywords in this pseudo-document of categories and added them to the initial set of top 10 keywords.

### Wiki-Categories-2

In this method, we also select the top 10 keywords ranked with Wiki-TF-IDF and their category names. However, to rank these categories we used the same weight of the keyword it contains. For example, if the keyword ”munition” has weight 0.8 and belongs to the categories ”firearms” and ”artillery”, the keywords ”firearms” and ”artillery” will have the same weight: 0.8.

## 4.2.2 Experimental Evaluation

We analysed the performance of our proposed methods as part of a content targeted advertising system. Our keyword extraction methods were responsible for extracting a set of keywords from the target web page where the ads should be presented. These keywords were then submitted as a query to a search system in order to select and rank advertising stored in an ad inventory database. The objective of the proposed methods is to obtain a set of ads highly related to the page's content.

### Datasets

A problem we endured in the experiments is that real data collections of ads are not publicly available for experiments and the collection we adopted when developing and studying *ACAKS* is considered small and not updated, which would introduce a gap between the ad collection and the external source of information adopted to select keywords, the Wikipedia. We then decided to create a dataset composed by product offers extracted from an online shopping system. The product collection we used in our experiments was obtained from a real case product collection composed of 3,016,544 products from Neemu[5]. Neemu is a price comparison service that crawls product offers from a large set of Brazilian e-commerce stores. In this work we considered that each product offer is described through the concatenation of three distinct attributes: name, brand and category. This collection is referred from now on as *Ad Collection 1*.

While we recognize that this collection is different from the ones available on ad networks, such as the ones maintained by Google and Yahoo, it has the advantage of being now public, which will allow easy comparison of our results with future work. Further, we believe it will also be useful for other future research in the area of content targeted advertising. The list of products available in the dataset is quite extensive, including books, CDs, electronic products, furniture, car accessories, games, groceries, clothes, shoes and almost every type of product sold on the Internet, thus being a quite rich sample of product offers. Further, the books included in the collection cover all themes usually found in a library, thus opening possibilities to matches with pages about almost every topic. Given these properties, we believe that it can be used as a good reference collection to compare the effectiveness of keyword selection strategies for advertising. Further, the announcement of products constitutes anyway a quite common type of advertising usually shown on the Internet.

We also used a real advertising collection composed by 93.972 ads from 1.744 distinct advertisers which is referred here as *Ad Collection 2*.

As target pages (i.e.: the pages where the ads should be displayed) we used two different sets of pages described below:

$P_{Wiki}$: a set of 300 random web pages obtained from the Portuguese version of Wikipedia[6].

$P_{News}$ a set of 300 pages extracted from a Brazilian newspaper.

[5] http://neemu.com

[6] http://pt.wikipedia.org

As we have no preference for particular topics, both the sets of pages cover diverse subjects, such as culture, music, personalities, sports, politics, technologies among others.

Those ads and pages are combined in such a way that results in three distinct scenarios, described on Table 4.5 where each algorithm were experimented.

| | **Scenario 1** | **Scenario 2** | **Scenario 3** |
|---|---|---|---|
| **Pages** | $P_{Wiki}$ | $P_{News}$ | $P_{News}$ |
| **Ad** | Ad Collection 1 | Ad Collection 1 | Ad Collection 2 |

Table 4.5: $S$cenarios where the experiments were conducted.

The Wikipedia database used to compute the idf and the co-occurrence of terms was a dump downloaded in February 2010, from which we obtained 533,358 distinct titles, including full articles, stub articles, disambiguation pages, category pages, list pages and redirection pages. The product offer dataset, the web pages adopted in the experiments and the relevance judgement of the ads associated to each web page will be available for future experiments. The reader can directly contact the authors to obtain the collection.

### Baselines

We used four different methods to compare the effectiveness of our approach. First, we simply used all the terms (except stopwords) in the web page to retrieve items, i.e., we used no keyword extraction algorithm at all. This method we referred to as "All terms".

Second, we used a plain TF-IDF method [31] that uses words as semantic units. We used Wikipedia to calculate the Inverse Document Frequencies. We adopted this strategy because TF-IDF is a method largely used to select keywords from a text. Further, using the TF-IDF extracted from Wikipedia (the same source from where we select keywords using our proposed method) we are able to evaluate the improvements of our method when compared to this previously used strategy.

Third, we used for comparison purposes a method called *Keyphraseness* [25]. It was previously used as part of a successful strategy to extract keywords from web pages using information extracted from Wikipedia [15]. This method uses the probability of a term $t_i$

to be selected as a keyword in a new document. The method considers as keywords the terms that are linked to other Wikipedia articles, i.e., it considers link identification and keyword extraction as the same problem. The Keyphraseness of a term $t_i$ is given by:

$$Keyphraseness\ (t_i) = P(is\ link|t_i) \approx \frac{n_{link}}{n_i} \tag{4.4}$$

in which $n_{link}$ is the number of documents the term $t_i$ occurs as anchor text and $n_i$ is the number of documents where the term occurs at least once. The extracted keywords are ranked and the top $k$ keywords are used to represent the document. Following the procedure in [25], we only considered as candidate keywords the terms which occurred in more than 5 Wikipedia articles.

Fourth, we used the initial approach from what the proposed methods were derived, the Wiki-TF-IDF method. This method is included to explicitly show the difference of including the information about the Wikipedia categories in the keyword selection.

### Evaluation Methodology

We evaluated the quality of the products retrieved by the keywords of each method. Each set of keywords returned by each method was used as a query submitted to a system which returned a ranking of products based on the keywords given as input. The relevance judgement was performed by a group of 30 volunteers, each evaluating the ads returned by the methods to an average of 10 web pages. Volunteers were asked to evaluate each retrieved ad as “relevant” or “non-relevant” in relation to a source web page. They labelled an ad as relevant if they considered that the user who was reading the page would click in the ad presented. Given a web page, we presented to the users the union of results provided by all the variants of the studied methods. The results for a page were presented in a random order to avoid a possible bias caused by the order of results presented .

The system adopted to process the queries and select ads is Lucene [7], configured to rank documents by using the Vector Space Model. The keywords composed of more than

[7] http://lucene.apache.org

one word, such as “South Africa”, were submitted as phrases to Lucene for simplicity. In a practical advertising system a better option could be to change the indexing system to detect these keywords when indexing the ad collection, thus allowing fast search for keywords composed of more than one word.

The methods TF-IDF, Keyphraseness, Wiki-TF-IDF, Wiki-Categories-1 and Wiki-Categories-2 provide a ranking of keywords and associate a weight to each of these keywords. We used as query the top $n$ keywords of each method ($1 \leq n \leq 30$) and included the computed weight for each keyword as part of the query (Lucene allows the assignment of weights to the words in its query processing interface). As using all terms return a set of not ordered terms, their results are constant and presented as an horizontal straight line on the graphics. The “All terms” has an average of 185 keywords per web page. For the methods TF-IDF, Keyphraseness, Wiki-TF-IDF, Wiki-Categories-1 and Wiki-Categories-2 we submitted the keywords with their weights, instructing the query processor for taking the weights into account in the ranking.

All the methods were evaluated using $p$@3 and $r$@3 as described on Section 3.4.

### Results and Discussion

On this Section we present and discuss the results obtained by each method in terms of precision and recall in the three different scenarios.

### Scenario 1

Scenario 1 is composed by the set of pages from Wikipedia ($P_{Wiki}$) and the Ad Collection 1. On this scenario analysed, the proposed methods (Wiki-Categories-1 and Wiki-Categories-2) achieved the best results. Figures 4.3 and 4.4 shows the $r$@3 and $r$@3 values of the methods and the baselines. With only 16 keywords, both the proposed methods achieve $p$@3 values higher than 48%, while the ”All terms” approach, with an average of 185 words per page has a $p$@3 value of 38.6%. Also, the $p$@3 value for Wiki-Categories-2, is 33% higher than TF-IDf, 35% higher than Keyphraseness and 20% higher than Wiki-

TF-IDF. In terms of $r@3$, the results are quite similar, with the Wiki-Categories-1 and Wiki-Categories-2 achieving the best results among all the methods. All these gains are statistically meaningful.

Thus, in this scenario, it is possible to conclude that using the proposed methods could not only lead to a better performance as also reducing significantly the computational costs in comparison with the All Terms approach, achieving better results with less words.

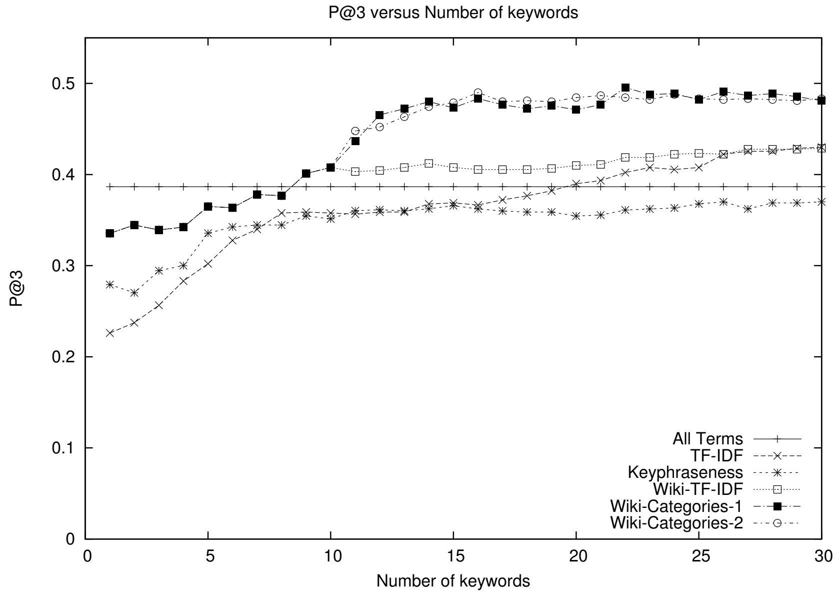


Figure 4.3: p@3 for each method using different number of keywords on Scenario 1. Note that for the method "All terms", we do not range the number of keywords. The method "All terms" have a average of 185 keywords per web page.

### Scenario 2

Scenario 2 is composed by news web pages ($P_{News}$) and Ad Collection 1. The results on this scenario, are present on Figure 4.5 and 4.6. The results were different from those obtained on scenario 1. The proposed methods, did not achieve the same performance, although Wiki-Categories-2 shows stable and competitive results. The statistical analysis of the differences between the best results achieved by each method shows that there is a significant difference only between the Keyphraseness and the other methods. Thus, it is possible to say that, on the scenario 2, Wiki-Categories-1, Wiki-Categories-1, All-Terms,

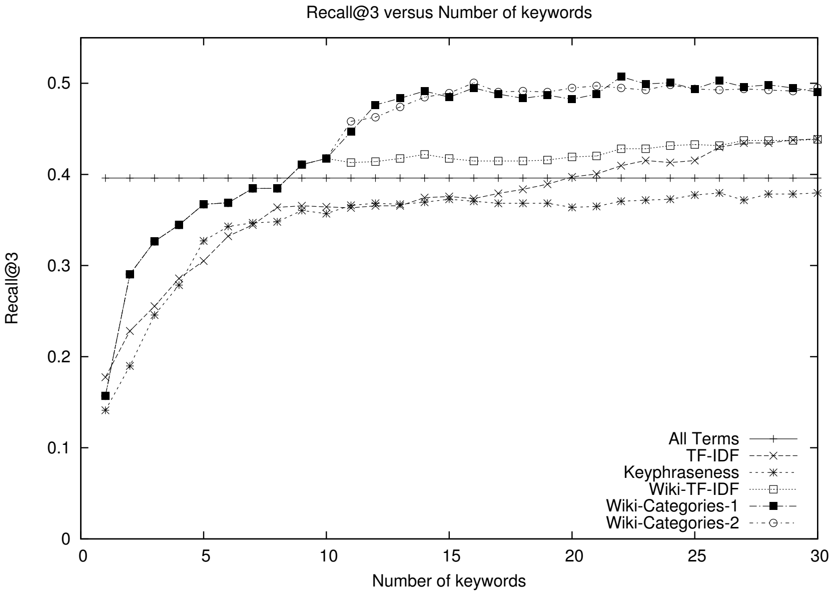


Figure 4.4: r@3 for each method using different number of keywords on Scenario 1. Note that the for method "All terms", we do not range the number of keywords. The method "All terms" have a average of 185 keywords per web page.

TF-IDF and Wiki-TF-IDF presented the same performance.

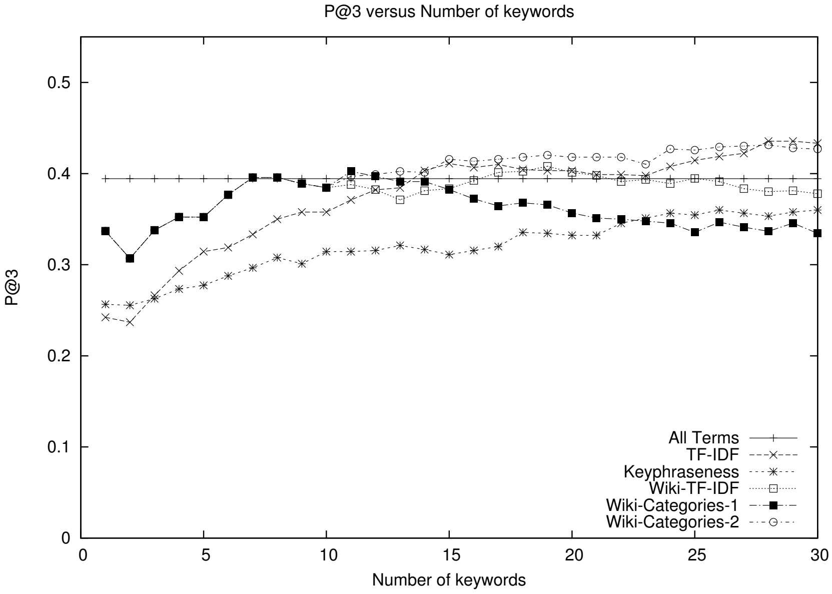


Figure 4.5: p@3 for each method using different number of keywords on Scenario 2. Note that for the method "All terms", we do not range the number of keywords. The method "All terms" have a average of 185 keywords per web page.

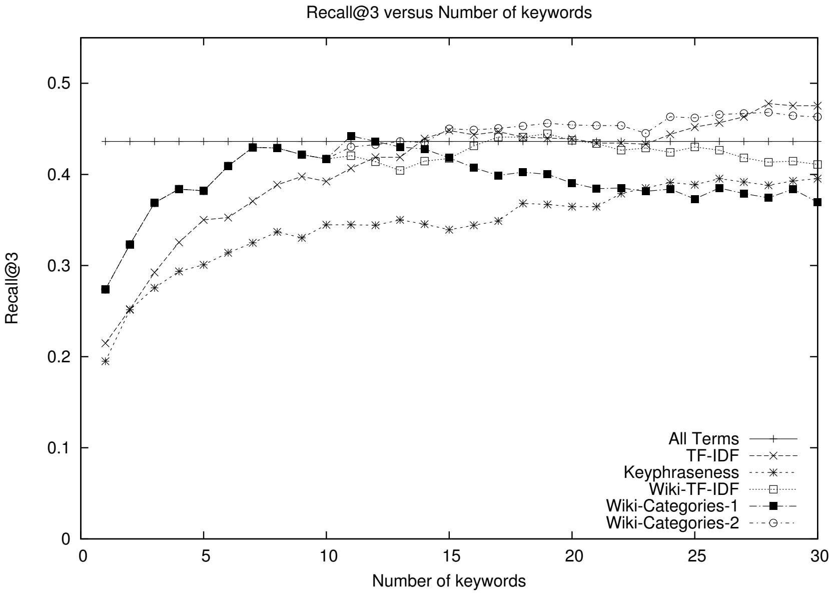


Figure 4.6: r@3 for each method using different number of keywords on Scenario 2. Note that for the method "All terms", we do not range the number of keywords. The method "All terms" have a average of 185 keywords per web page.

### Scenario 3

This last scenario is composed by the set of news web pages ($P_{News}$) and Ad Collection 2. The results obtained on this last Scenario are similar to that on Scenario 2. Thus, we can also conclude that although no gain was obtained, the usage of the proposed methods to selecting keywords is interesting to reduce the number of words necessary to represent the web page.

Through analysing these results presented it is possible to say that both the approaches, Wiki-Categories-1 and Wiki-Categories-2, are competitive alternatives to select keywords on a web page. In scenario 1, where the best results were achieved, 11 keywords are enough to achieve gains over the baselines. And on Scenarios 2 and 3 besides the proposed method present a performance similar to almost all the baselines, the Wiki-Categories-2 presented better results than Keyphraseness.

The results presented show that using Wikipedia can yield better overall results than using only the information present on the document and its collection (as in TF-IDF) because Wikipedia is a rich source of information. Also, the poor performance of Keyphrase-

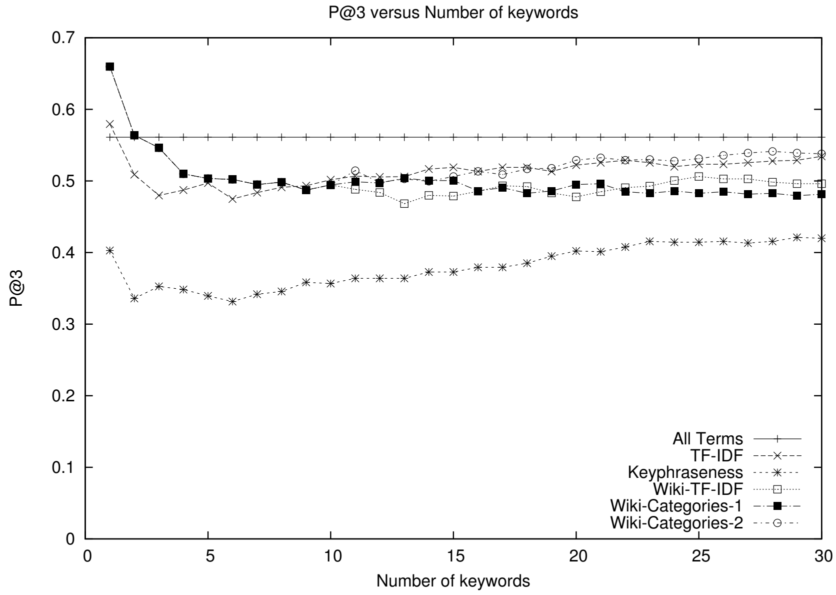


Figure 4.7: p@3 for each method using different number of keywords on Scenario 3. Note that for the method "All terms", we do not range the number of keywords. The method "All terms" have a average of 185 keywords per web page.

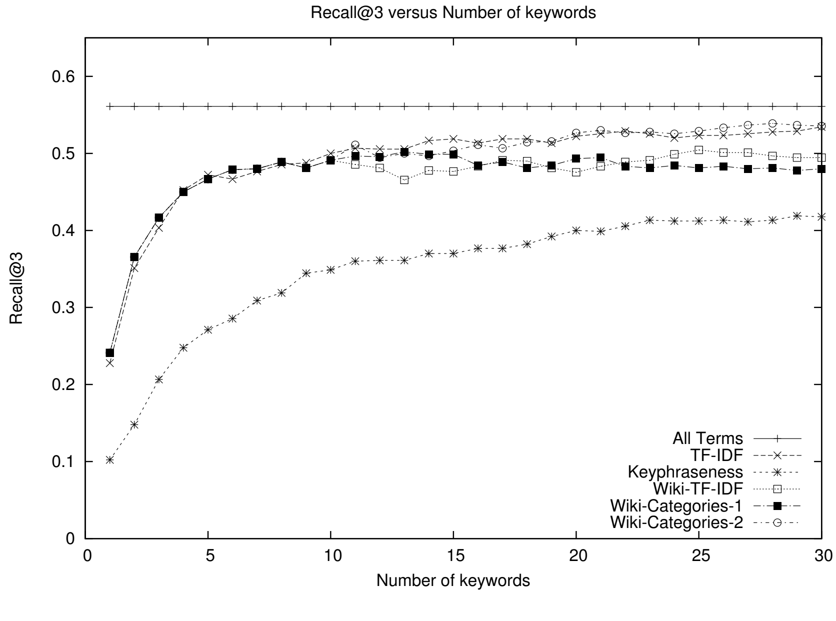


Figure 4.8: r@3 for each method using different number of keywords on Scenario 3. Note that for the method "All terms", we do not range the number of keywords. The method "All terms" have a average of 185 keywords per web page.

ness can be explained because it does not incorporate any information about the frequency of the keyword in the document, which is a very important indicator of its importance.

Finally, the usage of the categories the keywords belongs to seems to be an interesting alternative to expand the initial result obtained by Wiki-TF-IDF without dropping relevance, even achieving the best results in some scenarios.

# Chapter 5

# Selecting Keywords from Short Texts

Several application environments available nowadays allow users to post short texts to express comments, opinions, feelings and other types of statements about a vast variety of subjects. Examples of such applications include microblogs, which are a new media where users publish content over the Web in an extremely easy fashion. For example, Twitter is a microblog that has more than half a billion registered users[1]. Social networks, such as Facebook[2] and Orkut[3], represent another category of popular applications where users usually post short texts.

Besides the popular systems mentioned above, it is common to see short posts of text available on blogs, on-line news systems, social games and nowadays virtually all users that have access to the Internet also have access to such type of resources.

Due to limitations of the applications and devices or even the desire of not spending much time elaborating long texts, most of the posts performed in the examples above are usually very short texts. Finding out the meaning of posts is an important step to take advantage of all the valuable information present in such huge amount of data. Several social media applications, such as search, classification and advertisement need to assess the meaning of the text to work properly. However, the little information provided by

[1] http://twopcharts.com/twitter500million.php
[2] http://www.facebook.com
[3] http://www.orkut.com

such small pieces of text turns this task into a non-trivial one and traditional techniques for mining keywords on texts usually may not be able to achieve satisfactory results when applied to this scenario.

In this work we present a set of methods that take advantage of connectivity information available in Wikipedia to discover the main concepts present on small text portions and associate them with Wikipedia articles, these concepts are here referred as keywords. On this approach, we represent the terms of a text and the relations between them through graphs and then apply link analysis algorithms to predict the keywords available in each short text. The vertices of the graph represent all concepts while their edges represent the relations between them. The intuition behind our method is that if terms from the posts have links connecting them in Wikipedia, they are related terms and have a high chance to be related to the main topic of the post. We use this idea to build several methods in order to find keywords to represent short pieces of text. We performed several experiments and the results achieved show that our proposed methods can lead to very good results. We experiment the scores produced by the proposed methods as a new set of features and apply a machine learning algorithm to improve the results obtained. Also we assess the performance of the approach on selecting products from an online store.

On the task of selecting keywords the proposed methods yielded precision values up to 49.78% and recall values up to 47.66% better than the Commonness method [24]. When considering the usage of machine learning methods, we also achieve an improvement on recall of more than 20% over the baseline approach boosted with machine learning and similar precision values.

## 5.1 Proposed Approach

The problem we address here is to find keywords that describe what the content of a given short text is about. We not only find keywords, but also link them to a unique and unambiguous entry in a knowledge base such as Wikipedia.

The main challenge faced when trying to find keywords that better describe the content of short texts is the absence of contextual information, since contrary to other texts found on the web they do not contain many “clues” to indicate context, such as HTML tags and surrounding text. Also, common information retrieval measures, such as the total frequency of a term, are ineffective in this scenario. Indeed, there is no point in using the frequency of a term in a scenario where usually each word occurs only once.

To circumvent the lack of contextual information, we use Wikipedia as an external source of information, from which we can gather relationships between terms and use them to discover keywords in a short text. Our main assumption is that the links between Wikipedia articles may be used to determine which concepts are most related to each other. We assume that when two or more words appear together in a short fragment of text and these words are interconnected within Wikipedia, they are likely to be related to the main topic of the text. While at first sight such assumption may appear to be naive, we present experiments that indicate that this simple approach yields very accurate results.

We model the connectivity information using what we call *Context Graphs.* Context graphs represent the keywords found in a text as their vertices and the relationship between them as edges. We propose and study several alternative ways of creating such graphs by using the connectivity information available in Wikipedia. After creating these graphs, we extract information from them using link analysis algorithms. In this Section we describe how each of these graphs is constructed and the link analysis algorithms we used.

### 5.1.1 Full Local Context Graphs

The first step of our approach consists of building a graph representing all the concepts present in the text and the relationships between them. This graph is called *Full Local Context Graph.* The Full Local Context Graph, which is also referred by its acronym:

$flcg$, generated to represent the text $t$, is defined as:

$$G_{flcg}(t) = \langle V_{flcg}, E_{flcg} \rangle$$

The set $V_{flcg}$ of vertices is determined as follows:

a) all the *n-grams* present in the text are extracted and compose the initial set of vertex candidates;

b) *n-grams* containing punctuation marks between two of their words are removed from this initial set;

c) the *n-grams* that do not match the title of any Wikipedia article in the normalized form are also removed from the set;

d) the resulting set is then represented by the set of vertices $V_{flcg}$.

To obtain the normalized form $norm(t)$ of a Wikipedia article title $t$, we remove its disambiguation field, if it exists, as well as any accent marks it may have, and convert it to lower case. For example:

- $norm(House\_(series)) = house$
- $norm(Tiger\_(animal)) = tiger$

To reduce noise we also discard disambiguation articles and articles which the normalized title is a stopword[4] or are formed by only one single character.

The set of vertices of a graph represents all the possible concepts present in a short text, but we can also extract from Wikipedia information about how these concepts relate to each other to create the set of edges of the Context Graph. We generate the set of edges $E_{flcg}$ as follows: as each vertex $v$ represents a Wikipedia article $w(v)$, there is an edge $e = (v, u)$ from a vertex $v$ to a vertex $u$ if, and only if, there is a link in Wikipedia from

[4] Stopwords are terms with high frequency and low importance, such as 'the', 'a', 'or'. The list of stopwords can vary according to the language considered

$w(v)$ to $w(u)$. To avoid mutual reinforcement by articles originated from the same *n-gram*, we discard links between articles with the same normalized title, like the examples below:

- $Apple\_(company)$ and $Apple\_(fruit)$
- $Java\_(programming\_language)$ and $Java\_(island)$
- $Beast\_(x - men\_character)$ and $Beast\_(disney\_character)$

The resulting set of edges and vertices is the graph we call $flcg$. Figure 5.1 shows an example of a Full Local Context Graph generated for the short text *"I was happy to welcome Her Majesty Queen Elizabeth II to the UAE. We share a strong relationship w Britain based on a friendship and common goals."*. In this example, we show the 10 n-grams that matched Wikipedia titles, thus generating 10 distinct vertices. The connection between vertices "Elizabeth II" and "Queen" means that there is a link in the Wikipedia article with title "Elizabeth II" pointing to the Wikipedia article with title "Queen".

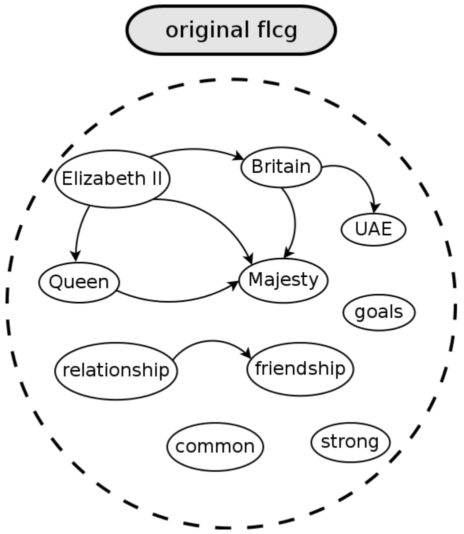


Figure 5.1: Full Local Context Graph generated from the text: *"I was happy to welcome Her Majesty Queen Elizabeth II to the UAE. We share a strong relationship w Britain based on a friendship and common goals."*

### 5.1.2 Degree Local Context Graphs

Some concepts from a short text may not be related to its main topic. We observed that most of the time these irrelevant concepts do not connect to any other concept in the graph (i.e. they have no incoming nor outgoing edges). For this reason, we propose the *Degree Local Context Graph* (also referred to as *dlcg*) which is a variation of the $flcg$ where all the vertices with degree equal to zero (i.e. with no incoming or outgoing edge.) are removed from the vertex set.

The *dlcg* built from text $t$ can be defined as:

$$G_{dlcg}(t) = \langle V_{dlcg}, E_{dlcg} \rangle, \text{ where } E_{dlcg} = E_{flcg}$$

$$V_{dlcg} = \{u \in V_{dlcg} | (u,v) \vee (v,u) \in E_{dlcg}\}$$

Figure 5.2 shows an example of a $flcg$ and its respective *dlcg* generated for the same text as Figure 5.1. In this example, vertices “common”, “goals” and “strong” are not present on the Degree Local Context Graph, since they represent articles that neither point nor are pointed to by other articles represented in the $G_{flcg}$ derived from the sample text.

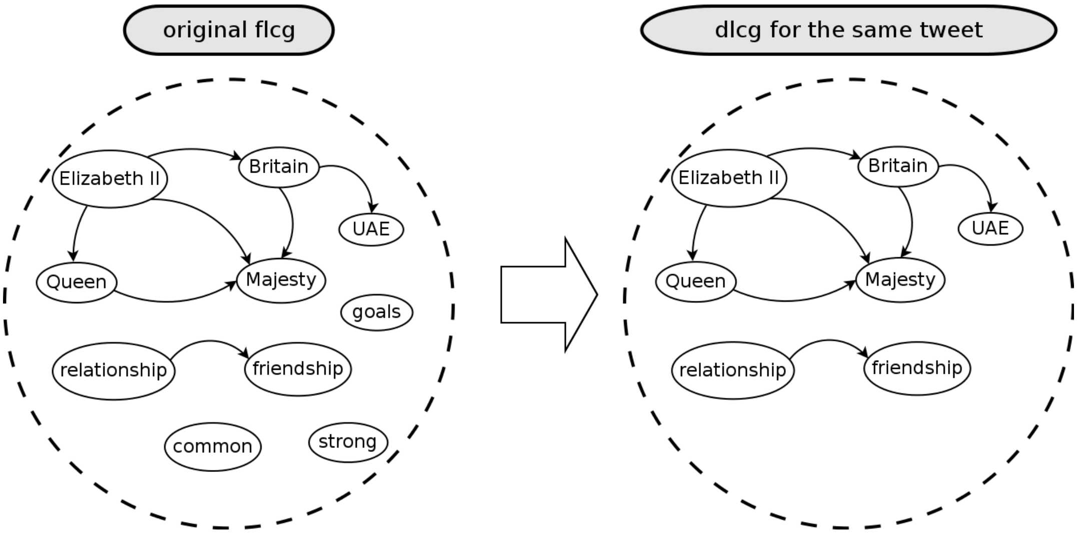


Figure 5.2: flcg and dlcg for the same text as Figure 5.1 .

### 5.1.3 Connected Local Context Graphs

In some cases, there are two or more sets of concepts related to each other in the *dlcg* which may not have any relation to the main topic of the text. To avoid picking up such small sets of spurious concepts we also propose *Connected Local Context Graph* (also referred to as *clcg*). It is a Context Graph containing only the biggest connected component, i.e. the biggest set of concepts in which we should have a path from one concept to another if it were an undirected graph.

More specifically, the *clcg* of a text $t$ can be defined as the biggest connected subgraph of the $flcg$. A directed graph is said to be connected if the undirected underlying graph obtained by replacing its directed edges with undirected edges is a connected undirected graph.

Figure 5.3 shows an example of a $flcg$ and its respective *clcg* generated for the same text as Figure 5.1 .

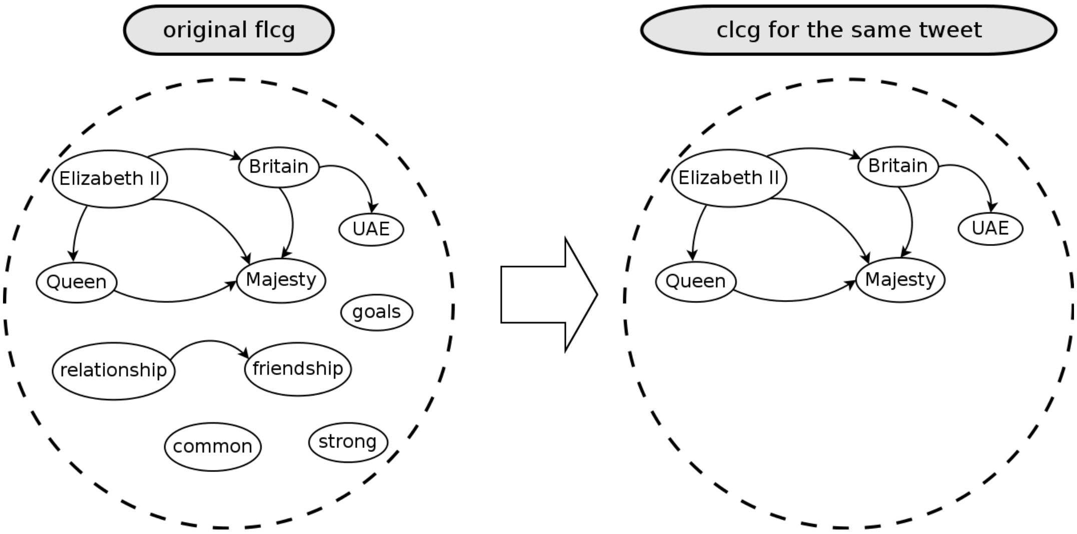


Figure 5.3: flcg and clcg for the same text as Figure 5.1 .

### 5.1.4 Strong Local Context Graphs

The relationship between concepts is not always transitive, which means that sometimes there is a relationship between concepts $a$ and $b$, and there is also a relationship between concepts $b$ and $c$ but there is no relationship between $a$ and $c$. Such behaviour may result on including unrelated terms in the Context Graph. To deal with these situations, we propose the *Strong Local Context Graph*, or *slcg*. The *slcg* is a graph with the biggest set of vertices where all pairs of vertices have at least one edge to or from each other.

The *slcg* of the text $t$ can be defined as $G_{slcg}(t) = \langle V_{slcg}, E_{slcg} \rangle$. Such that $V_{slcg}$ is the maximal set of vertices, where for all pair of vertices $u, v \in V_{slcg}$, $(u, v) \in E_{flcg} \vee (v, u) \in E_{flcg}$.

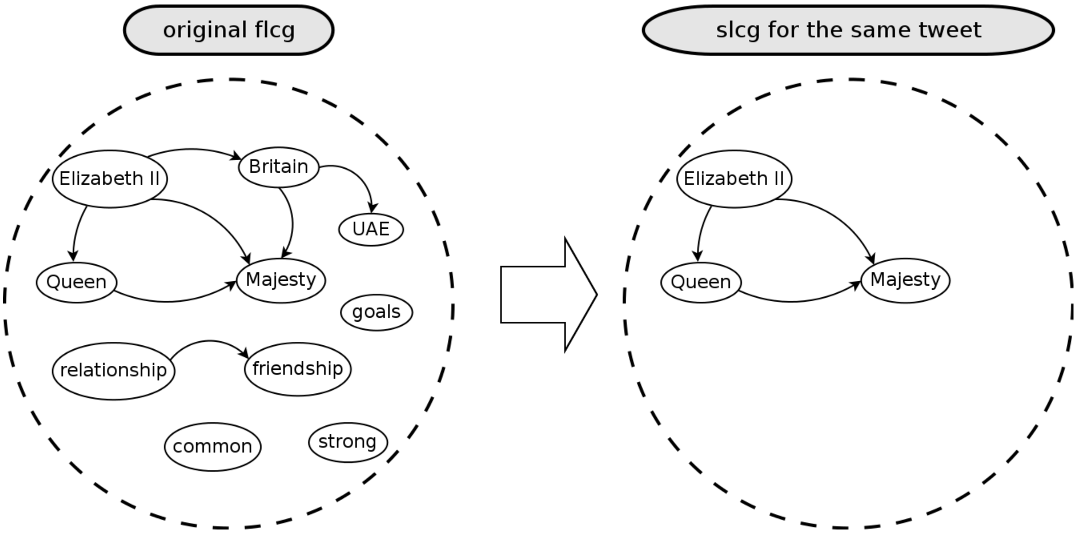


Figure 5.4: flcg and slcg for the same text as Figure 5.1 .

Figure 5.4 shows an example of a $flcg$ and its respective *slcg* generated for the same text as Figure 5.1.

### 5.1.5 eXpanded Full Local Context Graph

Some Wikipedia pages have no content themselves but redirect the user to another page where the information is actually available. Such pages are called *redirects* and are very helpful especially when a single concept can be described in several different ways. Some examples of redirect pages are:

- *Apple_Computer_Inc.*, *Apple_Computer* and *Apple_Inc.* redirect to *Apple*.
- *Stanley_Martin_Lieber* redirects to *Stan_Lee*.

- $Einstein$ redirects to $Albert\_Einstein$.

- $Digital\_Video\_Disk$ and $DVD\_Players$ redirect to $DVD$.

To take advantage of this information, we propose another set of Context Graphs called *eXpanded Local Context Graphs*.

The eXpanded Full Local Context Graph (a.k.a. $xflcg$) is similar to the $flcg$ with the difference that it also represents as vertices all the Wikipedia articles that redirect to or from any vertex present on $V_{flcg}$. We can define the $xflcg$ as:

$G_{xflcg}(t) = \langle V_{xflcg}, E_{xflcg} \rangle$. $v \in V_{xflcg}$, if $(v \in V_{flcg}) \vee (w(v)$ redirects to or from $w(u)$ and $u \in V_{flcg})$.

The set of edges $E_{xflcg}$ is defined as $(u, v) \in E_{xflcg}$ if $u \in V_{xflcg}$, $v \in V_{xflcg}$ and $norm(u) \neq norm(v)$.

Likewise, we can also define extended counterparts of the other types of local graphs we have defined, namely eXpanded Degree Local Context Graph (*xdlcg*), eXpanded Connected Local Context Graph (*xclcg*) and eXpanded Strong Local Context Graph (*xslcg*), which are expanded versions of the *dclg*, *clcg* and *slcg* respectively. All these expanded graphs are derived from the $xflcg$ exactly the same way their respective non-expanded versions are derived from the $flcg$.

### 5.1.6 Link Analysis

After modelling the short texts as Context Graphs, we use link analysis algorithms to rank their vertices, thus obtaining us an ordered list of the keywords of each text. Such algorithms evaluate the relationships (represented by the edges) among the vertices of a graph to predict a ranking where the most popular vertices are placed at the top. In this study each link analysis algorithm receives as input a Context Graph and outputs a ranking of keywords. The algorithms we used are:

- Indegree [4]. This is a naive but effective technique. Indegree gives a score to the vertices of a graph according to the number of edges pointing to them. In this study we experimented with the local indegree, where we consider only the edges of the Context Graph to compute the score of each vertex (referred to as *ind*) and also with the global indegree, which takes into account all the Wikipedia links that point to the vertex's article (referred to as global indegree, or *gind*).

- HITS [19]. In this algorithm, each vertex has two scores: hubs and authorities. They are based on a recursive assumption: to be considered as a good hub, a vertex must point to good authorities and to be considered a good authority, a vertex must point to good hubs. In this study, we experimented with both scores (hubs and authorities, referring to them respectively to as *hub* and *auth*) to rank keywords.
- Pagerank [5] (also referred to as *pr*). In this algorithm the score of each vertex is calculated taking into account the scores of the vertices pointing to it in a recursive way. The Pagerank of a vertex is propagated to the vertices it points to. Pagerank is also a robust technique and has the advantage of summarizing the score of each vertex into one single value.

### 5.1.7 Machine Learning Framework

As link analysis algorithms associate a score with each vertex in a graph, we can obtain several different values to assess the importance of a keyword by using different link analysis algorithms and different types of Context Graphs to represent the short text the concept belongs to. All these scores may carry important information about the relevance of the concept in the text. For this reason, we propose a machine learning framework to use all of them, in order to obtain a better performance when trying to figure out what concepts better describe a short text.

The proposed approach is based on the work by [24] and consists of two steps: building the initial ranking and applying a selective filter to eliminate the spurious concepts from this ranking. To build the initial ranking we use a high-coverage method to avoid excluding any important concept from this initial set. After this, we train a Random Forest Classifier to predict which concepts are relevant or are not using a set of features to describe them. Figure 5.5 illustrates this process.

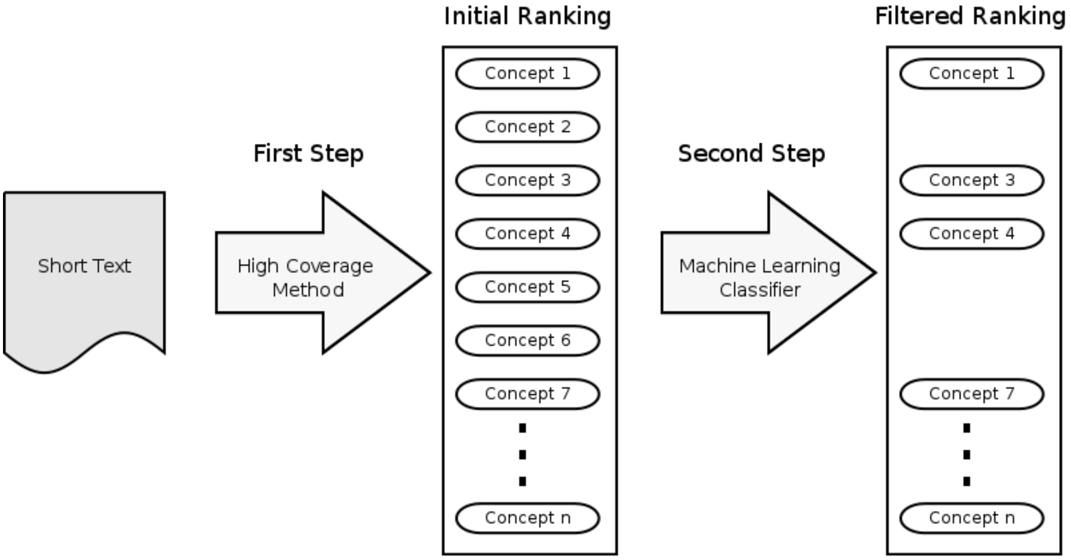


Figure 5.5: Using a Random Forest Classifier as a filter.

Choosing a representative set of features is one of the most important challenges faced when dealing with learning-based strategies. The features we used in this study can be grouped into three main sets, described as follows.

### Link Analysis Features

The user of link analysis algorithms with Context Graphs results in several different values associated with each concept from the text. In this work, we used all the possible combinations among the algorithms described in Section 5.1.6 over the graphs proposed on Sections 5.1.1 to 5.1.5. All these combinations are shown in Table 5.1.

### Graph Centrality/Vitality Features

Besides Link Analysis, we also used Centrality and Vitality algorithms to describe the importance of a vertex in a graph. The algorithms we use are:

- Closeness Vitality [3]. The Closeness Vitality of a vertex $v$ in graph $G$ stands for

| | Global Indegree | Indegree | Pagerank | HITS: Hub | Hits: Authority |
|---|---|---|---|---|---|
| flcg | gind(flcg) | ind(flcg) | pr(flcg) | hh(flcg) | ha(flcg) |
| dlcg | gind(dlcg) | ind(dlcg) | pr(dlcg) | hh(dlcg) | ha(dlcg) |
| clcg | gind(clcg) | ind(clcg) | pr(clcg) | hh(clcg) | ha(clcg) |
| slcg | gind(slcg) | ind(slcg) | pr(slcg) | hh(slcg) | ha(slcg) |
| xflcg | gind(xflcg) | ind(fxlcg) | pr(xflcg) | hh(xflcg) | ha(xflcg) |
| xdlcg | gind(xdlcg) | ind(xdlcg) | pr(xdlcg) | hh(xdlcg) | ha(xdlcg) |
| xclcg | gind(xclcg) | ind(xclcg) | pr(xclcg) | hh(xclcg) | ha(xclcg) |
| xslcg | gind(xslcg) | ind(xslcg) | pr(xslcg) | hh(xslcg) | ha(xslcg) |

Table 5.1: Combinations of link analysis algorithms and graphs used to generated this set of features.

the change in the sum of distances between all the vertices of $G$ if $v$ is removed from $G$.

- Closeness Centrality [12]. The Closeness Centrality of a vertex $v$ is the inverse of the average distance from $v$ to all other vertices in the graph.
- Communicability Centrality [9]. The Communicability Centrality of a vertex $v$ is the sum of all closed walks starting and ending at $v$.
- Load Centrality [29]. The Load Centrality of vertex $v$ in graph $G$ is the fraction of all the shortest paths in $G$ that include $v$.

As each of these algorithms produces a different score for each vertex according to the Context Graph adopted, we used all the possible combinations between algorithms and graphs as features, thus resulting in a set of 32 features.

### Baseline Features

To enrich the information obtained using the Context Graphs proposed in this work, we also used a set of features proposed on [24]. The authors perform a solid feature analysis over the set of features they adopted, we choose to use only the top features they recommended. Considering that each n-gram $n$ is associated with a set of concepts $S = \{c_1, c_2, c_3, ..., c_i\}$ and each concept $c_j$ is related to a Wikipedia article $w(c_j)$, the set of features we used is detailed in Table 5.2.

| **Feature** | **Description** |
|---|---|
| $IDF_{content}(n)$ | Inverse Document Frequency of $n$ in the content of Wikipedia articles. |
| $IDF_{title}(n)$ | Inverse Document Frequency of $n$ in the title of Wikipedia articles. |
| $IDF_{anchor}(n)$ | Inverse Document Frequency of $n$ in the anchor text of Wikipedia articles. |
| $TF_{paragraph}(n, c)$ | Relative frequency of $n$ in the first paragraph of $w(c)$. |
| $TF_{sentence}(n, c)$ | Relative frequency of $n$ in the first sentence of $w(c)$. |
| $TF_{title}(n, c)$ | Relative frequency of $n$ in the title of $w(c)$. |
| $TCN(n, c)$ | True if the title of $w(c)$ contains $n$. |
| $TEN(n, c)$ | True if the title of $w(c)$ is equal to $n$. |
| $TWCT(c, S)$ | True if the short text $S$ contains the title of $w(c)$. |
| $REDIRECT(c)$ | Number of redirect articles pointing to $w(c)$. |
| $LINKPROB(n)$ | Probability of $n$ being used as an anchor text in Wikipedia (considering all the occurrences). |
| $KEYPHRASENESS(q)$ | Probability of $n$ being used as an anchor text in Wikipedia. |
| $SNIL(n)$ | Number of Wikipedia articles whose title is equal to a sub-n-gram of $n$. |
| $SNCL(n)$ | Number of Wikipedia articles whose title matches a sub-n-gram of $n$. |
| $POS_1(n, c)$ | Position of the first occurrence of $n$ in $c$. |
| $GEN(c)$ | Function of depth of $w(c)$ in Wikipedia category hierarchy. |
| $COMMONNESS(n, c)$ | Probability of $c$ being the target of a link with anchor text equals to $n$. |

Table 5.2: Best features from [24].

Features $URL$ (which stands for the occurrence of the n-grams in any web page cited in the short text) and $WIG$ (which stands for the weighted information gain of the concepts associated with the n-grams) were not used in order to reduce the costs, as recommended by [24].

## 5.2 Experiments

In this Section, we present the experiments we performed to evaluate our proposed approach. We begin by presenting the data sets we used.

The results of the proposed methods are presented in two Sections. First we evaluate the performance of each method proposed individually, and then we use the methods to generate the features which will represent the concepts provided as input to the machine learning framework based on a Random Forest Classifier.

Another experiment we performed was using the same approach proposed on Chapter 4 to train the classifier to pick the keywords which are more likely to retrieve good products.

### 5.2.1 Datasets

In our experiments we used two collections of short texts: *data_br* and *data_en*. The first is a collection of 1,000 posts from the 100 Brazilian profiles on twitter with the highest number of followers according to TweetRank[5]. It includes mostly celebrities, artists, journalists and news profiles. For each profile we collected the last 10 tweets posted before May, 2012. As external source of knowledge, we adopted a collection of Brazilian Wikipedia articles downloaded from Wikimedia[6] on May, 2012. This collection contains $1,286,688$ articles with $42,156,867$ links between them. From these articles, $553,056$ redirect to another one.

The second collection used, *data_en*, is the same adopted by [24] which was originally composed by 562 tweets from random *verified accounts*[7]. As some of these tweets were no longer available at the time we ran the experiments, the total number of tweets was then reduced to 375. As Wikipedia data, we used another set of Wikipedia articles downloaded from Wikimedia, including $9,304,901$ articles with $131,362,508$ links between them. From these articles, $5,438,875$ are redirections. This collection is referred to in this work as *data_en*.

To evaluate the performance of the methods in each dataset we used $p@1$, $MRR$, $Prec$, $Rec$ and $F1$, the same set of metrics adopted by the baseline [24]. $Prec$ and $Rec$ are respectively the precision and recall considering the whole set of keywords returned by each method.

[5] http://www.tweetrank.com.br/

[6] http://wikimedia.org

[7] A verified account is a Twitter profile that received from Twitter a certified of authenticity

### 5.2.2 Results of Each Proposed Method Individually

Table 5.3 presents the results of each proposed method in the *data_br* dataset. We can see that the link analysis algorithms applied to the *slcg* produced results with the higher precision values. However, the recall achieved while using these Context Graphs is extremely low, and therefore, they have the worst $F1$ among all the methods.

| Method | P@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(flcg) | 0.5928 | **0.7142** | 0.3119 | **0.6077** | **0.4122** |
| gout(flcg) | 0.4718 | **0.6107** | 0.3119 | **0.6061** | **0.4119** |
| gin(dlcg) | 0.6684 | 0.1995 | 0.5528 | 0.0854 | 0.1479 |
| gout(dlcg) | 0.6580 | 0.1984 | 0.5528 | 0.0854 | 0.1479 |
| auth(dlcg) | 0.6321 | 0.1946 | 0.5528 | 0.0854 | 0.1479 |
| hub(dlcg) | 0.5389 | 0.1783 | 0.5528 | 0.0854 | 0.1479 |
| ind(dlcg) | 0.6477 | 0.1970 | 0.5528 | 0.0854 | 0.1479 |
| out(dlcg) | 0.5389 | 0.1786 | 0.5528 | 0.0854 | 0.1479 |
| pr(dlcg) | 0.6425 | 0.1965 | 0.5528 | 0.0854 | 0.1479 |
| gind(clcg) | 0.6528 | 0.1950 | 0.5624 | 0.0802 | 0.1404 |
| gout(clcg) | 0.6632 | 0.1961 | 0.5624 | 0.0802 | 0.1404 |
| auth(clcg) | 0.6321 | 0.1908 | 0.5624 | 0.0802 | 0.1404 |
| hub(clcg) | 0.5596 | 0.1789 | 0.5624 | 0.0802 | 0.1404 |
| ind(clcg) | 0.6528 | 0.1949 | 0.5624 | 0.0802 | 0.1404 |
| out(clcg) | 0.5389 | 0.1774 | 0.5624 | 0.0802 | 0.1404 |
| pr(clcg) | 0.6528 | 0.1952 | 0.5624 | 0.0802 | 0.1404 |
| gind(slcg) | 0.7333 | 0.0168 | **0.7111** | 0.0070 | 0.0139 |
| gout(slcg) | 0.7333 | 0.0168 | **0.7111** | 0.0070 | 0.0139 |
| auth(slcg) | 0.7333 | 0.0166 | **0.7111** | 0.0070 | 0.0139 |
| hub(slcg) | 0.6000 | 0.0152 | **0.7111** | 0.0070 | 0.0139 |
| ind(slcg) | **0.8667** | 0.0182 | **0.7111** | 0.0070 | 0.0139 |
| out(slcg) | 0.6667 | 0.0161 | **0.7111** | 0.0070 | 0.0139 |
| pr(slcg) | **0.8667** | 0.0182 | **0.7111** | 0.0070 | 0.0139 |

Table 5.3: Performance of indegree, outdegree, global indegree, global outdegree, pagerank and HITS for each Context Graph in the *data_br* dataset. Best results for each metric are in bold.

In this dataset, the results do not fluctuate when we change the link analysis algorithm over *dlcg*, *clcg* and *slcg* because the number of keywords returned by these methods is often smaller than the value we set to be the maximal output considered for each method (the maximum size of the set we used for the set was 50). For this reason, even if the order in which the keywords are ranked changes, the list of keywords remains the same and the metrics that consider the output of the method as a set of keywords (*Prec*, *Rec* and $F1$) give the same value.

The *dlcg* and *clcg* Context Graphs achieved very similar results. This happens because the average size of the Context Graphs is quite small, thus in practice both end up to be very similar. Also, we can observe that using these Context Graphs improves $p$@1 as well as the overall precision while decreasing recall in comparison with the $flcg$ graph. The best $MRR$ values are also obtained with $flcg$ because the other Context Graphs have lower recall values. This reflects the fact that the other graphs are subgraphs of the $flcg$ and end up not selecting any concepts. When there are several posts from which they do not select any keyword, it results in a reciprocal rank of 0, which decreases the final MRR value.

In Table 5.4 we show the results of the proposed methods for the *data_en* dataset. Similarly to the results presented in Table 5.3 we can see that the *slcg* graph achieves the best precision (P@1 and Prec) and the worse recall values at the same time. This reaffirms our idea that they increase precision while significantly decreasing recall.

Another conclusion we can draw is that the *dlcs* and *clcs* Context Graphs can yield better precision values in comparison with the $flcg$. This is due to the fact that connected concepts indeed are most related to the main topic of the post.

We present the results obtained with the expanded version of our Context Graphs in Tables 5.5 and 5.6. In these scenarios, the results obtained using *xdlcg* and *xclcg* are quite similar, confirming the aforementioned hypothesis that they are very similar graphs.

The expanded versions of the strong Context Local Graphs also presented the best precision and the worst recall values, confirming the behaviour of their *slcg* counter-part. The best *recall* is obtained by $xflcg$ with a large gain over the other graphs. It results in a better $MRR$ and $F1$ as well.

By comparing the results achieved by the expanded graphs ($xflcg$, *xdlcg*, *xclcg* and *xslcg*) with their original versions ($flcg$, *dlcg*, *clcg* and *slcg*) we can see that expanding the Context Graphs by including redirect information improves recall while decreasing the precision. This happens because the expanded versions of the graphs have more relevant

| Method | P@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(flcg) | 0.4731 | **0.5835** | 0.1144 | **0.3822** | **0.1761** |
| gout(flcg) | 0.3441 | **0.4539** | 0.1173 | **0.3887** | **0.1802** |
| gin(dlcg) | 0.4719 | 0.2724 | 0.3831 | 0.0660 | 0.1126 |
| gout(dlcg) | 0.3989 | 0.2490 | 0.3824 | 0.0659 | 0.1124 |
| auth(dlcg) | 0.4943 | 0.2735 | 0.3830 | 0.0643 | 0.1101 |
| hub(dlcg) | 0.3409 | 0.2235 | 0.3837 | 0.0648 | 0.1109 |
| ind(dlcg) | 0.5056 | 0.2808 | 0.3815 | 0.0652 | 0.1114 |
| out(dlcg) | 0.3427 | 0.2271 | 0.3824 | 0.0659 | 0.1124 |
| pr(dlcg) | 0.4877 | 0.2519 | 0.3885 | 0.0552 | 0.0967 |
| gind(clcg) | 0.4551 | 0.2562 | 0.3800 | 0.0533 | 0.0935 |
| gout(clcg) | 0.3933 | 0.2379 | 0.3772 | 0.0527 | 0.0925 |
| auth(clcg) | 0.5000 | 0.2644 | 0.3768 | 0.0518 | 0.0911 |
| hub(clcg) | 0.3295 | 0.2137 | 0.3768 | 0.0518 | 0.0911 |
| ind(clcg) | 0.5056 | 0.2690 | 0.3770 | 0.0525 | 0.0922 |
| out(clcg) | 0.3427 | 0.2214 | 0.3772 | 0.0527 | 0.0925 |
| pr(clcg) | 0.5031 | 0.2474 | 0.3821 | 0.0442 | 0.0792 |
| gind(slcg) | 0.5000 | 0.0153 | **0.4667** | 0.0024 | 0.0048 |
| gout(slcg) | 0.4000 | 0.0140 | **0.4667** | 0.0024 | 0.0048 |
| auth(slcg) | **0.6000** | 0.0170 | **0.4667** | 0.0024 | 0.0048 |
| hub(slcg) | 0.4000 | 0.0144 | **0.4667** | 0.0024 | 0.0048 |
| ind(slcg) | **0.6000** | 0.0170 | **0.4667** | 0.0024 | 0.0048 |
| out(slcg) | 0.4000 | 0.0144 | **0.4667** | 0.0024 | 0.0048 |
| pr(slcg) | **0.6000** | 0.0170 | **0.4667** | 0.0024 | 0.0048 |

Table 5.4: Performance of indegree, outdegree, global indegree, global outdegree, pagerank and HITS for each Context Graph in the *data_en* dataset. Best values for each metric are presented in bold.

concepts but also more noise.

In Table 5.7 we summarize the best results obtained with the proposed approach and compare them to the best baselines presented by [24]: Spotlight[8], which is a tool for automatic annotating concepts from DBpedia in fragments of text. MW, a machine learning approach to identify significant terms within unstructured text, and enrich it with links to the appropriate Wikipedia articles. Tagme, which use anchor texts from Wikipedia to annotate texts and CMNS (also referred as Commonness), which scores each concept based on the relative frequency with which the n-gram is used as an anchor text for that particular concept. For a fair comparison, we included in this table only the methods that do not require learning and also adopted the same dataset the authors used, *data_en*.

[8] http://spotlight.dbpedia.org/

| Method | P@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(xflcg) | 0.5378 | **0.6810** | 0.2586 | **0.7737** | **0.3876** |
| gout(xflcg) | 0.4195 | **0.5826** | 0.2575 | **0.7680** | **0.3857** |
| gin(xdlcg) | 0.5926 | 0.2565 | 0.4923 | 0.1120 | 0.1825 |
| gout(xdlcg) | 0.5852 | 0.2541 | 0.4923 | 0.1120 | 0.1825 |
| auth(xdlcg) | 0.5963 | 0.2549 | 0.4923 | 0.1120 | 0.1825 |
| hub(xdlcg) | 0.4815 | 0.2291 | 0.4923 | 0.1120 | 0.1825 |
| ind(xdlcg) | 0.5926 | 0.2562 | 0.4933 | 0.1122 | 0.1828 |
| out(xdlcg) | 0.5074 | 0.2361 | 0.4923 | 0.1120 | 0.1825 |
| pr(xdlcg) | 0.5926 | 0.2555 | 0.4933 | 0.1122 | 0.1828 |
| gind(xclcg) | 0.5926 | 0.2530 | 0.4999 | 0.1019 | 0.1693 |
| gout(xclcg) | 0.5889 | 0.2512 | 0.4999 | 0.1019 | 0.1693 |
| auth(xclcg) | 0.5926 | 0.2504 | 0.4999 | 0.1019 | 0.1693 |
| hub(xclcg) | 0.4778 | 0.2260 | 0.4999 | 0.1019 | 0.1693 |
| ind(xclcg) | 0.6000 | 0.2539 | 0.4999 | 0.1019 | 0.1693 |
| out(xclcg) | 0.4963 | 0.2307 | 0.4999 | 0.1019 | 0.1693 |
| pr(xclcg) | 0.6000 | 0.2544 | 0.4999 | 0.1019 | 0.1693 |
| gind(xslcg) | **0.6563** | 0.0336 | **0.6250** | 0.0124 | 0.0243 |
| gout(xslcg) | 0.6250 | 0.0329 | **0.6250** | 0.0124 | 0.0243 |
| auth(xslcg) | 0.6250 | 0.0327 | **0.6250** | 0.0124 | 0.0243 |
| hub(xslcg) | 0.5625 | 0.0309 | **0.6250** | 0.0124 | 0.0243 |
| ind(xslcg) | 0.6875 | 0.0341 | **0.6250** | 0.0124 | 0.0243 |
| out(xslcg) | 0.5938 | 0.0313 | **0.6250** | 0.0124 | 0.0243 |
| pr(xslcg) | 0.6563 | 0.0334 | **0.6250** | 0.0124 | 0.0243 |

Table 5.5: Performance of indegree, outdegree, global indegree, global outdegree, Pagerank and HITS for each expanded Context Graph in the *data_br* dataset. Best values for each metric are presented in bold.

The difference in performance between the methods means that to achieve better results, the choice of the most appropriate method has to be guided by the the final objective of the target application. If it is necessary to pick only one keyword and develop a precision-oriented application, the *MW*, *Tagme* and *CMNS* methods may present the best performance, although *CMSN* also provides a better ranking than the others, resulting in a higher *MRR*. If the application requires selecting a more accurate set of keywords, then *MW* can be a good choice, yielding the highest *Prec* value. Among the methods we propose, we can highlight $gind(xflcg)$, which achieved the best *Rec* and $F1$ at the same time. So, if the application is recall-oriented or even if it requires a good balance between recall and precision, these methods are more appropriate.

These results shows that it is possible to achieve a very interesting performance using

| Method | P@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(xflcg) | 0.4113 | **0.5486** | 0.1330 | **0.5487** | **0.2141** |
| gout(xflcg) | 0.3495 | **0.4730** | 0.1279 | **0.4985** | **0.2036** |
| gin(xdlcg) | 0.4524 | 0.3827 | 0.3226 | 0.1174 | 0.1722 |
| gout(xdlcg) | 0.4206 | 0.3664 | 0.3225 | 0.1172 | 0.1719 |
| auth(xdlcg) | 0.4777 | 0.3839 | 0.3238 | 0.1123 | 0.1668 |
| hub(xdlcg) | 0.3279 | 0.3118 | 0.3239 | 0.1125 | 0.1670 |
| ind(xdlcg) | 0.4841 | 0.3949 | 0.3225 | 0.1172 | 0.1719 |
| out(xdlcg) | 0.3373 | 0.3259 | 0.3225 | 0.1171 | 0.1718 |
| pr(xdlcg) | 0.4279 | 0.3363 | 0.3171 | 0.1037 | 0.1563 |
| gind(xclcg) | 0.4405 | 0.3623 | 0.3369 | 0.0912 | 0.1435 |
| gout(xclcg) | 0.3968 | 0.3447 | 0.3369 | 0.0910 | 0.1433 |
| auth(xclcg) | 0.4498 | 0.3623 | 0.3373 | 0.0882 | 0.1398 |
| hub(xclcg) | 0.3253 | 0.3047 | 0.3373 | 0.0882 | 0.1398 |
| ind(xclcg) | 0.4484 | 0.3679 | 0.3369 | 0.0910 | 0.1433 |
| out(xclcg) | 0.3413 | 0.3182 | 0.3368 | 0.0909 | 0.1432 |
| pr(xclcg) | 0.4454 | 0.3314 | 0.3337 | 0.0792 | 0.1280 |
| gind(xslcg) | 0.6207 | 0.0541 | **0.5460** | 0.0108 | 0.0212 |
| gout(xslcg) | 0.4828 | 0.0489 | **0.5460** | 0.0108 | 0.0212 |
| auth(xslcg) | **0.6552** | 0.0554 | **0.5460** | 0.0108 | 0.0212 |
| hub(xslcg) | 0.4483 | 0.0471 | **0.5460** | 0.0108 | 0.0212 |
| ind(xslcg) | 0.6207 | 0.0541 | **0.5460** | 0.0108 | 0.0212 |
| out(xslcg) | 0.4483 | 0.0476 | **0.5460** | 0.0108 | 0.0212 |
| pr(xslcg) | 0.5517 | 0.0515 | **0.5460** | 0.0108 | 0.0212 |

Table 5.6: Performance of indegree, outdegree, global indegree, global outdegree, Pagerank and HITS for each expanded Context Graph in the *data_en* dataset. Best values for each metric are presented in bold.

| Method | p@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(flcg) | 0.4731 | 0.5835 | 0.1144 | 0.3822 | 0.1761 |
| ind(dlcg) | 0.5056 | 0.2808 | 0.3815 | 0.0652 | 0.1114 |
| gind(xflcg) | 0.4113 | 0.5486 | 0.1330 | 0.5487 | 0.2141 |
| ind(xdlcg) | 0.4841 | 0.3949 | 0.3225 | 0.1172 | 0.1719 |
| Spotlight | 0.4389 | 0.4154 | 0.3653 | 0.0828 | 0.1350 |
| MW | 0.6167 | 0.4154 | 0.4256 | 0.0969 | 0.1579 |
| Tagme | 0.6006 | 0.6233 | 0.2997 | 0.1556 | 0.2048 |
| CMNS | 0.4946 | 0.6344 | 0.0685 | 0.2269 | 0.1052 |

Table 5.7: Performance of the best proposed methods and baselines in the *data_en* dataset.

the proposed approach. Also, the link analysis algorithms, which are based on incoming links (*gin*, *ind*, *pr* and *auth*), have proven to be often better than the ones based on outgoing links (*gout*, *out* and *hub*). Another interesting conclusion we drawn is that each rank obtained by each of the link analysis algorithms based on incoming links is very similar to each other. This is a consequence of the small average size of the Context

Graphs.

### 5.2.3 Results Using the Machine Learning Framework

As described in Section 5.1, we also used the value obtained by each method proposed as an individual feature in a machine learning framework. As initial set we used *gind*($flcg$) and *gind*($xflcg$) because of the high recall they achieved. Then we used Random Forest as classifiers and a 10-fold cross-validation methodology, using 8 folds to train, 1 fold to validate and 1 fold to test the results. The parameters were tuned on the validation set and the results reported are the average of the test sets.

In Tables 5.8 and 5.9 we present the results obtained by each original method, as well as the results achieved by the machine learning framework using different metrics as optimization objective. For example, $gind(flcg) - RF - p1$ stands for the results obtained by Random Forests while using the ranking produced by *gind*($flcg$) as initial ranking and the precision at rank 1 as the metric to be optimized.

| Method | Prec@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(flcg) | 0.4731 | 0.5835 | 0.1144 | 0.3822 | 0.1761 |
| gind(flcg)-RF-p1 | 0.6950 | 0.6310 | 0.6097 | 0.1390 | 0.2264 |
| gind(flcg)-RF-mrr | 0.6925 | 0.6388 | 0.6074 | 0.1401 | 0.2277 |
| gind(flcg)-RF-prec | 0.7087 | 0.6193 | 0.6215 | 0.1262 | 0.2098 |
| gind(flcg)-RF-rec | 0.6657 | 0.6419 | 0.5822 | 0.1470 | 0.2347 |
| gind(flcg)-RF-f1 | 0.6637 | 0.6442 | 0.5836 | 0.1474 | 0.2354 |

Table 5.8: Application of machine learning to filter results produced by the gind(flcg) ranking in the *data_en* dataset.

| Method | Prec@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(xflcg) | 0.4113 | 0.5486 | 0.1330 | 0.5487 | 0.2141 |
| gind(xflcg)-RF-p1 | 0.6440 | 0.6227 | 0.5728 | 0.1693 | 0.2614 |
| gind(xflcg)-RF-mrr | 0.6273 | 0.6322 | 0.5554 | 0.1803 | 0.2722 |
| gind(xflcg)-RF-prec | 0.6449 | 0.6126 | 0.5707 | 0.1513 | 0.2392 |
| gind(xflcg)-RF-rec | 0.6202 | 0.6387 | 0.5525 | 0.1894 | 0.2821 |
| gind(xflcg)-RF-f1 | 0.6320 | 0.6427 | 0.5431 | 0.1866 | 0.2778 |

Table 5.9: Application of machine learning to filter results produced by the gind(xflcg) ranking in the *data_en* dataset.

Analysing the results, we can see that the use of Random Forest to filter the results improves the results obtained by the methods. $p@1$ improves by $46,90\%$ with non-expanded graphs and $56,58\%$ with the expanded version of the Context Graphs. With $MRR$ we achieved an improvement between $9,48\%$ and $15,24\%$ over the non-trained-based approach. Precision is the measure where we achieved the best improvements, about 4 to 5 times better. Because we used a classifier as a filter, *recall* was the only metric that suffered a performance loss with less than 40% of the original performance. Nonetheless, even with such a loss we achieved an improvement in the $F1$ values of 33.67% and 29.75%.

Table 5.10 presents a comparison between the results obtained with our machine learning framework using the proposed methods as features and the best results obtained by [24]. $CMNS$ and the proposed methods presented the best $Prec$@1 results in comparison with other methods. The $MRR$ achieved by the $CMNS$-based methods is about 16% better than the results obtained with our methods.

| Method | Prec@1 | MRR | Prec | Rec | F1 |
|---|---|---|---|---|---|
| gind(flcg)-RF-f1 | 0.6637 | 0.6442 | 0.5836 | 0.1474 | 0.2354 |
| gind(xflcg)-RF-f1 | 0.6320 | 0.6427 | 0.5431 | 0.1866 | 0.2778 |
| CMNS-GBRT | 0.6667 | 0.7358 | 0.0668 | 0.2267 | 0.1032 |
| CMNS-iGBRT | 0.6720 | 0.7408 | 0.0669 | 0.2276 | 0.1034 |
| CMNS-RF | 0.6747 | 0.7438 | 0.0670 | 0.2273 | 0.1035 |
| Spotlight | 0.4389 | 0.4154 | 0.3653 | 0.0828 | 0.1350 |
| MW | 0.6167 | 0.4154 | 0.4256 | 0.0969 | 0.1579 |
| Tagme | 0.6006 | 0.6233 | 0.2997 | 0.1556 | 0.2048 |

Table 5.10: Performance of best methods.

However, when analysing the complete set of results yielded by each method, we can see that both the proposed methods achieved a $Prec$ more than eight times better than the $CMNS$-based methods and up to 37.12% better than the best baseline, which is $MW$. Due to this impressive performance, even with $Rec$ being 35.26% worse than $CMNS - RF$, $CMNS - GBRT$ and $CMNS - iGBRT$, the $gind(flcg) - RF - f1$ and $gind(xflcg) - RF - f1$ achieved the best overall $F1$ values. The $CMNS$ based methods present high $Rec$ and $MRR$ values and a low $Prec$ value, which indicates that they usually select a large set of keywords which include relevant keywords (yielding a high recall) but also many not relevant keywords, which causes a drop on their precision values.

The $F1$ achieved by $gind(xflcs)$ is 168.41% better than the $CMNS$-based approaches, 105.78% better than *Spotlight*, 75.93% better than $MW$ and 35.64% better than $Tagme$.

## 5.3 Results using ACAKS

We also evaluated the performance of each keyword as input on an advertising system to retrieve products related to short texts. To achieve a good performance we also trained the machine learning algorithm using the *ACAKS* approach described on Chapter 4.

*ACAKS* is an approach which uses the relevance of the products a keyword can retrieve as score to predict which keyword is better than others. The objective of this strategy is to optimize the results in order to obtain a model that will select the keywords that are more capable of being associate to relevant products or ads.

Thus, to use *ACAKS* on the selection of keywords from short texts we adopted the same machine learning framework described on Section . As the dataset for the products, we adopted the web site Amazon[9], since the short texts from the reference collection adopted are written in English. Each keyword was used as a query to retrieve products from Amazon. The top three products were evaluated by a group of 15 volunteers who labelled them as relevant or not. A relevant product is a product that would be considered interesting by the author of the tweet. We then consider as relevant a keyword which retrieved at least one relevant product.

In order to reduce the effort required to evaluate the complete set of products associated to each tweet, we used a subset from *data_en* with 254 tweets. We removed from this subset the tweets where no product returned by Lucene was labelled as relevant, and also the tweets where the users could not identify the subject. This process ended up in a set of 184 tweets. The results presented are the average of these tweets.

We here refer to the *ACAKS* method as $gind(flcg)-RF-ACAKS$ and $gind(xflcg)-$

[9] http://amazon.com

$RF - ACAKS$. The $gind(flcg) - RF - ACAKS$ adopts the ranking provided by the $gind$ method over the $flcg$ as initial set of concepts. The $gind(xflcg) - RF - ACAKS$ adopts the ranking obtained by the $gind$ method over the $xflcg$.

Our main objective is to study how this new approach for learning keywords can improve the results in comparison with the traditional approach, where the user labels the keyword only taking into account its capacity of describing the subject of the text, not considering the ads it could retrieve. For this reason, we compare the results of the $ACAKS$ approach with the results obtained by $gind(flcg) - RF - f1$ and $gind(xflcg) - RF - f1$ which are focused on improving the $F1$ measure. Also, we present the results of the $gind(flcg)$ and $gind(xflcg)$, which are the initial set of concepts before the filtering step performed by the machine learning framework. With this setup, we can see how much each machine learning improved or (worsen) the final result.

Table 5.11 shows the precision of each method using the top ranked keyword as input to retrieve products from Amazon. We can see that using the traditional approach to select the keywords highly decreases the precision obtained by the original set of concepts. On the other hand, the precision achieved by $ACAKS$ approaches is between 4% and 10% better than the original precision obtained using the gind as link analysis method to rank the concepts from $flcg$. With the expanded version of the graph as initial set of concepts, the improvement is even better, between 14% and 21%.

| Method | p@1 | p@2 | p@3 |
|---|---|---|---|
| gind(flcg) | 0.3478 | 0.3641 | 0.3659 |
| gind(flcg)-RF-f1 | 0.2000 | 0.2000 | 0.2065 |
| gind(flcg)-RF-ACAKS | 0.3827 | 0.3796 | 0.3889 |
| gind(xflcg) | 0.3279 | 0.3388 | 0.3315 |
| gind(xflcg)-RF-f1 | 0.2139 | 0.2168 | 0.2100 |
| gind(xflcg)-RF-ACAKS | 0.3966 | 0.3908 | 0.3774 |

Table 5.11: Precision at top 1,2 and 3 ads for each method.

Table 5.12 presents the recall of each method considering top one, two and three products retrieved. Once more, the methods $gind(flcg) - RF - f1$ and $gind(flcg) - RF - f1$ present the worst results. The recall obtained by the $gind(flcg) - RF - f1$ is slightly worse than the one obtained with the $gind(flcg)$. This difference varies between 3% and 7%, and can be attributed to the fact that the classifier is a filter which removes noisy terms from the initial set. Sometimes good terms can be also removed, leading to a drop in recall.

Although, the expanded version of the graphs when trained using $ACAKS$ improves the recall from 10% to 15% in comparison to the $gind(xflcg)$. This can be related to the fact that the expanded version of the graphs have a higher number of vertices, thus being less susceptible to loses in recall.

| Method | r@1 | r@2 | r@3 |
|---|---|---|---|
| gind(flcg) | 0.3478 | 0.3641 | 0.3659 |
| gind(flcg)-RF-f1 | 0.1685 | 0.1685 | 0.1757 |
| gind(flcg)-RF-ACAKS | 0.3370 | 0.3370 | 0.3551 |
| gind(xflcg) | 0.3261 | 0.3397 | 0.3351 |
| gind(xflcg)-RF-f1 | 0.2011 | 0.2065 | 0.1993 |
| gind(xflcg)-RF-ACAKS | 0.3750 | 0.3750 | 0.3678 |

Table 5.12: Recall at top 1,2 and 3 ads for each method.

The results show that using the $ACAKS$ as a strategy to select keywords to be used to retrieve products from an online store is quite better than using the traditional approach of selecting keywords labelled by the user as good descriptors to the content of the tweet. It presents a precision at least 88% better than the traditional approach and a recall more than 80% better.

In comparison to the original set of concepts present on the graph sorted by the global indegree, the proposed approach improves the precision while presenting a competitive recall. These results indicate that $ACAKS$ is a good choice also in the short text scenario, as it is in the context of selecting keywords from web pages.

# Chapter 6

# Conclusions and Future Work

This thesis presents a study about the problem of extracting contextual information from web pages and from short texts available online with the goal of later associating them with ads or products.

Regarding the selection of keywords from web pages, this thesis proposes a new approach for selecting keywords in contextual advertising systems. Our main contribution was a change in the strategy to compose the training collection to guide the learning process. Instead of asking users to directly giving examples of what are the good keywords found in the training pages, we checked which ads have a match with each keyword candidate found in the training pages, and asked the users to evaluate the relevance of the ads that would be associated with these keywords. We found this strategy provide quite competitive results when compared to a previous method proposed recently in literature.

The new approach proposed led to significant gains over the baseline, with gains of 62% in $r@3$ when considering just the features proposed by [40]. Further, our experiments indicate that even when increasing the size of the training in the baseline approach, still the $ACAKS$ presents superior results, which brings the conclusion that the $ACAKS$ approach is a viable and attractive alternative for keyword selection in ad placement systems.

We also studied alternative methods to select keywords without requiring training effort. We proposed three novel approaches for selecting keywords on Web pages: Wiki-

TF-IDF, Wiki-Categories-1 and Wiki-Categories-2.

Experimental results have shown that the three methods are competitive in practice. For instance, when selecting keywords from Wikipedia articles, our best method outperformed the representations based on all the terms (i) and TF-IDF weighting (iii) with gains of about 33% and 26%,respectively. In the worst scenario we found in our experiments, our methods achieved results similar to the approach proposed by the baseline.

We also have shown that Wiki-Categories-1 and Wiki-Categories-2 presented a good performance using small sets of keywords to represent each web page, presenting small computational costs. They achieved, in some scenarios, results with a quality even superior to the other methods experimented. For instance, on Scenario 1 they achieved results slightly superior to the ones obtained when using all terms of a page, which could be a trivial solution to the problem of representing a web page.

Although we experimented the proposed keyword selection methods only with advertising systems, the results presented indicate they may be specially useful in any application where there is a requirement of representing the content of a web page with an small set of keywords. This is the case, for instance, when this small set of keywords is used as a query to an API of web service that limits the maximum number of keywords in a query. For instance, the methods proposed could be useful to automatically select videos related to a page from online video servers, such as Youtube, or to automatically recommend books when a user is browsing a page in a Web site. We will further investigate these and other applications to our method as future work. Finally, since our methods are very cheap to compute, they could be used as complementary features for more sophisticated strategies such as the ones based in machine learning.

Besides the methods for selecting keywords for web pages, the thesis presents and evaluates new methods which take advantage of the connectivity information present on Wikipedia to detect a set of descriptive keywords on a short text. As an apparatus to achieve this task we use context graphs, which are graphs whose nodes represent the concepts related to the text and their edges represent the relation between them. We then experimented some classical link analysis algorithms to rank the nodes of each graph.

The results obtained using each method individually were very appealing. While the methods using the strong local context graph (*slcg*), which is one of context graphs we propose, achieved an overall precision between 0.4667 and 0.7111 depending on the dataset adopted. The $gind(xflcg)$ obtained a recall value of 0.7737 on the *data_br* dataset and 0.5487 on the *data_en*. An interesting observation is that to achieve these results the methods use only the connectivity information from Wikipedia, avoiding high costs for indexing huge amount of textual data.

As the combination of each link analysis algorithm and each context graph produced different values for same keywords, we also experimented using all these values together with the set of features adopted on [24], as features on a Machine Learning Framework. On this framework we evaluated the $gind(flcg)$ and $gind(xflcg)$, the methods which presented the higher recall among our proposals, as input for a Random Forest Algorithm that tries to filter out the irrelevant concepts.

This approach achieved outstanding results even when compared with another first-class approach from the literature. The $F1$ value (which is a metric that summarize the performance of a method on a single value) obtained by the $gind(xflcg) - RF - f1$ approach is more than 35% better than the better baseline. We thus conclude that the proposed methods represent an effective and useful alternative to be adopted in the task of finding keywords on short texts.

As a future work related to the selection of keywords from short texts, we intend to extend our approach to find keywords on other kinds of text, such as web pages and e-mails. We think that it is possible to achieve even better results by taking advantage of the additional information available on such scenarios. Also, we think that we can improve the performance of our methods by studying some more sophisticated link analysis algorithms.

As future work related to the selection of keywords from web pages, we intend to expand our research in order to contemplate additional evidence and other contexts, such as video. As another future work, we intend to study the performance of other machine learning methods aiming to obtain results closer to the ideal one described here. We will particularly investigate the performance of SVM [18] as the classification method adopted to select keywords. Also, we intend to apply the method proposed on [8] to drop the number of keyword candidates to be considered in each page and thus reducing also the number of ads to be evaluate in order to create the training set.

# Bibliography

[1] Aris Anagnostopoulos, Andrei Z. Broder, Evgeniy Gabrilovich, Vanja Josifovski, and Lance Riedel. Just-in-time contextual advertising. In *CIKM 2007: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 331–340, 2007.

[2] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* 1st edition, 1999.

[3] U. Brandes and T. Erlebach. *Network analysis: methodological foundations*, volume 3418. Springer, 2005.

[4] Tim Bray. Readings in information visualization. chapter Measuring the Web, pages 469–492. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[5] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[6] Deepayan Chakrabarti, Deepak Agarwal, and Vanja Josifovski. Contextual advertising by combining relevance with click feedback. In *WWW 2008: Proceeding of the 17th international conference on World Wide Web*, pages 417–426, 2008.

[7] Kino Coursey and Rada Mihalcea. Topic identification using wikipedia graph centrality. In *Proceedings of Human Language Technologies: The 2009 Annual Conference*

*of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short '09, pages 117–120, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[8] Kushal S. Dave and Vasudeva Varma. Pattern based keyword extraction for contextual advertising. In *CIKM 2010: Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1885–1888, 2010.

[9] E. Estrada and N. Hatano. Communicability in complex networks. *Phys. Rev. E*, 77:036111, Mar 2008.

[10] Paolo Ferragina and Ugo Scaiella. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA, 2010. ACM.

[11] R. A. Fisher. Applications of "student's" distribution. *Metron*, 5:90–104, 1925.

[12] L.C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[13] Evgeniy Gabrilovich and Shaul Markovitch. Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Int. Res.*, 34:443–498, March 2009.

[14] Joshua Goodman and Vitor R. Carvalho. Implicit queries for email. In *Second Conference on Email and Anti-Spam*, 2005.

[15] Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 661–670, New York, NY, USA, 2009. ACM.

[16] Taher H. Haveliwala. Topic-sensitive pagerank. In *Proceedings of the 11th international conference on World Wide Web*, WWW '02, pages 517–526, New York, NY, USA, 2002. ACM.

[17] Xiaohua Hu, Xiaodan Zhang, Caimei Lu, E. K. Park, and Xiaohua Zhou. Exploiting wikipedia as external knowledge for document clustering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 389–396, 2009.

[18] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98*, pages 137–142, 1998.

[19] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[20] Mitsumasa Kondo, Akimichi Tanaka, and Tadasu Uchiyama. Search your interests everywhere!: wikipedia-based keyphrase extraction from web browsing history. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, HT '10, pages 295–296, New York, NY, USA, 2010. ACM.

[21] Anísio Lacerda, Marco Cristo, Marcos André Gonçalves, Weiguo Fan, Nivio Ziviani, and Berthier Ribeiro-Neto. Learning to advertise. In *SIGIR 2006: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 549–556, 2006.

[22] Decong Li, Sujian Li, Wenjie Li, Wei Wang, and Weiguang Qu. A semi-supervised key phrase extraction approach: learning from title phrases through a document semantic network. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 296–300, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[23] Olena Medelyan, Eibe Frank, and Ian H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1318–1327, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[24] Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 563–572, New York, NY, USA, 2012. ACM.

[25] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, pages 233–242, 2007.

[26] David Milne and Ian H. Witten. Learning to link with wikipedia. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518, New York, NY, USA, 2008.

[27] T. Mitchell. *Machile Learning.* McGraw-Hill, 1997.

[28] Óscar Muñoz-García, Andrés García-Silva, Óscar Corcho, Manuel de la Higuera Hernández, and Carlos Navarro. Identifying topics in social media posts using dbpedia. In *Proceedings of the NEM Summit 2011*, pages 81–86. Halid Hrasnica,and Florent Genoux, 2011.

[29] M.E.J. Newman. Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality. *Physical review E*, 64(1):016132, 2001.

[30] Berthier Ribeiro-Neto, Marco Cristo, Paulo B. Golgher, and Edleno Silva de Moura. Impedance coupling in content-targeted advertising. In *SIGIR 2005: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 496–503, 2005.

[31] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.

[32] Gerard Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. Technical report, 1974.

[33] Karen Sparck Jones. A statistical interpretation of term specificity and its application to retrieval. *Journal of Documentation*, 28(1):11–20, 1972.

[34] Karen Sparck Jones. Index term weighting. *Information Storage and Retrieval*, 9(11):619–633, 1973.

[35] Karen Sparck Jones. Experiments in relevance weighting of search terms. *Information Processing & Management*, 15(13):133–144, 1979.

[36] Adriano A. Veloso, Humberto M. Almeida, Marcos A. Goncalves, and Wagner Meira Jr. Learning to rank at query-time using association rules. In *SIGIR 2008: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 267–274, 2008.

[37] Li Wan, Jianxin Liao, and Xiaomin Zhu. Cdpm: Finding and evaluating community structure in social networks. In *Proceedings of the 4th international conference on Advanced Data Mining and Applications*, ADMA '08, pages 620–627, Berlin, Heidelberg, 2008. Springer-Verlag.

[38] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. Kea: practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, DL '99, pages 254–255, New York, NY, USA, 1999. ACM.

[39] Xiaoyuan Wu and Alvaro Bolivar. Keyword extraction for contextual advertisement. In *WWW 2008: Proceeding of the 17th international conference on World Wide Web*, pages 1195–1196, 2008.

[40] Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. Finding advertising keywords on web pages. In *WWW 2006: Proceedings of the 15th international conference on World Wide Web*, pages 213–222, New York, NY, USA, 2006.

[41] Weinan Zhang, Dingquan Wang, Gui-Rong Xue, and Hongyuan Zha. Advertising keywords recommendation for short-text web pages using wikipedia. *ACM Trans. Intell. Syst. Technol.*, 3(2):36:1–36:25, February 2012.

[42] Justin Zobel and Alistair Moffat. Inverted files for text search engines. In *ACM Comput. Surv.*, volume 38, page 6, New York, NY, USA, 2006.