A text categorisation tool for open source communities based on semantic analysis

M.R. Martínez-Torres[a], S.L. Toral[b]*, F.J. Barrero[b] and D. Gregor[b]

[a]*Departamento de Administración de Empresas y Comercialización e Investigación de Mercados (Marketing), Universidad de Sevilla, Sevilla, Spain;* [b]*Departamento de Ingeniería Electrónica, Universidad de Sevilla, Sevilla, Spain*

Open source software (OSS) projects are supported by communities interacting through software repositories and mailing lists. Thousands of contributors participate in the development of the projects although they rarely meet each other. The result is a huge archived repository with thousands of questions, answers and contributions usually difficult to explore. We propose a tool based on semantic analysis for both performing an automatic knowledge discovery and a categorisation of the content of mailing lists repositories. Semantic analysis is a practical method for extracting and inferring relations of words in passages of discourse, producing measures of relations among words or passages that are well correlated with semantic similarity. The objective of this article is two-fold: (1) to develop a text categorisation tool based on indexing terms and semantic annotation, and (2) to apply the developed tool to extract the main dimensions related to knowledge sharing activities in virtual communities. Debian Linux ports to embedded processors are used as a case study to accomplish the proposed double objective.

Keywords: semantic analysis; text categorisation; open source; virtual communities

1. Introduction

There is a vast amount of information present on the web for a wide range of domains. As the volume of information within Internet continues increasing, there is a growing need for tools to help people finding, filtering and managing these resources more efficiently. Such a system requires a solution to a number of key problems on the web (Weal *et al.* 2007):

- To find documents that might contain useful content.
- To identify and extract the relevant bits of information from the documents.
- To understand and structure the extracted information.
- To generate the categorisation from the processed information.

Text categorisation is an important component in many information management tasks, such as sorting of emails or files (Menon *et al.* 2004). Usually, text categorisation is a complete unsupervised task with the goal of discovering groups of similar documents in a collection without a-priori knowledge on an applicable class structure. The lack of a predefined ontology of categories makes this task quite difficult, even for humans (Rigutini and Maggini 2005).

This is the case of open source software (OSS) projects and mailing lists repositories, where developers can find huge pieces of information posted by other members of the community supporting the project. In them, expert programmers at different levels, supporters and users voluntarily contribute to a collaborative software project that is managed via the Internet (Hemetsberger and Reinhardt 2006). They collectively develop the software in a decentralised, self-directed, highly interactive and knowledge-intensive process (Kogut and Metiu 2001). Fortunately, open source projects mailing list data are widely available and easy to extract, providing an excellent infrastructure to study the community interactions in an OSS project (Toral *et al.* 2009a). Data is usually archived per month, and its posts can be sorted by thread, subject, author and date. Despite these facilities, it is not so easy to find the information one is looking for, as there are thousands of archived posts through the years. Developers usually search information reading the subjects of posts, but sometimes they are recurrent, and discussed in different months, or they are not exactly summarising the content of the post. Due to these interactions among participants, mechanisms underlying the development of virtual communities are frequently studied from the perspective of social network analysis (Cho *et al.* 2005, Sowe *et al.* 2006, Toral *et al.* 2009b). From this

*Corresponding author. Email: toral@esi.us.es

perspective, community members are modelled as the nodes of the network while arcs represent the flow of knowledge among them (Martínez-Torres *et al*. 2010). Some other studies propose the extension of social network analysis considering not only the activity of the community members but also their patterns of communication (Klamma *et al*. 2006). These patterns allow the identification of different members' profiles in digital social networks, like trolls, spammers, conversationalists, questioners and answering persons. A different approach is the network text analysis developed by Diesner and Carley (2008). They extract different entity classes in texts, so the relations among the elements within and across any entity classes form certain types of networks. Finally, the combination of social network analysis with semantic networks has also been proposed in other works for ontology emergence and orientation (Harrer *et al*. 2007, Mika 2007). In particular, Mika represents networks of folksonomies as a tripartite graph with hyper-edges considering the set of actors (users), the set of concepts (tags, keywords) and the set of objects annotated (bookmarks, photos, etc.). The actor-concept-instance model of ontologies is then applied to several semantic social networks.

In this article, we propose a tool for an automatic categorisation system based on text semantic analysis (CATOSEM). Semantic analysis allows extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text. CATOSEM has been developed in MATLAB^{TM}, and it is based on the latent Dirichlet allocation (LDA) algorithm developed by Blei *et al*. (2003). Using the proposed CATOSEM, a user can browse a mailing list repository more naturally and researchers can advance in the study of open source community behaviour, in particular, measuring knowledge discovery and knowledge sharing inside the community (Hildreth *et al*. 2000, Pan and Leidner 2003, Martínez-Torres 2006).

This article is structured as follows. The next section shows a review of text categorisation tools. After that, the methodology used in this study based on indexing terms and semantic analysis is described. A case study based on Debian Linux ports to embedded processors is proposed in section 4. First, CATOSEM is applied to one of them, in order to show the way in which the proposed tool is working and how the main topics of discussion are extracted, and then it is applied to several Debian Linux ports. Factor analysis is used to extract the main dimensions related to knowledge sharing activities. The obtained results are discussed in section 5 and finally, the article is concluded in section 6.

2. Review of text categorisation tools

Text categorisation tools commonly makes use of vector space model (Salto and McGill 1983), in which documents are summarised and represented by vectors of words (term vectors). However, a central problem in statistical text categorisation is the high dimensionality of the feature space (one dimension for each unique word). Therefore, it is desirable to first project the documents into a lower-dimensional subspace in which the semantic structure of the document space becomes clear (Cai *et al*. 2005). In this low-dimensional semantic space, the traditional clustering algorithms can be then applied. To this end, spectral clustering (Shi and Malik 2000, Ng *et al*. 2001), clustering using latent semantic indexing (LSI) (Zha *et al*. 2001), and clustering based on non-negative matrix factorisation (Xu *et al*. 2003, Xu and Gong 2004) are the most well-known techniques. Particularly, LSI decomposes a term document matrix using a technique called singular value decomposition to construct new features as combinations of the original features, significantly reducing the high-dimensionality problem of the feature space (Deerwester *et al*. 1990, Abedin and Sohrabi 2009). Moreover, LSI also considers documents that have many words in common to be semantically close, while those with few words in common are considered to be semantically distant. The LSI approach makes three basic claims: (1) semantic information can be derived from a word-document co-occurrence matrix; (2) dimensionality reduction is an essential part of this derivation; and (3) words and documents can be represented as points in Euclidean space. Different to LSI, generative models consider topics as a new variable related to words and documents. It is based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words (Hofmann 2001, Blei *et al*. 2003, Griffiths and Steyvers 2004). These models are consistent with the first two of these claims, but it differs in the third: documents are represented in a reduced space in which the semantic properties of words and documents are expressed in terms of probabilistic topics. In this article, the LDA described by Griffiths and Steyvers (2004) has been applied (see Appendix for more details).

The generative process underlying topic model is illustrated in Figure 1 with two topics (Steyvers and Griffiths 2007). Topics 1 and 2 are shown as bags containing different distributions over words. Different documents can be produced by picking words from a topic depending on the weight given to the topic. For example, documents 1 and 3 were generated by sampling only from topic 1 and 2, respectively while document 2 was generated by an equal mixture of the two topics. Note that the superscript numbers associated with the words in documents indicate which topic was used to

sample the word. Consequently, there is no notion of mutual exclusivity that restricts words to be part of one topic only. This allows topic model to capture polysemy, where the same word has multiple meanings. A variant of LDA called latent Dirichlet allocation category language model (LDACLM) was proposed by Zhou *et al.* (2008). The most interesting feature of LDACLM is that the model assumes each word would be an independent topic and assume extra topics other than word topics would be model the correlation among the words. Some other variants are focused on smoothing LDA for improving text categorisation under the generative framework (Li *et al.* 2008). In the field of information retrieval, LDA model has been used as a tool for text categorisation. By labelling each word with a topic, LDA allows representation of a document in the form of its semantic topic content rather than the words of vocabulary, thus achieving a significant reduction in the dimensionality of text representation, usually from tens of thousands to hundreds (Lu *et al.* 2006).

3. Objective and methodology

The objective of this article is two-fold:

(1) To develop a text categorisation tool based on indexing terms and semantic annotation.
(2) To apply the developed tool to extract the main dimensions related to knowledge sharing activities in virtual communities.

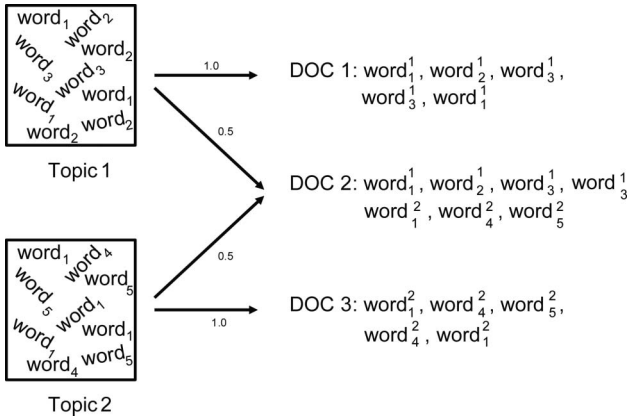

Figure 1. Probabilistic generative process underlying topic models.

To achieve these objectives, a methodology based on the LDA algorithm has been used to develop a text categorisation tool, called CATOSEM. This tool has been applied to several virtual communities to extract their main features from the point of view of semantic analysis, and the obtained results were statistically treated through factor analysis to extract the main latent dimensions of the considered virtual communities.

Figure 2 shows a dataflow of the proposed categorisation tool, which consists of four parts:

(1) Information extraction: mailing lists information is usually organised per month and year, and posts may be sorted by threads, subjects, authors and date inside each month. The extraction process involves the use of specific software to automatically extract the required information. CATOSEM has been programmed using MATLAB^{TM}. Data is accessed per month downloading and processing HTML web pages. Each post is taken as a document, but just the subject and body of the message is considered for the subsequent analysis. As we need a set of indexing terms, it is necessary to pre-process each extracted document. Therefore, punctuation marks, HTML tags and prepositions are removed. Afterward, all the words are lower case converted. Once preprocessing is completed, CATOSEM obtain the frequency-of-occurrence-rates for word tokens.
(2) Ontology: an ontological model of the domain is used as a facilitator throughout all the processes. This provides a common vocabulary and specifies the semantics of key relationships within the domain (Maedche and Staab 2001). An ontology is an explicit formal specification of terms and relations among them, representing the intended meaning of concepts in a specific domain. The proposed generative topic model needs a selection of indexing terms comprising the vocabulary of a topic area. As sometimes it is difficult to find an accepted ontology in a given area, this selection can be done using obtained frequency-of-occurrence-rates for word tokens.

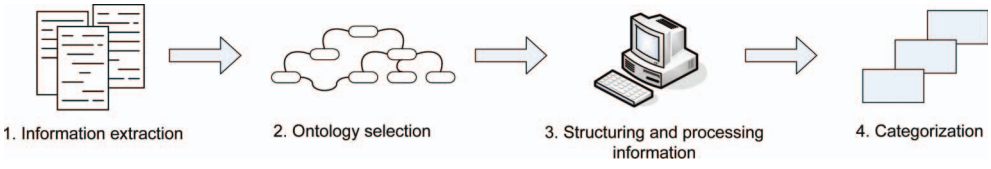

Figure 2. Dataflow of the proposed categorisation tool.

(3) Structuring and processing information: LDA algorithm is executed in MATLAB. Input information is provided as a bag of words by two vectors W_S and D_S. W_S (k) and D_S (k) contain the word and document indices for the kth token. The maximum of W_S is W, the vocabulary size. The maximum of D_S is D, the number of documents. T is the number of topics, and it is also an input parameter of the algorithm.
(4) Categorisation: the algorithm outlined above can be used to find the topics that account for the words used in a set of documents. In our study, each post is considered a document, so the algorithm would extract the main topics of the archived post during a period of time.

4. Case study

The proposed case study is based on Linux, which is one of the most famous and cited OSS projects. There are several reasons for this choice. First, Linux represents the best-known case of OSS. The 'community-based model' of knowledge creation of the Linux community as opposed to the 'firm-based model' has been deeply analysed in the literature (Hertel *et al.* 2003, Lee and Cole 2003). Nevertheless, we have avoided the basic Linux kernel choice because its outstanding position could produce atypical results. Instead, we have focused on Linux port to embedded processors, which are more oriented to researchers and professionals.

Among the Linux distributions, Debian is one of the most well-known. The Debian project, which was founded in 1993 by Ian Murdock, is a worldwide group of volunteers who endeavour to produce an operating system distribution that is composed entirely of free software (Michlmayr and Senyard 2006). Debian GNU/Linux software distribution, which includes the Linux operating system kernel and thousands of pre-packaged applications, is developed through distributed development all around the world. Much of the conversation between Debian developers and users is managed through several mailing lists, which can be easily accessed through the Debian web site (http://lists.debian.org/).

The proposed tool, CATOSEM, will be first applied to the Debian port to Advanced RISC Machines (ARM) mailing list and then it will be extended to the rest of Debian Linux ports. The aim of this first analysis consists of illustrating the way in which topics are extracted in a particular case. Then, the same methodology will be next followed to obtain the main dimensions related to the mailing lists activities from the semantic analysis perspective.

4.1. *Debian Linux port to ARM architecture*

The ARM processor architecture, which stands for advanced reduced instruction set computing (RISC) machine, is a family of processors maintained and promoted by ARM Holdings Ltd. Contrary to other chip manufacturers such as IBM, Motorola and Intel, ARM Holdings does not manufacture its own processors (Barrero *et al.* 2008, Toral *et al.* 2009c). Instead, ARM designs the CPU cores for its customers based on the ARM core, charges customers licensing fees on the design, and lets them manufacture the chip wherever they see it fits. Currently, ARM CPUs are manufactured by Intel, Toshiba, Samsung, Freescale, Texas Instruments and many others. The ARM architecture is very popular in many fields of application and there are hundreds of vendors providing products and services around it. Today, Linux supports more than 1200 related ARM-based boards.

The Debian port to ARM mailing list (http://lists.debian.org/debian-arm/) has been analysed during its lifetime from 1999 to 2007. A manual categorisation of topics would have required the participation of several experts during a very long period of time. It must be taken into account that there are thousands of archived posts and, sometimes, discussions reach a high level of complexity. The proposed tool based on the topic model will avoid the participation of experts and will reduce the amount of time required to perform the categorisation. A key feature of this model is that it is an unsupervised learning technique, which means that the often human-intensive task of finding labelled examples is completely eliminated. Unsupervised also means that one can model a collection of documents through topics without being a domain expert – in fact one can even model a collection of documents through topics in other languages, without needing to know much about the language (Newman *et al.* 2006). Up to 7482 different messages were considered in this study. They have been downloaded and processed using specific software developed using MATLAB® (Register 2007). MATLAB® is a high-level language and interactive environment that enables you to perform easily computationally intensive tasks, thanks to the provided toolbox (Moler 2004). MATLAB® provides high-level language for technical computing, a development environment for managing code, files and data, interactive tools for iterative exploration, design and problem solving, and mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimisation and numerical integration. The sequence of steps followed to process the information is next detailed. All of them are repeated for each of the considered years.

- The developed program downloads all the messages corresponding to each year, and then

extracts the header and the body of the message. In particular, HTML tags are processed for this purpose.
- Punctuation marks, HTML tags and prepositions are removed, and then all the words are lower case converted. Finally, the frequency-of-occurrence-rates for word tokens are obtained.
- The most repeated words are used as the basic vocabulary to describe the domain of study.
- Topics are extracted using LDA algorithm.

The LDA algorithm involves the selection of several parameters to achieve a topic description of the domain under study. Table 1 details the key dimensions or size parameters that describe a corpus (D, W, N and L) and a topic model run (T and Gibbs sampler iterations (ITER)). In our case study, each year is considered a different corpus. The aims are to extract topics for each year and to compare the topic evolution over time. Table 2 details the particular parameters of the Debian ARM mailing list during the considered period of time.

Although the number of documents (D column) exhibits high variations among years, the average length of documents in words (L column) show slighter variations. The most active years are the last two ones, 2006 and 2007, which is in accordance with the growing interest in Linux ports to embedded processors. After pre-processing the content of 12,022 posts, we have extracted an initial vocabulary of more than 120,000 words. The final length of the vocabulary ($W = 410$) has been selected considering the occurrence frequency rate (Ng *et al*. 2001).

Two parameters must be selected before running the LDA algorithm (Blei *et al*. 2003). The number of ITER has been chosen equal to 200. This value is large enough to guarantee the convergence of the algorithm (Ng *et al*. 2001). The number of topics has been selected using perplexity. Perplexity is a standard measure of performance for statistical models of natural language (Manning and Schutze 1999), and it is defined by Equation (1).

$$pplex = \exp\left(-\frac{1}{W}\sum_{n=1}^{W}\log P(w_n|d_n)\right) \qquad (1)$$

Perplexity varies from 1 to W; lower perplexity is better, and the maximum perplexity of W is reached when all words in the vocabulary are equally likely. In our case study, LDA algorithm has been run for a number of topics varying between 1 and 50. Results for one of the considered years are illustrated in Figure 3. Perplexity varies with the number of topics, and the topic parameter is selected following the minimum perplexity criterion.

The results of running the topic model during the years 1999–2007 is detailed in Table 3. In particular, the resulting topics for each year are detailed including the five more relevant words for each topic. The following considerations can be highlighted from Table 3:

- The number of topics (obtained as the number of topics minimising the perplexity of Equation (1)) varies from one year to another. In the analysed period of time, the minimum value is 9 and corresponds to years 2000 and 2004, while the

Table 1. Dimensions used in topic model.

Parameter	Description
D	Number of documents in corpus (messages)
W	Number of words in vocabulary
N	Total number of words in corpus
L	Average length of document in words ($L = N/D$)
T	Number of topics
ITER	Number of iterations

Table 2. Parameters of the Debian Linux ARM mailing list.

Year	D	W	N	L
1999	552	410	100,339	181.77
2000	667	410	115,869	173.72
2001	956	410	193,313	202.21
2002	848	410	218,035	257.12
2003	377	410	82,904	219.90
2004	684	410	113,440	165.85
2005	625	410	108,523	173.64
2006	1098	410	218,565	199.06
2007	1675	410	356,481	212.82

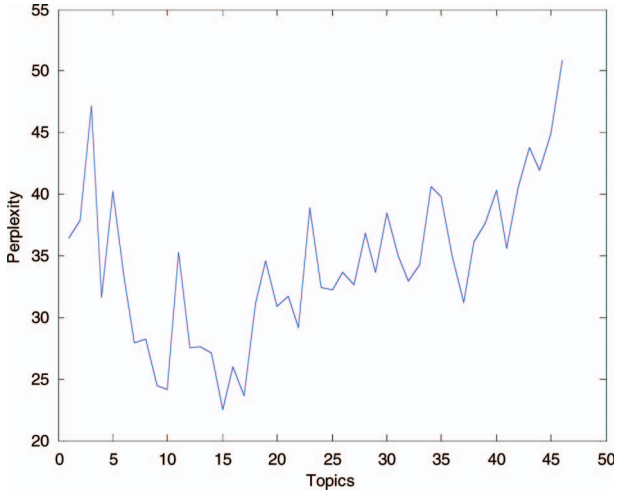

Figure 3. Perplexity value as a function of the number of topics.

Table 3. Topic distribution per year.

	Topic1	Topic2	Topic3	Topic4	Topic5	Topic6	Topic7	Topic8	Topic9	Topic10	Topic11	Topic12	Topic13	Topic14	Topic15	Topic16	Topic17	Topic18	Topic19
1999	kernel	debian	Problem	struct	ram	build	time	king	machine	arm	files								
	linux	image	Gcc	instruction	system	start	source	config	compile	binary	server								
	gnu	ftp	Built	support	program	running	patch	version	main	write	standard								
	cross	install	Error	status	install	script	pass	glibc	fix	Documentation	basic								
	compiler	stuff	Direct	memory	archive	trace	share	even	upload	value	debian								
2000	kernel	install	Arm	image	config	build	compile	fix	debian										
	problem	time	Machine	ftp	ram	gcc	patch	main	arm										
	hosting	group	Linux	linux	server	error	version	source	binary										
	start	support	Compiled	system	built	software	king	gnu	embedded										
	share	arm	Memory	files	off	direct	glibc	power	gdb										
2001	kernel	build	debian	time	install	linux	compile	ram	arm	error	problem								
	king	arm	ftp	start	version	include	arm	source	files	report	machine								
	config	glibc	main	king	process	fix	patch	program	system	direct	gcc								
	support	cpu	open	write	status	stable	built	stuff	float	ports	gnu								
	image	default	basic	server	rmk	power	compiler	block	flash	struct	even								
2002	block	kernel	king	arm	ram	debian	build	system	machine	report	off	problem	linux	debian					
	server	config	patch	fix	include	install	built	files	family	gnu	time	start	arm	main					
	flash	source	even	debian	program	running	stable	image	support	software	write	error	gcc	ftp					
	stuff	script	available	reference	device	project	target	user	king	ram	number	open	compile	source					
	install	version	struct	traffic	module	process	upload	random	pass	system	memory	debug	cross	stable					
2003	stuff	gcc	debian	address	arm	support	script	arm	block	build	error	time	compile	problem	machine	image	arm	linux	kernel
	king	linux	commercial	result	ram	fpu	even	module	receive	binutils	fix	user	system	install	cross	config	debian	build	available
	return	include	tudelft	open	main	cpu	report	built	start	running	memory	files	king	version	source	process	Development	embedded	install
	software	error	flash	number	build	debug	upload	patch	include	pass	function	console	compiler	toolchain	power	stable	project	main	ftp
	direct	arm	mailing	family	registered	struct	problem	driver	client	executable	main	failure	install	off	issue	gnu	device	abi	off
2004	build	include	debian	install	problem	linux	usb	arm	off										
	arm	version	main	ram	built	arm	system	compile	support										
	upload	config	stable	machine	time	gcc	board	struct	king										
	available	error	kernel	patch	king	gnu	time	issue	even										
	machine	report	ftp	driver	fix	program	ram	instruction	family										
2005	gcc	install	system	debian	fix	machine	arm	time	kernel	arm	build	main	arm						
	compile	debian	ram	arm	struct	off	block	king	family	support	built	source	linux						
	patch	problem	board	report	software	access	stable	start	usb	user	upload	version	gnu						
	error	running	embedded	network	abi	available	issue	pass	image	big-endian	archive	process	config						
	cross	platform	files	cpu	result	hardware	flash	isp	script	float	accept	open	direct						
2006	arm	kernel	debian	family	system	arm	build	ram	block	king	config	kernel							
	machine	partition	install	user	process	linux	problem	flash	abi	time	device	image							
	main	loader	toolchain	address	compile	source	upload	direct	arm	start	usb	support							
	ports	off	switch	archive	even	gnu	issue	driver	struct	script	support	version							
	target	ramdisk	Developers	server	cross	gcc	fix	memory	float	available	sound	install							
2007	arm	error	gcc	patch	install	system	kernel	build	block	time	debian	config							
	abi	king	gnu	king	main	mount	image	source	struct	off	issue	usb							
	ports	server	arm	glibc	ftp	running	linux	compile	partition	problem	report	device							
	applied	process	linux	debian	debian	write	flash	machine	instruction	open	compiled	family							
	available	share	abi	support	problem	user	ram	cpu	download	version	archive	register							

maximum value is 19, corresponding to year 2003. The number of topics is a measure of the variety of discussions and the knowledge revealed. Notice that no topic is exactly repeated during the analysed period of time. That means that each topic means a new field in which practice is revealed.

- Although topics are not repetitive, they show words in common with some other topics. This is one of the advantages of the proposed methodology: its capability of dealing with words that can be used in different contexts. For instance, the word kernel appears 10 times, but in different contexts related to compilation, configuration or availability issues.

4.2. *Debian Linux port to embedded processors*

Twelve Debian Linux port communities have been considered for this study. They are detailed in Table 4. Each community was analysed during the period of time detailed in the fourth column of Table 4. Basically, the initial year is the one in which the community had a certain activity every month. For each year and community, summarised indicators from Table 5 were obtained. Consequently, 110 case studies were considered.

The first indicator I1 is the number of topics. This is a parameter of the topic model that has been chosen attending to the perplexity criterion explained in subsection 4.1.

Indicators I2 and I3 are related to polysemy. Generative models have the ability of capturing polysemy, where the same word has multiple meanings. This is because they do not impose restrictions about mutual exclusivity that restricts words to be part of one topic. I2 measures the polysemy as the number of times a word w_i appears more than once in different topics, while I3 considers this previous value rated with the probability $P(w_i|z_i = j)$ of the word w_i under the jth topic.

The next four indicators refer to the messages. They consider the average size of messages in characters and words, the average number of messages per topic and their distribution over topics.

Finally, the last three indicators refer to threads of discussion. Online communities are usually organised by threads of discussion. Threads are groups of messages sharing the same subject. A thread is initiated by someone who posts a message asking for help, suggesting some improvements or just considering some new idea. Then people start answering this initial message, posting possible solutions, sources of information or just extending posted considerations. The indicator I8 relative to the number of threads considers

Table 4. Virtual communities considered.

	URL	Description	Period
Debian port to m68k (D-68k)	http://lists.debian.org/debian–68k/	Motorola 68k port of Debian GNU/Linux. Debian currently runs on the 68020, 68030, 68040 and 68060 processors.	1998–2008
Debian port to Alpha (D-Alpha)	http://lists.debian.org/debian-alpha/	The purpose of this project is to assist developers and others interested with the ongoing project to port the Debian distribution of Linux to the Alpha family of processors.	1998–2008
Debian port to AMD64 (D-AMD64)	http://lists.debian.org/debian-amd64/	The port consists of a kernel for all AMD 64bit CPUs with AMD64 extension and all Intel CPUs with EM64T extension, and a common 64bit userspace.	2004–2008
Debian port to ARM (D-ARM)	http://lists.debian.org/debian-arm/	ARM port for Debian GNU/Linux. Debian fully supports a port to little-endian ARM.	1999–2008
Debian port to BSD (D-BSD)	http://lists.debian.org/debian-bsd/	This is a port of the Debian operating system, complete with apt, dpkg, and GNU userland, to the NetBSD kernel.	2001–2008
Debian port to HPPA (D-HPPA)	http://lists.debian.org/debian-hppa/	This is a port to Hewlett-Packard's PA-RISC architecture.	2001–2008
Debian port to Hurd (D-HURD)	http://lists.debian.org/debian-hurd/	The GNU Hurd is a totally new operating system being put together by the GNU group.	1999–2008
Debian port to IA64 (D-IA64)	http://lists.debian.org/debian-ia64/	Debian port to Intel IA-64.	2001–2008
Debian port to MIPS (D-MIPS)	http://lists.debian.org/debian-mips/	MIPS port of Debian GNU/Linux, able to run at both endiannesses.	1999–2008
Debian port to PowerPC (D-PPC)	http://lists.debian.org/debian-powerpc/	PowerPC port of Debian GNU/Linux. The PowerPC architecture allows both 64-bit and 32-bit implementations.	1999–2008
Debian port to S390 (D-S390)	http://lists.debian.org/debian-s390/	Debian port to IBM S/390	2001–2008
Debian port to SPARC (D-SPARC)	http://lists.debian.org/debian-sparc/	This port runs on the Sun SPARCstation series of workstations, as well as some of their successors in the sun4 architectures.	1998–2008

those threads with at least one answer. Indicators I9 and I10 are the same as I5 and I6, but using threads instead of topics as the unit of analysis.

A factor analysis was applied to extract the main dimensions related to knowledge sharing in virtual communities. Factor analysis attempts to identify underlying variables or factors, which can explain the pattern of correlations within a set of observed variables. Factor analysis is a way to fit a model to multivariate data, estimating their interdependence. It addresses the problem of analysing the structure of interrelationships among a number of variables by defining a set of common underlying dimensions, the factors, which are not directly observable, segmenting a sample into relatively homogeneous segments (Rencher 2002). These groups represent the underlying variables or factors, which can explain the pattern of correlations within a set of observed variables (Stevens 1992). Each observable variable is assumed to be dependent on a linear combination of the common factors, and the coefficients are known as loadings (Rencher 2002). There are several extraction methods: principal components and principal axis factoring (or principal factor analysis) are among the most widely used. According to Hair *et al*. (1995), the former is used when the objective is to summarise most of the original information in a minimum number of factors, whereas the latter is used to identify the underlying dimensions reflecting what the variables share in common. In most applications, both methods arrive at essentially identical results. In this case, factor analysis has been performed using the principal component method.

Factor analysis can be used for either exploratory or confirmatory purposes: exploratory analyses do not set any a priori constraints on the estimation of factors or the number of factors to be extracted, while confirmatory analysis does. In our case, we have developed an exploratory analysis as we did not know the number of underlying dimensions. That means a decision must be made about the number of factors to be extracted. There are several criteria for doing this, being the most extensive the eigenvalue and percentage of variance criterion. The percentage of variance criterion considers all factors accounting for about 70% of the variance of the original variables (Hair *et al*. 1995).

Once the number of factors has been determined, the next step is to interpret them according to the factor loadings matrix. The estimated loadings from an unrotated factor analysis fit can usually have a complicated structure. The goal of orthogonal factor rotation is to find a parameterisation in which each variable has only a small number of large loadings, i.e. is affected by a small number of factors. The rotated factor analysis fit ensures that factors represent unidimensional constructs while preserving the essential properties of the original loadings. The most popular of these techniques is the varimax rotation, which seeks rotated loadings that maximise the variance of the squared loadings in each column of the factor loading matrix (Rencher 2002).

The eigenvalues of the sample covariance matrix are shown in Table 6. In our case study, the criterion of accounting for more than 70% of the total sample variance is achieved with three factors.

Using the associated eigenvectors, factor loadings can be estimated. Varimax rotation simplifies the interpretation of the factors. Table 7 reports the rotated factor loadings with varimax rotation for each one of the indicators analysed.

To extract the meaning of each factor, we move horizontally through Table 7, from left to right, across the three estimated loadings of each variable, identifying the highest loading and the corresponding factor. To assess significance of factor loadings, a threshold value of 0.7 was considered (Rencher 2002, Martínez-Torres and Toral 2010). The association between variables and factors is highlighted in grey in Table 7.

On the other hand, factor scores are used to categorise the original sample, which can be

Table 5. Indicators related to knowledge sharing.

Indicator	Description
I1	Number of topics
I2	Polysemy
I3	Rated Polysemy
I4	Average messages size (characters)
I5	Average number of messages per topic
I6	Messages per topic distribution
I7	Average messages size (words)
I8	Number of threads (al least one answer)
I9	Average number of threads per topic
I10	Threads per topic distribution

Table 6. Total variance explained.

	Initial eigenvalues		
Component	Total	% of variance	Cumulative %
1	4.852	48.524	48.524
2	2.821	28.214	76.738
3	1.861	18.610	95.347
4	0.157	1.569	96.916
5	0.099	0.985	97.901
6	0.081	0.812	98.713
7	0.051	0.513	99.227
8	0.036	0.364	99.591
9	0.024	0.236	99.828
10	0.017	0.172	100.000

Table 7. Rotated Component matrix with Varimax rotation.

	Component		
	1	2	3
I1	−0.093	0.959	−0.070
I2	0.209	0.951	0.066
I3	−0.069	0.982	0.033
I4	−0.117	−0.016	0.973
I5	0.984	−0.052	−0.052
I6	0.945	0.066	−0.011
I7	−0.065	0.036	0.980
I8	0.976	0.120	−0.078
I9	0.977	−0.081	−0.069
I10	0.962	−0.008	−0.124

Table 8. Statistical significance of ANOVA.

Indicator	*F*	Sig
I1	30.29	0.000
I2	31.99	0.000
I3	41.38	0.000
I4	31.29	0.000
I5	60.94	0.000
I6	38.48	0.000
I7	32.78	0.000
I8	83.83	0.000
I9	75.29	0.000
I10	86.63	0.000

Table 9. Mean values per factor of selected indicators.

Indicator	*F*1	*F*2	*F*3
I1	28.80	30.85	22.96
I2	57.96	71.94	47.34
I3	8.46	12.38	7.99
I4	1565.93	1662.29	2250.87
I5	153.77	38.06	33.60
I6	120.44	36.43	27.42
I7	276.24	288.66	381.22
I8	682.12	207.14	133.35
I9	46.67	14.30	13.54
I10	195.95	57.92	46.51

approximated to one of the identified latent factors. Consequently, the original sample of communities can be categorised in three groups. An analysis of variance (ANOVA) has been applied to the categorisation of the original sample in the three groups obtained from factor analysis. The aim of this analysis consists of checking the null hypothesis of equal population means. Table 8 details the *F* statistic, the ratio of two different estimators of population variance, which appears together with its corresponding critical level or observed significance. The result is that the null hypotheses have been rejected in all the cases with a significance value below 0.05. That means the obtained categorisation from factor analysis is well defined. Table 9 details the mean value of the considered indicators per each of the distinguished groups. Using this information as well as the resulting aggregation of variables of factor analysis, the following latent factors or dimensions can be distinguished:

- The first factor refers to topic activity. The activity around topics is highlighted by the high value of I5, I8 and I9 indicators which account for the number of messages and threads associated to topics. However, the high values of the standard deviation in the messages and thread distributions per topic suggest that all the topics are not treated the same way. This factor shows the fact that communities tend to specialise on certain topics attracting the interest of users.
- The second factor is related to knowledge creation and reuse. The number of topics and polysemy is a measure of knowledge creation and reuse. This is precisely one of the main abilities of communities of practice (CoP). Topics are continuously evolving and previous knowledge is mixed and combined to generate new knowledge.
- The third factor refers to the amount of provided information. I4 and I7 indicators refer to the average size of messages. The availability and depth treatments of topics are also a determinant factor for a successful development of the underlying community.

The obtained results demonstrate the necessity of guiding the evolution of the virtual community. Although virtual communities are based on the volunteer collaboration of community members, people posting a message hope to find an answer to their question or an alternative solution. Consequently, virtual communities are not only a question of social participation, but also of the quality of the provided information. Obviously, it is very difficult to assess individually the quality of each answer, as there are thousands of them, or to evaluate when a new knowledge is created or reused. For this reason, it is necessary to set a group of indirect indicators able to measure activity around the extracted topics, or the knowledge created through the evolution of the community.

5. Discussion

A virtual community can be defined as a social relationship aggregation, facilitated by Internet-based technology, in which users communicate and build personal relationships (Rheingold 1993, Hew 2009). Individuals get engaged in knowledge sharing, problem

solving and learning through posting and responding to questions on professional advice, storytelling of personal experiences and debate on issues relevant to the network. Virtual communities have been frequently connected with CoP (Wellman and Gulia 1995, Lin and Lee 2006), in the sense that communities develop their own routines, formal and informal 'rules', and practices evolve as a result of learning. The concept of CoP was developed by Lave and Wenger (1991). This concept refers to the process of social learning that occurs when people who have a common interest in some subject or problem collaborate over an extended period to share ideas, find solutions and build innovations. CoPs are not formal structures, such as departments or project teams. Instead, they are informal entities, which exist in the mind of their members, and are glued together by the connections the members have with each other, and by their specific shared problems or areas of interest (Wenger and Snyder 2000, Ardichvili *et al*. 2003). Several researchers have noted that CoPs appear to be a more effective tool for dealing with unstructured problems and knowledge sharing/creation than traditional and formal ways of structuring interaction in organisations (Kankanhalli *et al*. 2003). Understanding the processes and mechanisms that enable members to share knowledge in CoPs is very important for knowledge sharing within and between such communities (Pan and Leidner 2003, Toral *et al*. 2010).

One of these mechanisms is participation, which is the process by which a newcomer is integrated into the community. In this process, new members learn how to function as a community member through participation, and acquire the language, values and norms of the community. In the case of OSS projects, they provide web sites with forums and mailing lists where participants and contributors can report software improvements, needs or bugs and share and discuss solutions to posted messages. Participation can be analysed using social network analysis techniques (Toral *et al*. 2009d) and it can be associated to the softer aspects of knowledge according to the duality defined by Hildreth and Kimble (2002).

The second process involved in the development of virtual communities is reification, which means giving concrete form to something that is abstract. It is the process underlying the construction of the so called hard knowledge (Hildreth and Kimble 2002). In the case of OSS projects, it refers to the knowledge stored and publicly available through forums, discussion and repositories.

The developed tool CATOSEM is attending to the reification process involved in the construction of virtual communities. According to the theory, three specific types of knowledge sharing categories can be distinguished. The first one consists of revealing personal uniquely acquired experience and knowledge, which is shared with the rest of the community. Table 3 shows some examples of revealed knowledge. Notice that Linux ports to embedded processors are usually focused on installation and configuration issues, as they are not as homogeneous as desktop PCs. That is the reason why a lot of topics are related to kernel image, cross compilation and installation issues. An example of revealed knowledge is USB, which appears for the first time during the year 2004 (topic 7). Although USB is a well known input/output interface for desktop PCs, they have not been incorporated to embedded devices till later dates. The same can be said for flash memories, which appears for the first time in 2001 in a topic related to the incorporation of flash memories to Linux file system. Once the knowledge is revealed, it can be reused in other fields or re-applied in different practices. Knowledge reusing avoids duplication of effort in re-inventing solutions (Wai 2008). In the case of USB example, it is used later in different contexts related to drivers' configuration, initialisation scripts or support for sound applications. Flash memories are also discussed in contexts related to installation problems, drivers' design or block operation. Finally, knowledge can be recombined to generate new knowledge (Kuk 2006). Following with the previous example, USB and the kernel image knowledge are recombined to deal with the issue of incorporating USB drivers as part of the kernel image (topic 9, 2005). A similar consideration can be mentioned about flash memories and kernel image. Flash memories are usually employed in embedded systems as non-volatile memory, so topic 7 in 2007 deals with the issue of writing the kernel into flash.

These three categories are embedded in the reification process, as it can be concluded from the dimensions obtained in the proposed case study. The first and the second factor are highly related to the knowledge sharing categories described in the literature. Some other studies conclude positive relationship between community knowledge sharing activity and community performance (Koh and Kim 2004), but the main difference is that they use posting and viewing activity indicators as the proper measures for community knowledge sharing activity, while we claim the use of a text categorisation tool to obtain a more accurate measure of knowledge sharing activities.

The present study has certain limitations. First, we have considered a specific set of communities related to Linux. Obviously, there are thousands of virtual communities in the web, and this study could be extended to these other types of virtual communities. Second, we have considered the reification process separately from the participation process. As a future

extension of this work, we propose to analyse jointly both processes involved in the development of virtual communities, that is, participation and reification, in order to extract some conclusion about how these two processes are interrelated.

6. Conclusions

The purpose of this article has been the development of a text categorisation tool for the analysis of knowledge sharing activities in OSS projects. The base of the categorisation tool is the topic model, which is a generative model based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. LDA algorithm has been used to extract the topics, and the number of topics has been given by the minimum value of perplexity. As a case study, several communities related to Debian–Linux ports to embedded processors have been analysed, measuring a set of predefined indicators. The application of a statistical technique like factor analysis allows the extraction of the main dimensions related to community knowledge sharing processes.

Acknowledgements

This work has been supported by the Spanish Ministry of Education and Science (Research Project with reference DPI2007-60128) and the Consejería de Innovación, Ciencia y Empresa (Research Project with reference P07-TIC-02621).

References

Abedin, B. and Sohrabi, B., 2009. Graph theory application and web page ranking for website link structure improvement. *Behaviour & Information Technology*, 28 (1), 63–72.

Ardichvili, A., Page, V., and Wentling, T., 2003. Motivation and barriers to participation in virtual knowledge-sharing communities of practice. *Journal of Knowledge Management*, 7 (1), 64–77.

Barrero, F., Toral, S.L., and Gallardo, S., 2008. EDSPLAB: remote laboratory for experiments on DSP applications. *Internet Research*, 18 (1), 79–92.

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

Cai, D., He, X., and Han, J., 2005. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17 (12), 1624–1637.

Cho, H., *et al*., 2005. Development of computer-supported collaborative social networks in a distributed learning community. *Behaviour & Information Technology*, 24 (6), 435–447.

Deerwester, S., *et al*., 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science and Technology*, 41 (6), 391–407.

Diesner, J. and Carley, K.M., 2008. Conditional random fields for entity extraction and ontological text coding. *Journal of Computational and Mathematical Organization Theory*, 14, 248–262.

Griffiths, T.L. and Steyvers, M., 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101, 5228–5235.

Hair, J.F. Jr., *et al*., 1995. *Multivariate data analysis with readings*. London: Prentice Hall International.

Harrer, A., *et al*., 2007. Combining social network analysis with semantic relations to support the evolution of a scientific community. *In*: C. Chinn, G. Erkens, and S. Puntambekar, eds., *Mice, minds, and society – The Computer Supported Collaborative Learning (CSCL) Conference 2007*, 16–21 July, New Brunswick, NJ. Hong Kong: International Society of the Learning Sciences, 267–276.

Hemetsberger, A. and Reinhardt, C., 2006. Learning and knowledge-building in open-source communities: a social-experiential approach. *Management Learning*, 37 (2), 187–214.

Hertel, G., Niedner, S., and Herrmann, S., 2003. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. *Research Policy*, 32, 1159–1177.

Hew, K.F., 2009. Determinants of success for online communities: an analysis of three communities in terms of members' perceived professional development. *Behavior and Information Technology*, 28 (5), 433–445.

Hildreth, P., Kimble, C., and Wright, P., 2000. Communities of practice in the distributed international environment. *Journal of Knowledge Management*, 4 (1), 27–38.

Hildreth, P.M. and Kimble, C., 2002. The duality of knowledge. *Information Research*, 8 (1), 1–17.

Hofmann, T., 2001. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning Journal*, 42 (1), 177–196.

Kankanhalli, A., *et al*., 2003. The role of IT in successful knowledge management initiatives. *Communications of the ACM*, 46 (9), 69–73.

Klamma, R., *et al*., 2006. Pattern-based cross media social network analysis for technology enhanced learning in Europe. *In*: W. Nejdl and K. Tochtermann, eds., *Innovative approaches to learning and knowledge sharing, Proceedings of the 1st European Conference on Technology Enhanced Learning (EC-TEL 2006)*, 1–3 October, Hersonissou, Greece, LNCS 4227. Berlin: Springer-Verlag, 242–256.

Kogut, B. and Metiu, A., 2001. Open source software and distributed innovation. *Oxford Review of Economic Policy*, 17 (2), 248–264.

Koh, J. and Kim, Y.G., 2004. Knowledge sharing in virtual communities: an e-business perspective. *Expert Systems with Applications*, 26 (2), 155–166.

Kuk, G., 2006. Strategic interaction and knowledge sharing in the KDE developer mailing list. *Management Science*, 52 (7), 1031–1042.

Lave, J. and Wenger, E., 1991. *Situated learning: legitimate peripheral participation*. Cambridge, UK: Cambridge University Press.

Lee, G.K. and Cole, R.E., 2003. The Linux kernel development: an evolutionary model of knowledge creation. *Organization Science*, 14 (6), 633–649.

Li, W., *et al*., 2008. Smoothing LDA model for text categorization. *In*: H. Li *et al*., eds., *AIRS 2008*, *LNCS 4993,* 15–18 January, Harbin, China. New York: Springer, 83–94.

Lin, H.-F. and Lee, G.-G., 2006. Determinants of success for online communities: an empirical study. *Behaviour & Information Technology*, 25 (6), 479–488.

Lu, X., *et al*., 2006. Enhancing text categorization with semantic-enriched representation and training data augmentation. *Journal of the American Medical Informatics Association*, 13 (5), 526–535.

Maedche, A. and Staab, S., 2001. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16 (2), 72–79.

Manning, C.D. and Schutze, H., 1999. *Foundations of statistical natural language processing*. Cambridge, MA: MIT Press.

Martínez-Torres, M.R., 2006. A procedure to design a structural and measurement model of intellectual capital: an exploratory study. *Information & Management*, 43 (5), 617–626.

Martínez-Torres, M.R., *et al*., 2010. The role of Internet in the development of future software projects. *Internet Research*, 20 (1), 72–86.

Martínez-Torres, M.R. and Toral, S.L., 2010. Strategic group identification using evolutionary computation. *Experts Systems with Applications*, 37 (7), 4948–4954.

Menon, R., *et al*., 2004. On the effectiveness of latent semantic analysis for the categorization of call centre records. *In*: *Proceedings of the IEEE International Engineering Management Conference*, 18–21 October, Singapore. Vol. 2. Piscataway, NJ: IEEE, 2, 546–550.

Michlmayr, M. and Senyard, A., 2006. A statistical analysis of defects in Debian and strategies for improving quality in free software projects. *In*: J. Bitzer, Philipp, J.H. Schröder, eds. *The economics of open source software development*. Amsterdam, the Netherlands: Elsevier, 131–148.

Mika, P., 2007. *Semantic web and beyond: computing for human experience*. New York: Springer.

Moler, C.B., 2004. *Numerical computing with MATLAB*. Philadelphia, PA: Mathworks Inc.

Newman, D., *et al*., 2006. *Analyzing entities and topics in news articles using statistical topic models* LNCS 3975. New York: Intelligence and Security Informatics, Springer.

Ng, A.Y., Jordan, M., and Weiss, Y., 2001. On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press, 849–856.

Pan, S.L. and Leidner, D.E., 2003. Bridging communities of practice with information technology in pursuit of global knowledge sharing. *Journal of Strategic Information Systems*, 12, 71–88.

Rheingold, H., 1993. *The virtual community: homesteading on the electronic frontier*. Reading, MA: Addison-Wesley.

Register, A.H., 2007. *A guide to MATLAB object-oriented programming*. Boca Raton, FL: Chapman & Hall/CRC Press.

Rencher, A.C., 2002. *Methods of multivariate analysis*. *Wiley Series in Probability and Statistics*. 2nd ed. New York: John Wiley & Sons.

Rigutini, L. and Maggini, M., 2005. A semi-supervised document clustering algorithm based on EM. *In*: *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, 19–22 September, Compiègne, France. Piscataway, NJ: IEEE, 200–206.

Salto, G. and McGill, M.J., 1983. *An introduction to modern information retrieval*. New York: McGraw-Hill.

Shi, J. and Malik, J., 2000. Normalized cuts and image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 22 (8), 888–905.

Sowe, S., Stamelos, I., and Angelis, L., 2006. Identifying knowledge brokers that yield software engineering knowledge in OSS projects. *Information and Software Technology*, 48 (11), 1025–1033.

Stevens, J., 1992. *Applied multivariate statistics for the social sciences*. 2nd ed. Mahwah, NJ, USA: Lawrence Erlbaum.

Steyvers, M. and Griffiths, T., 2007. Probabilistic topic models. *In*: T. Landauer *et al*., ed. *Handbook of latent semantic analysis*. Hillsdale, NJ: Erlbaum.

Toral, S.L., Martínez-Torres, M.R., and Barrero, F., 2009a. Modelling mailing list behaviour in open source projects: the case of ARM embedded Linux. *Journal of Universal Computer Science*, 15 (3), 648–664.

Toral, S.L., Martínez-Torres, M.R., and Barrero, F., 2009b. Virtual communities as a resource for the development of OSS projects: the case of Linux ports to embedded processors. *Behavior and Information Technology*, 28 (5), 405–419.

Toral, S.L., Vargas, M., and Barrero, F., 2009c. Embedded multimedia processors for road-traffic parameter estimation. *Computer*, 42 (12), 61–68.

Toral, S.L., Martínez-Torres, M.R., and Barrero, F., 2009d. An empirical study of the driving forces behind online communities. *Internet Research*, 19 (4), 378–392.

Toral, S.L., Martínez-Torres, M.R., and Barrero, F., 2010. Analysis of virtual communities supporting OSS projects using social network analysis. *Information and Software Technology*, 52 (3), 296–303.

Wai, F.B., 2008. Reuse of knowledge assets from repositories: a mixed methods study. *Information & Management*, 45 (6), 365–375.

Weal, M.J., *et al*., 2007. Ontologies as facilitators for repurposing web documents. *International Journal of Human–Computer Studies*, 65, 537–562.

Wellman, B. and Gulia, M., 1995. *Net surfers don't ride alone: virtual communities as communities*. Berkeley: University of California Press.

Wenger, E.C. and Snyder, W.M., 2000. Communities of practice: the organizational frontier. *Harvard Business Review*, 78 (1), 139–144.

Xu, W. and Gong, Y., 2004. Document clustering by concept factorization. *In*: *Proceedings of international conference research and development in information retrieval*. New York: ACM, 202–209.

Xu, W., Liu, X., and Gong, Y., 2003. Document clustering based on non-negative matrix factorization. *In*: *Proceedings of international conference research and development in information retrieval*. New York: ACM, 267–273.

Zha, H., *et al*., 2001. Spectral relaxation for k-means clustering. *Advances in neural information processing systems 14*. Cambridge, MA: MIT Press, 1057–1064.

Zhou, S., Li, K., and Liu, Y., 2008. Text categorization based on topic model. *In*: G. Wang, T. Li, J.W. Grzymala-Busse, D. Miao, A. Skowron, and Y. Yao, eds. *Proceedings of the 3rd international conference on rough sets and knowledge technology*, 17–19 May, Chengdu, China. Lecture notes in computer science. Berlin: Springer-Verlag, 572–579.

Appendix

Representing the content of words and documents with probabilistic topics has one distinct advantage over the purely spatial representation of LSI. Each topic is individually interpretable, providing a probability distribution over a word that picks out a coherent cluster of correlated terms. Words are the only

observable variables and they implicitly reflect the latent structure. Each topic is on the basis of the random variable θ that is sampled from a Dirichlet distribution $p(\theta, \alpha)$ where α is a hyper parameter. The topic z conditioned on θ and the word w conditioned on the topic and on ϕ (word distribution over topics) are sampled from multinomial distributions $p(z_n|\theta)$ and $p(w_n|z_n;\phi)$, respectively. The probability of a document can be computed as:

$$p(w) = \int_{\theta} [\Pi_{n=1}^{N} \Sigma_{z_n=1}^{k} p(w_n|z_n;\phi)p(z_n|\theta)]p(\theta;\alpha)d\theta \tag{A1}$$

In the implementation of LDA proposed by Blei *et al*. (2003), α and ϕ are learnt by variational inference to maximise the log likelihood of the data. An easier implementation has been proposed by Griffiths and Steyvers (2004) introducing a simple modification to the model. A Dirichlet prior is introduced on the parameter ϕ, with hyper parameter β. Despite this modification, computation of the conditional probability $p(z|w)$ is still unmanageable. As a solution, they propose to approximate it by Gibbs sampling based on the following distribution:

$$p(z_i = j|z_{i-1}, w) \propto \frac{n_{-i,j}^{(w_i)} + \beta}{n_{-i,j}^{(\cdot)} + w\beta} \frac{n_{-i,j}^{(d_i)} + \alpha}{n_{-i}^{(d_i)} + \tau\alpha} \tag{A2}$$

This distribution represents the probability that word w_i should be assigned to topic j given all other assignments z_{-i}. The quantities $n_{-i,j}^{(w_i)}$ and $n_{-i,j}^{(\cdot)}$ represent the number of times the word w_i has been already assigned to topic j and the total number of words assigned to topic j, respectively. The quantities $n_{-i,j}^{(d_i)}$ and $n_{-i}^{(d_i)}$ represent the number of times the word w_i in the document d_i has been already assigned to topic j and the number of words in document d_i that are assigned to topic j. The hyper parameters α and β are computed using the method described by Griffiths and Steyvers (2004), that is, $\beta = 0.01$ and $\alpha = 50/T$.

Considering T topics, the probability of the i-th word in a given document can be written as:

$$P(w_i) = \sum_{j=1}^{T} P(w_i|z_i = j)P(z_i = j) \tag{A3}$$

where z_i is a latent variable indicating the topic from which the ith word was drawn, and $P(w_i|z_i = j)$ is the probability of word w_i under the jth topic. $P(z_i = j)$ gives the probability of choosing a word from topic j in the current document, which will vary across different documents. Intuitively, $P(w|z)$ indicates which words are important to a topic, whereas $P(z)$ is the prevalence of those topics within a document. LDA combines Equation (A1) with a prior probability distribution to provide a complete generative model for documents.