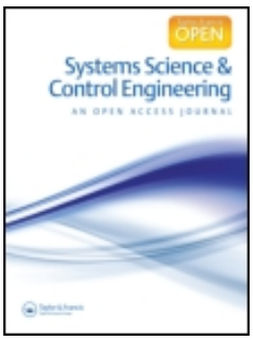

Systems Science & Control Engineering: An Open Access Journal

ISSN: (Print) 2164-2583 (Online) Journal homepage: https://www.tandfonline.com/loi/tssc20

Text stream mining for Massive Open Online Courses: review and perspectives

Safwan Shatnawi, Mohamad Medhat Gaber & Mihaela Cocea

To cite this article: Safwan Shatnawi, Mohamad Medhat Gaber & Mihaela Cocea (2014) Text stream mining for Massive Open Online Courses: review and perspectives, Systems Science & Control Engineering: An Open Access Journal, 2:1, 664-676, DOI: 10.1080/21642583.2014.970732

To link to this article: https://doi.org/10.1080/21642583.2014.970732

Published online: 31 Oct 2014.

Systems Science & Control Engineering: An Open Access Journal, 2014
Vol. 2, 664–676, http://dx.doi.org/10.1080/21642583.2014.970732

Taylor & Francis
Taylor & Francis Group

Text stream mining for Massive Open Online Courses: review and perspectives

Safwan Shatnawi[a,b*], Mohamad Medhat Gaber[a] and Mihaela Cocea[c]

[a]*School of Computing Science and Digital Media, University of Robert Gordon, Scotland, Aberdeen, UK;* [b]*College of Applied Studies, University of Bahrain, Sakhair Campus, Bahrain;* [c]*School of Computing, University of Portsmouth, UK*

(*Received 20 November 2013; final version received 25 September 2014*)

Massive Open Online Course (MOOC) systems have recently received significant recognition and are increasingly attracting the attention of education providers and educational researchers. MOOCs are neither precisely defined nor sufficiently researched in terms of their properties and usage. The large number of students enrolled in these courses can lead to insufficient feedback given to the students. A stream of student posts to courses' forums makes the problem even more difficult. Students'–MOOCs' interactions can be exploited using text mining techniques to enhance learning and personalise the learners' experience. In this paper, the open issues in MOOCs are outlined. Text mining and streaming text mining techniques which can contribute to the success of these systems are reviewed and some open issues in MOOC systems are addressed. Finally, our vision of an intelligent personalised MOOC feedback management system that we term *iMOOC* is outlined.

Keywords: data mining; artificial intelligence; intelligent systems

1. Introduction

Evolution in computer hardware and software increases the amount of generated and stored data. This unbridled growth of data creates the need to reveal patterns in our business and scientific aspects. Statistical and machine learning techniques have been used to learn and discover hidden patterns in data sets. As a result, the data mining field emerged and flourished.

Data mining is defined as an automatic or semiautomatic analysis of substantial quantities of data stored in databases, text, or images to discover valid, useful, and understandable patterns. This in turn allows nontrivial prediction on unseen data (Liu, 2007; Witten & Frank, 2005). Most common tasks of data mining are classification, clustering, association rule mining, and sequential pattern mining (Liu, 2007).

Traditional data mining uses structured data found in relational databases, spreadsheets, or structured text files. However, due to the staggering volume of text documents and web pages, researchers started to apply traditional data mining techniques to web documents and text documents. As a result, the web and text mining fields emerged. Unlike traditional data mining, text mining and web mining deal with heterogeneous, unstructured, or semi-structured data (Liu, 2007).

The advent of web forums, blogs, and social network sites, such as Facebook, MySpace, and Cyworld, allow users to interact with these sites and to send comments or feedback. As a large volume of users interact with these systems which generate massive volumes of continuous streaming data, researchers focus on stream mining and social network analysis and mining. In stream data massive volumes of continuous structured and unstructured data arrive at high speed and require real-time analysis (Aggarwal, 2011; Gaber, Zaslavsky, & Krishnaswamy, 2005). Data stream processing has its own challenges such as a limited amount of memory and the fact that data points are accessed in the order they arrive, that is, random access to the data points is not allowed (O'Callaghan, Mishra, Meyerson, Guha, & Motwani, 2002).

A significant number of e-learning systems do exist on the Internet. These systems benefited from findings and techniques of data mining and text mining, which led to the emergence of the educational data mining (EDM) field. EDM aims to provide better experiences to learners when they interact with these systems. The advent of e-learning 2.0 systems created new challenges for EDM. Social learning is adopted using social software such as blogs, forums, and wikis. These systems allow learners to engage in the teaching process; moreover, it allows learners to participate in peer grading which adds more challenges to the credibility of these systems.

In order to compete globally higher education institutions must improve their services to attract and maintain students. E-learning solutions are key parts in increasing and maintaining student interest, interactivity, and motivation. E-learning solutions aim to make learning more efficient. These systems strive to personalise learners'

*Corresponding author. Email: safwan.shatnawi@gmail.com

interactions with e-learning systems which in turn motivate students and support institutions to achieve efficiency. To personalise learners' interactions, learners' behaviours are analysed to deeply understand learners and enhance the learning processes which are the main objectives of EDM.

The volume of learners enrolled in educational systems has dramatically increased from hundreds to thousands or even millions (Coursera for example). Each class has its own forums. Learners send massive volumes of comments, questions, or answers on a daily basis. E-learning systems that are scalable and open are called Massive Open Online Course (MOOCs) which allow a large number of learners to interact with these systems (Kop, Fournier, & Mak, 2011). This creates massive volumes of text stream where streaming data clustering can contribute to enhancing learners–educator/system interaction. Also, topic detection in streaming data can give a summary of what is going on in these systems. These techniques are the main components of our proposed system to manage MOOC systems. More details of our proposed system are given in Section 6.

E-learning systems that are scalable and open are called MOOCs which allow a large number of learners to interact with these systems (Kop et al., 2011). This creates massive volumes of text stream where streaming data clustering can contribute to enhancing learners–educator/system interaction. Also topic detection in streaming data can give a summary of what is going on in these systems. These techniques are the main components of our proposed system to manage MOOC systems. More details of our proposed system are given in Section 6.

MOOCs are new phenomena in e-learning, which started in 2008. Currently, two types of MOOCs can be distinguished: cMOOCs and xMOOCs. The former represents the early style of MOOCs which is based on connectivisim and networking, while the latter, that is, xMOOCs, belongs to the behaviouristic learning approaches. The latter has been adopted by prestigious institutions such as MIT and Stanford. MOOCs share many features with traditional e-learning systems, however, they have their own characteristics which are that they are free, open access, and set no upper limit to the number of enrolled learners (Daniel, 2012). The large volume of learners participating in MOOCs generates a massive volume of stream text that cannot be handled by a small group of academics. Therefore, a need arises to develop new or to tailor existing stream techniques to manage stream data in MOOCs.

Clustering, as aforementioned, is one of the data mining tasks. It is defined as the process of grouping data instances based on a defined proximity function. Data instances are also referred to as data objects or data points. Different types of clustering algorithms can be used to cluster data instances. We can classify these algorithms as partitioning, overlapping, subspace, and hierarchical algorithms (Liu, 2007). While clustering offline data instances almost reached a stable state, clustering stream data is still a challenging and prominent topic because streaming data needs to deal with continuous events happening rapidly. Time, memory, and large volume constraints contribute to the challenge of clustering stream data (Liu, Cai, Yin, & Fu, 2008).

In this research we give an overview of MOOCs and the challenges their features present for learning and personalisation. Subsequently, we present a systematic review of data mining algorithms which can be used to process learners' streaming forum posts, automate assessment evaluations for massive volumes of learners, and provide online feedback to learners. Using preset content in e-learning systems is another objective of this research. We aim to utilise text/stream clustering, data mining, and machine learning theories and techniques to achieve these objectives.

The remaining of this paper is organised as follows. Section 2 describes MOOCs and their features. Section 3 gives an overview of the text mining field, while Section 4 describes the state-of-the-art algorithms in text mining. Topic detection is presented in Section 5. Section 6 presents our proposed system architecture for managing MOOCs' feedback, and, finally, Section 7 summarises the paper and outlines directions for future work.

2. Massive Online Open Courses

MOOCs are new phenomena in the higher education field. Despite attracting a great deal of attention in the last couple of years, there is very little research into the various aspects of MOOCs and their usage. In this section, MOOCs are described in detail and their features are outlined. Moreover, potential areas for research in MOOCs and the associated research challenges are discussed.

The development of MOOCs has its roots back to 2001–2002 when William and Flora Hewlett founded the Carnegie Mellon University Open Learning Initiative and the MIT Open Courseware project, which freely offered course materials from these institutions online under Creative Commons licenses (Open Learning Initiative, 2013). The term MOOC was coined by David Cormier and Bryan Alexander at the University of Manitoba in 2008. In 2012, Edx which is a joint project between Harvard and MIT was established to offer open courses online; Udacity and Coursera also appeared in 2012. Currently, more institutions started offering MOOCs.

MOOCs have similarities to an ordinary course, such as a predefined timeline and a weekly breakdown of topics. However, MOOCs have no fees, no prerequisites other than Internet access, no predefined expectations for participation, and generally no accreditation, that is, no credit or certificate offered for completion.

MOOCs have become a hot topic in higher education. E-learning and distance learning are well-known concepts in the educational field. In addition, the use of technology, such as radio and TV broadcasting, and the Internet,

has been practised for some time. However, MOOCs are different in many aspects. Two of the most important characteristics are that MOOCs are free, that is, institutions offer courses with no tuition fees, and that they are open, that is, students can enrol with no prerequisite. The success of MOOCs is due to its adoption by prestigious institutions, offering opportunities to make education accessible and affordable, and to the availability of the Internet, tablets, and smart phones. As a result, we have the massiveness feature of MOOCs.

Higher education has many challenges. Among these challenges are access, cost, and quality. MOOCs addressed and successfully resolved the access and cost challenges. However, the third challenge, which is quality, is the major controversial topic (Mazoue, 2013). Some higher education researchers criticise the quality of MOOCs (Vardi, 2012). Their view is that MOOCs lack a sophisticated learning architecture. In addition, they criticise the feedback and communication management in MOOCs. In current MOOC settings, instructors will not be able to interact with all students to answer their questions and comments. On the other hand, MOOCs support peer-to-peer interaction; however, this is not suitable for all types of courses (Mazoue, 2013).

Educational researchers who support the new phenomenon, see it as a solution for higher education challenges and a victory of democracy in education. They believe that the findings and the results of EDM, intelligent tutoring systems, and analytical learning researches will contribute to the success of MOOCs and will enhance the communication and feedback management. Table 1 summarises the advantages and limitations of MOOCs based on pro-/anti-MOOCs' perspectives (Cooper & Sahami, 2013; Hyman, 2012; Kaczmarczyk, 2013; Mazoue, 2013; Vardi, 2012).

Table 1. Advantages and limitation of MOOCs.

Advantages	Limitations
More effective than a professor monologuing to a large class	Inability of educators to assess student learning
It offers quizzes for retrieval practice which is an established method to improve learning	No accreditation
Open opportunities for millions of people who cannot access universities	Validation and plagiarism
Provides global access to education and can be scheduled to work with family and personal commitments	Lack of in-depth evaluation models to evaluate projects and assignments
Can be used as support materials for face to face courses	Lack of effective communication and feedback

2.1. *iMOOCs*

As aforementioned, the MOOC concept is relatively new. It does not have standard abbreviations for its terminologies. Many researchers used some abbreviations to describe MOOCs or to describe frameworks and systems for managing MOOCs.

As a result, some abbreviations such as iMOOC are referred to as internal MOOC or interactive MOOC. Internal MOOCs are courses offered by institutions which are open to all students within these institutions. However, courses are not available for students outside these institutions. Politecnico di Milano and National University of Singapore are examples of these institutions.

On the other hand, interactive MOOC aims to add more interactivity to the MOOC to increase learners' engagement, as a result, overcoming high drop-out rates problem in earlier MOOC settings. Interactivity is achieved by adding real-world simulating objects, games, or other interactive objects. Interactive MOOCs are built around the virtual pedagogical model. The premises underlying this model are interaction, flexibility, individual centeredness, and digital inclusion (Breslow et al., 2013; Weller, 2011).

Currently, we are not able to judge the correctness of any perspective due to the lack of in-depth educational research that is in favour of any of the aforementioned perspectives. However, in this paper, we advocate that the disadvantages of MOOCs can be addressed through data mining research. Therefore, we use the term iMOOC to stand for *intelligent MOOC*.

In particular, in this paper, we give an overview of data mining techniques that can contribute to solving the following questions:

- How can we reach the quality of individual tutoring with massiveness feature of MOOCs?
- How can we enhance the communication management between students and educators?
- How can we create a pedagogy that is structured and rich in feedback loops?

One of the most common criticisms of all MOOCs is the lack of direct student to professor communication. When a professor teaches hundreds of thousands of students, how does he communicate with them? When a student takes a MOOC, how does he reach out to his professor? We argue that communication and feedback in MOOCs can be improved by using data mining algorithms. In this paper, we give an overview of data mining algorithms that can be used to enhance the communication in the MOOCs between students and educators.

3. Text mining

As defined earlier, data mining aims to discover valid and useful information which allows nontrivial prediction.

Structured data can be easily mined, however, unstructured data mining such as text documents or stream text needs more intensive work before mining algorithms can be applied.

Many algorithms were introduced to mine text data which are information extraction, text summarisation, supervised learning, unsupervised learning, dimensionality reduction, transfer learning, probabilistic techniques, cross-lingual mining, and text stream mining (Liu, 2007).

In this section, we present methods that are related to mining text in MOOCs; these are unsupervised learning and streaming text mining.

Unsupervised learning methods do not require any manual labelling of the training data which is labour-intensive work. Manual labelling of the training data is used in other algorithms such as supervised learning and information extraction algorithms. The commonly used methods in unsupervised learning algorithms are clustering and topic modelling (Aggarwal, 2012).

Clustering is the process of grouping data instances based on similarity (Liu, 2007). Clustering methods were designed for quantitative and categorical data. However, general clustering algorithms such as k-means were developed to work with text data as well. Native clustering methods do not work effectively for text data since text data is sparse and has high dimensionality. Hence, text clustering requires designing text-specific clustering algorithms. On the other hand, topic modelling aims to overcome the computational inefficiency feature of text clustering.

Feature selection is the first step in text mining. This process is crucial to the quality of text mining methods. Noisy features must be eliminated before starting the clustering process. On the other hand, relevant features need to be identified. Various feature selection approaches were used in text mining such as frequency-based selection, term-strength selection, term contribution, and entropy-based ranking. Another method in text preprocessing is feature transformation which aims to improve the quality of document representation. These methods include latent semantic indexing (LSI), non-matrix factorisation, and probabilistic latent semantic analysis (PLSA) (Aggarwal & Zhai, 2012).

4. Text-clustering algorithms

In this section, we provide an overview of the state-of-the-art on text-clustering algorithms. Text documents are clustered based on a similarity function. Different similarity functions have been used in text clustering. A popular similarity function is cosine similarity. In addition, heuristic functions such as term frequency (TF), inverse document frequency, and document length normalisation have been used to optimise similarity functions (Aggarwal & Zhai, 2012). Probabilistic models of text represent text documents as probability distributions over words. In these models, similarity is obtained according to a theoretic measure of information (Zhai, 2008).

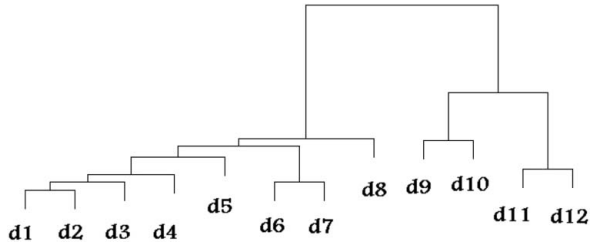

Figure 1. Dendogram of text documents.

4.1. *Agglomerated hierarchical algorithms*

Agglomerated hierarchical algorithms were used extensively in clustering quantitative and categorical data, and were later found to be also suitable for text data. Agglomerated clustering algorithms start with individual documents in the corpus as initial clusters, where each document represents a cluster. Subsequently, similar documents are merged in higher level clusters until all documents are grouped in one big cluster. This process is illustrated as a dendogram, such as the one in Figure 1.

According to Murtagh & Contreras (2012) hierarchical algorithms fall into three categories which are linkage and centroid, median, and minimum variance methods. Hierarchical linkage-based methods can be categorised in one of the following three similarity approaches (Murtagh & Contreras, 2012):

- Two groups of clusters are merged if they have the least interconnecting dissimilarity among all other documents pairs which is called *single-linkage clustering*. It is an extremely efficient method for clustering text documents. However, it suffers from the drawback of chaining, a phenomenon in which incompatible documents are grouped in the same cluster. As a result, it can generate poor-quality clusters.
- Instead of clustering documents based on the maximum similarity among document pairs, clusters are obtained by computing the average similarity of all possible combinations of document pairs of the clusters, a method known as *group-average linkage clustering*. The more documents in the clusters are, the less efficient this method becomes; however, it generates better quality clusters.
- Two groups of clusters are merged based on the worst-case similarity between two pairs of documents. Although this method overrides the chaining phenomenon which exists in a single-linkage clustering method, it is computationally more expensive than the aforementioned linkage methods; this method is known as *complete-linkage clustering*.

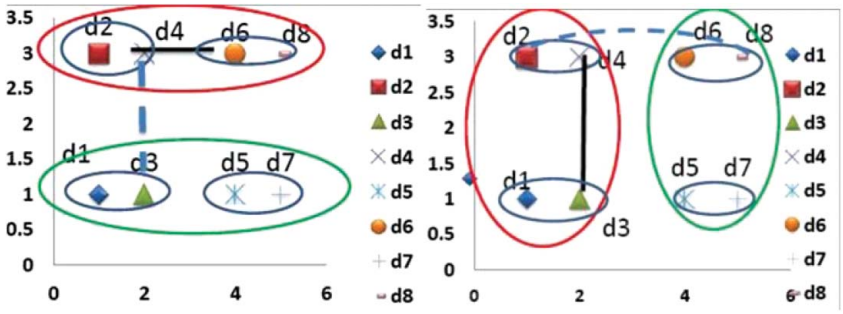

Figure 2. Agglomerated document clustering. (a) Single-linkage document clustering and (b) complete-linkage document clustering.

Figure 2 shows single-linkage clustering (a) and complete-linkage clustering (b). In the single-link method, the similarity between the upper two-point clusters is the distance between points d4 and d6 (the solid line). This similarity is greater[1] than the single-link similarity of the two left two-point clusters, which is calculated as the distance between d4 and d3 (the dashed line).

On the other hand, the complete-link similarity of the upper two-point clusters is the distance between points d2 and d8 (the dashed line). This similarity is less than the complete-link similarity of the left two-point clusters, which is the distance between d3 and d4 (the solid line). In both single-link and complete-link clustering algorithms, we obtained two clusters; however, each cluster contains different document sets.

4.2. *Partitional clustering algorithms*

Partitional clustering methods create flat (one level) partitioning of the data points (text documents). These methods find all desired clusters at once. K-means and k-medoid are two of the most used algorithms with text data.

The former starts with a set of kernels documents not necessarily from the original corpus; each of these documents is used to build the cluster by assigning documents in the corpus to one of these kernels using closest similarity. In the next iteration, the original kernel is replaced by the centroid of the previously formed clusters. The algorithm is terminated when convergence is achieved.

In the latter, the kernels are selected from the original documents in the corpus and then the clusters are built around these kernels. Each document then is assigned to the closest kernel using average similarity of each document to its closest kernel. Iteratively the algorithm improves the kernels using randomised interchanges. An objective function is used to determine whether the interchange process improves the cluster or not in each iteration. Once a convergence is achieved the algorithm is finished.

Performance-wise, k-means generally outperforms k-medoids and generates better quality clusters, mainly because k-means requires fewer iterations to converge. Additionally, k-medoids works inefficiently with sparse data (Aggarwal & Zhai, 2012). A variation of the k-means algorithm, called "bisecting" k-means, was also used with text documents. A comparative study (Steinbach, Karypis, & Kumar, 2000) found that bisecting k-means outperforms the original k-means algorithm and that it is as good as, or better than agglomerated clustering algorithms for variant evaluation measures.

Figure 3 illustrated aspects of partitional clustering algorithms. In (a) clusters generated by the k-means algorithm are displayed, while (b) shows how the cluster centroid is changing after each iteration. μ_i has been designated as the cluster centroid. The initial centroid is μ_0, while after four iterations μ_3 is the new cluster centroid.

4.3. *Hybrid text clustering*

Hierarchical clustering algorithms tend to be less efficient because they are computationally expensive; however, they tend to generate robust clusters. In contrast, partitioning algorithms are computationally efficient, but are less effective in terms of the quality of the generated clusters. Many attempts were introduced to improve both efficiency and effectiveness of text-clustering algorithms. It was proved that the initial selection of the seeds for the k-means algorithm significantly contributes to the quality of the generated clusters. As a result, many hybrid algorithms (Cutting, Karger, Pedersen, & Tukey, 1992; Luo, Li, & Chung, 2009) attempted to find good initial seeds for the k-means algorithm. Others (Cutting et al., 1992) proposed algorithms to refine cluster centroids, claiming that this refinement will enhance the effectiveness of the generated clusters. In the following, we overview the algorithms with the most significant improvements.

The clustering algorithm proposed in Cutting et al. (1992) starts by finding good initial seeds for the k-means algorithm. This is achieved by implementing two methods which are *buckshot and fractionation* as they called in Cutting et al. (1992).

The former randomly selects $\sqrt{kn}$ documents, where k is the number of desired clusters and n is the number of documents in the corpus. Next, an agglomerated algorithm is used to cluster this subgroup into k clusters, where the centroid of each cluster forms a seed for the k-means

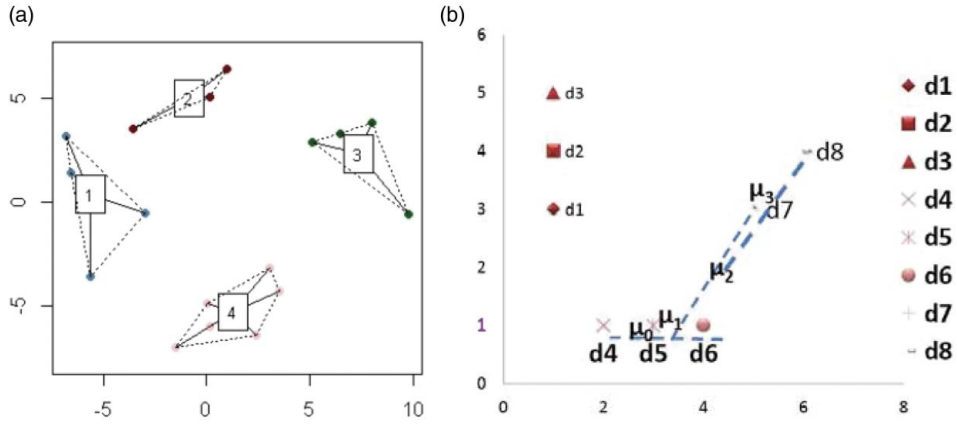

Figure 3. Partitional document clustering. (a) Clusters generated by k-means and (b) cluster-generating process.

algorithm. Multiple runs of this algorithm against the same corpus will not generate the same partitions. However, in practice, Cutting et al. (1992) found that multiple runs gave qualitatively similar partitions.

The latter brakes the corpus into fixed size groups, each with the size of n/m, where $m > k$. Next, an agglomerated algorithm will produce z clusters for each group. As a result, we will have $z \times m$ clusters. Each cluster is considered as an individual document by merging all documents in that cluster. This process is repeated until k clusters are obtained. These form the seeds for the k-means algorithm, where every document is assigned to the nearest cluster and the cluster centroid is modified after assigning the document to the cluster. As a result, the new centroid replaces the old one and is used as a seed in the next iteration.

4.4. Frequent term-based text clustering

One of the main challenges for text clustering is the large dimensionality of the document vector space. Frequent term-based clustering (FTC) methods cluster documents based on a subset of frequent terms, instead of the whole terms in the collection. The frequent item set is obtained from association rule mining. There are many algorithms for this purpose; more details can be found in Agrawal & Srikant (1994), Han, Pei, & Yin (2000), and Zaki (2000).

FTC algorithms consider each selected subset of frequent term sets as cluster descriptions, while the documents covering the subsets of frequent terms represent the cluster itself.

Two types of algorithms for text clustering based on frequent terms can be distinguished (Beil, Ester, & Xu, 2002): FTC and hierarchical frequent term-based clustering (HFTC).

The former is a bottom-up flat clustering algorithm which starts with an empty set of clusters. In every iteration, it selects one of the cluster descriptions (one set of frequent term sets) that have a minimum overlap with other clusters. The selected set will be removed from the database and the documents covering it are also removed from the document collection. The algorithm ends when all documents in the collection are clustered. This approach generates clusters with no overlap.

The latter algorithm exploits the monotonicity property of the frequent item set where all $k-1$ subsets of frequent k-terms are also frequent. It starts with one big cluster containing all documents. In the next iteration, it clusters the documents based on frequent 1-term sets. Then, it uses 2-terms sets and continues until no more frequent k-terms exist. The clusters generated by this algorithm are overlapped.

4.5. Graph-based text clustering

Using the graph model for clustering dates as far back as 1959 (Augustson & Minker, 1970). In graph-based models, the maximum complete subgraph of a graph is defined as a cluster.

In Dhillon (2001), a method to cluster text documents and words also known as co-clustering was introduced based on a bipartite graph structure. Documents and words represent vertices, while E is the set of edges between documents and words. In this structure there are no edges between words, nor between documents; only document to word edges exist. Edges are positively weighted, where the weights represent the word frequency in a document. To cluster documents, a cut function is defined for partition vertex set V: $\text{cut}(v_1, v_2) = \sum_{i \in v_1, j \in v_2} M_{ij}$.

Finding the minimum cut for set V is an nondeterministic polynomial (NP) complete problem. However, heuristic methods such as *spectral graph bipartitioning* can be used to solve this problem. As a result, V is partitioned into two nearly equally sized subsets V_1^* and V_2^* and this will give the document clusters. Word clustering is obtained by assigning words to the greatest edge weight connected document and simultaneously performs the k-means algorithm to obtain the bipartition.

Another graph-based approach was introduced in Aslam, Pelekhov, & Rus (2006), where they represented the documents in the corpus using a similarity graph G. The cosine similarity between documents is calculated and a set of weights E is obtained for the document set D. For each edge between documents $d_i, d_j \in D$ the weight $e_{ij} \in E$ represents the similarity value.

Unlike the work in Dhillon (2001) where edges exist between words and documents only, in Aslam et al. (2006) edges exist between documents only. A similarity ratio σ is set which represents the minimum threshold, that is, all edges under σ are ignored. Given the G_σ subgraph the highest similarity edge is set as the centre of the cluster (star as called in their work). All connected vertices (satellites) to this star form a cluster. The similarity between a star and its satellites is guaranteed, however, similarities between satellites are not guaranteed. Although similarity between satellites has not been proven mathematically, the authors claimed that experimental results show similarities among the satellites.

The Neighbours-based clustering algorithm is also a graph-based algorithm proposed in Luo et al. (2009) to select well-separated initial seeds for the k-means algorithm based on pairwise similarity value, link function value, and number of neighbours of documents in the corpus. It uses a new similarity function for assigning documents to the nearest centroid. Finally, a heuristic function selects the candidate cluster to be split for bisecting k-means.

The first step in this algorithm is to find similarities between pairs of documents (d_i, d_j), for all document pairs in the corpus using cosine similarity. If the similarity value is above the given θ specified by the user, then the pairs of the documents (d_i, d_j) are considered neighbours. The similarity information is represented using $n \times n$ matrix M, where n is the number of documents in the corpus. Each value in this matrix is represented using binary representation where 1 in $M[i,j]$ means that documents d_i and d_j are neighbours and 0 otherwise. The number of neighbours for document d_i denoted by $N(d_i)$ is $\sum_{j=1}^{n} M[i,j]$.

The second function is the link function of document pairs (d_i, d_j) which is the number of common neighbours between d_i and d_j. They calculate the value of the link function by multiplying the ith row by the jth col which is denoted by $\text{link}(d_i, d_j) = \sum_{m=1}^{n} M[i,m] \cdot M[m,j]$. The value of this function is proportionally related to the probability of d_i and d_j belonging to the same cluster.

Next, the algorithm finds candidate seeds for the k-means algorithm by selecting $(k+p)$ documents as candidate seeds set S_c, where k is the number of desired seeds and p any extra number of documents specified by the user. The set of candidate seeds is selected from the first minimum $(k+p)N(d_i)$ value documents. After that the algorithm finds similarity and link values for all document pairs combinations in S_c. Based on these values, it calculates the $rank_{\cos}$ and $rank_{\text{link}}$ for every pair of documents. The sum of the $rank_{\cos}$ and $rank_{\text{link}}$ gives the $rank_{\text{total}}$ value.

4.6. *Other clustering methods*

The winnowing-based text-clustering algorithm was introduced by Schleimer (2003) to find a similar text across documents to detect copy or plagiarism in research and student papers. The algorithm divides the document into k-gram substrings where the k value is specified by the user. Each k-substring is called hash. Some subsets of these hashes will be selected to represent the document fingerprint. When two or more documents share one or more fingerprints, they are considered similar. Based on that, authors in Parapar & Barreiro (2008) proposed a text-clustering algorithm. Experimental results show that winnowing-based text clustering outperforms k-mean and TF representations.

Table 2 shows a summary of the overviewed text-clustering algorithms. The table contains a brief description of every clustering algorithm mentioned in the review, the category of these algorithms, and their limitations and computational complexities.

5. Topic modelling

One of text clustering challenges is the large volume of words (terms) in documents. Many methods emerged to reduce the large volume of the documents by representing documents using a small subset of their words. These words represent the abstract or theme of the document. They can be obtained using statistical modelling of a field known as topic modelling in machine learning and natural language processing (NLP) (Blei, 2012; Landauer, Mcnamara, Dennis, & Kintsch, 2007). Statistical modelling for topic detection and tracking includes but is not limited to LSI and latent Dirichlet allocation (LDA).

5.1. *Probabilistic latent semantic analysis*

The vector space model used to represent documents and words is a high-dimensional sparsely space. Latent semantic analysis (LSA), also called LSI, is an automatic indexing method. It projects documents and words into a lower dimensional space. The projected terms represent the semantic concepts in the documents which hopefully overcome synonyms and polysemy problems where different terms have the same meaning or a term may have different meaning according to the context. This projection allows conceptual level document analysis. LSA has its root in information retrieval for indexing information retrieval system. To project the sparsely dimensional documents–words matrix, LSA uses singular value decomposition (SVD) to project documents and words into k-latent semantic spaces. Similarity between documents is measured using latent semantic space and so are the word

Table 2. Text-clustering algorithms summary.

Algorithm	Category	Description	Limitation and computational efficiency
Single linkage	Hierarchical	Merge data points based on the least interconnect dissimilarity	Chaining phenomenon, computationally efficient. Time complexity O($N \log N$). Space complexity O(N)
Group-average linkage	Hierarchical	Merges data points based on the average similarity of all possible combinations of documents	Overcome chaining phenomenon. Computationally expensive O(N^3). Space O(N^2)
Complete linkage	Hierarchical	Merges data points based on the worst similarity	Overcome chaining phenomenon. Computationally expensive O(N^2). Space O(N)
k-means	Partitioning	Starts by a set of kernels documents not necessarily from the original corpus and build the clusters around these documents using closest similarity. The centroid of the cluster is used in the next iteration. Bisecting k-means is a variation of k-means	Computationally efficient O($N \log N$). Requires few iterations to converge. Outperforms k-medoids
k-medoids	Partitioning	Starts by a set of kernels documents from the original corpus and build the clusters around these documents using average similarity. The quality of the clusters is improved using objective function	Robust clusters generated
Bipartite graph	Graph based	Documents and words are represented as bipartite graph. Cut function is used to cluster documents	NP complete problem. Heuristic function is used for optimal solution
Buckshot and fractionation	Hybrid	k-means-based clustering method. With improvement in the kernels set selection using buckshot and fractionation	Multiple runs of this algorithm generates different clusters
FTC and HFTC	Frequent item	Reduce the dimensionality of documents by representing document using its frequent terms set. FTC is bottom-up clustering. HFTC is top-down clustering	FTC generates clusters without overlapping. HFTC generates overlapped clusters. Both FTC and HFTC outperform k-means and k-medoids
Star-satellites	Graph based	Data points are represented as similarity graph. A cosine similarity function is used. A star is the centre of the cluster. Documents that are above user-defined threshold similarity are satellites	Similarity between satellites is not guaranteed. Theorem exists in this method failed to prove satellites similarities.Time complexity O($N \log^2 N$)
Windowing-based	Partitioning	It divides the document into k-gram sub-strings. Then, a subset of these sub strings represents the document fingerprints. Documents share two or more fingerprints are considered similar and clustered	Outperforms k-means and FTC

similarities. More details about LSA can be found in Aggarwal (2012).

In practice, documents are added to the collection (corpus) rapidly. As a result, the document–term matrix needs updating, which in turn leads to a re-calculation of the latent semantic space to reflect the added documents. Repeating the whole process is computationally inefficient. Instead, two methods have been used which are fold-in and semantic space updating. The former computes the projection of the new documents using the existing LSI, which is computationally efficient. The latter overcomes the outdated models by adding new documents to the collection over time; however, indexing is not guaranteed to provide the best rank approximation.

Probabilistic LSA was introduced by Hofmann (1999); this approach aims to statistically model co-occurrence information by applying a probabilistic framework to discover the latent semantic structure. The latent variables (topics) are associated with observed documents. For formal description, see Hofmann (1999).

5.2. Latent Dirichlet allocation (LDA)

LDA is a generative probabilistic modelling method. In practice, documents contain multiple topics and words distributed over many topics. LDA aims to capture all topics in the documents. It considers a topic as a distribution over words. These topics are assumed to be generated in advance. For each document, LDA is used to draw some topics that cover this document. Then, a topic is assigned to each word in the document and a word is selected from the topic words distribution. In practice,

topics, document topics distribution, and document words distribution over topic are unknown or hidden. Only documents are observed. As a result, the computational problem for topic modelling is to infer all hidden structures given the observed document.

In Blei, Ng, & Jordan (2003), a document collection of scientific research journals from 1880 to 2002 was used. These documents were not labelled and did not have any metadata, that is, only the text of the documents was observed. They assumed that 100 different topics exist in these documents, and they used LDA to infer the word distribution over these topics and the distribution of topics in all documents. They also studied how topics evolved over time.

5.3. *Hierarchical generative probabilistic model*

The hierarchical generative probabilistic model based on the bigram model was introduced in Wallach (2006). Marginal and conditional word counts are obtained from a corpus. The marginal count is the number of times a word occurred in the corpus. The conditional count is the number of times a word w_i immediately followed another word w_j. Unlike LDA where word positions are ignored, in this model each word w_k is predicted based on the word w_{k-1}.

The bigram model based on the marginal and conditional counts predicts w_k given the observed w_{k-1}. This approach integrates bigram-based and topic-based models to achieve a better predictive accuracy over LDA or hierarchical LDA.

In Tam & Schultz (2008), they extended the bigram model introduced by Wallach (2006). They present a correlated bigram LSA approach for an unsupervised language model adaptation for automatic speech recognition. They contributed to the bigram LSA by presenting a technique for topics correlation modelling using Dirichlet-tree prior. An algorithm for bigram LSA training via variational Bayes approach and model bootstrapping is proposed, which is scalable to large language model's settings. Moreover, they formulate the fractional Kneser–Ney smoothing to generalise the original Kneser–Ney smoothing which supports only integral counts.

5.4. *Discriminative probabilistic model*

In a study presented in He, Chang, Lim, & Banerjee (2010), the authors examined the time factor in documents. Instead of representing documents as word vector space only, documents are represented in words and time vector space. A temporal discriminative probabilistic model was proposed for both offline and online topic detection and evaluated it for performance issues. In addition, they investigated several types of topic detection models: deterministic, discriminative and probabilistic mixture, and mixed membership. Experimental results showed that a simple deterministic mixture is more efficient and effective than sophisticated models such as LDA.

The discriminative probabilistic model estimates posterior (conditional) probability of a given topic given an observed document. Adding a temporal element achieves best performance/complexity trade-off. In the offline topic detection model, they assume the existence of a set of features that discriminates documents in the corpus. Stop words and rare words are eliminated from these features. The probability of a new document is obtained by computing the conditional probability of the new document for all sets of discriminative features. On the other hand, online topic detection incrementally examines each incoming document to assess whether it belongs to a new topic or an existing topic. Some researchers refer to this process as evolved topic detection instead of online topic detection (Aggarwal, Han, Wang, & Yu, 2003).

5.5. *Non-probabilistic topic detection*

A non-probabilistic online topic detection technique was introduced in Allan et al. (2005) to cluster news stream. It detects events (topics) and assigns the incoming story to one of the existing topics or creates a new topic if the incoming story contains a new topic.

Each document is represented by the top 1000 weighted words that occur in the story as a vector, using the vector space model. Its similarity to every previous document is calculated using the cosine similarity function. The document is assigned to the nearest neighbour if the similarity value is above a given threshold or a new topic is created if the similarity is below that threshold. The authors explored several techniques to enhance the quality of the topic clusters, such as different weightings for words, different criteria for document selection and penalties. These, however, did not lead to a significant increase in cluster quality.

Finally, when they used the average-link clustering, where every cluster is represented by its centroid, the generated clusters were more robust and computationally efficient.

Table 3 presents a summary of topic detection methods.

6. Intelligent MOOCs feedback management system architecture

In this section, we introduce our proposed MOOCs feedback management system architecture (*iMOOC*), which is depicted in Figure 4. We use the *iMOOC* abbreviation to stand for intelligent MOOC. The system has three basic modules: topic detection, clustering, and feedback.

Students register for one of the MOOCs and interact with the system by viewing or downloading course materials, reading posts of other students and receiving feedback. Students can also send comments or feedback to the system using the MOOCs' forum utility. As each

Table 3. Topic detection techniques summary.

Approach	Category	Description
LSA	Vector space model	Using SVD to project documents and words into lower dimensional space. Fold-in and semantic space updating are two methods to enhance computational efficiency
PLSA	Probabilistic	Statistically model co-occurrences information using aspect model. It applies probabilistic framework to discover latent semantic structure
LDA	Probabilistic	Generative probabilistic modelling aims to capture multiple topics exist in a document
Hierarchical generative model (Bigram)	Probabilistic	Extends LDA where word position and co-occurrence are considered. Bigram model based on marginal and conditional counts is used. Space complexity O(V^2K), V: vocabulary, K: topics
Discriminative model	Probabilistic	Time factor is considered. Documents are represented by words and times vectors. The temporal discriminative probabilistic model is used for online and online documents
Experimental model	Cluster-based	Clustering documents based on the topics exist in these documents. Experimentally many techniques were implemented. Average link clustering outperforms other used clustering techniques

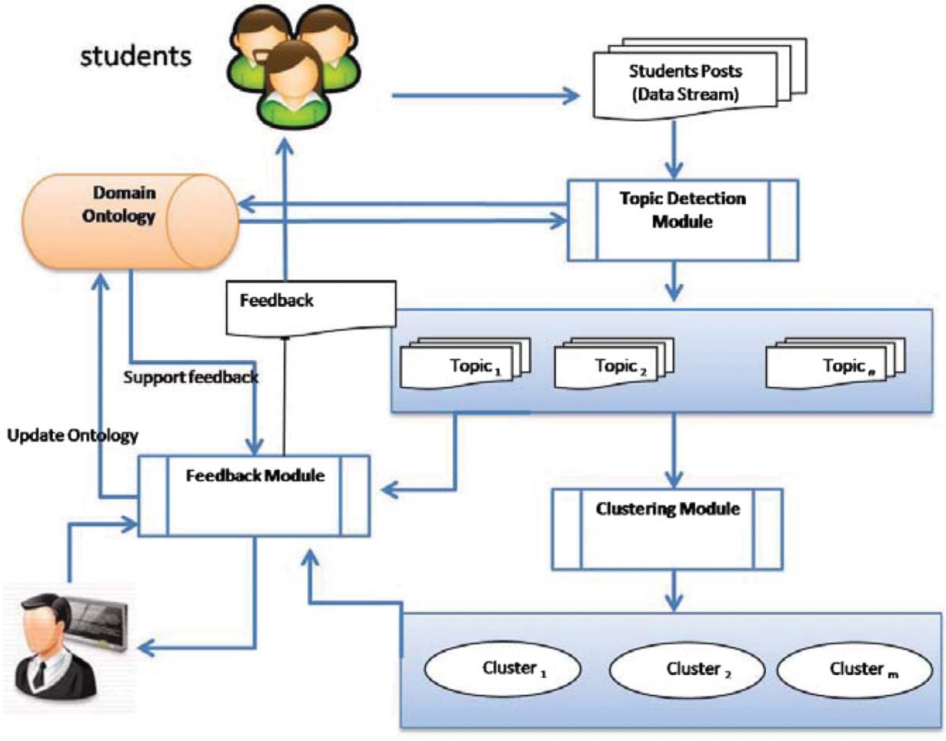

Figure 4. MOOCs feedback management system architecture.

course has a large volume of students, large amounts of streaming text data (exchanges) will be created.

MOOC discussions are a fertile environment for gaining insight into the cognitive process of the learners. Analysis of forums' information enables us to obtain information about participants' levels of content knowledge, learning strategy, or social communication skills.

A variety of participant exchanges exist in MOOC forums. These exchanges include, but are not limited to, getting other participants' help, scaffolding others understanding, or constructing content knowledge between learners. Effective exchanges require communication and content knowledge utilisation and integration. As a result, this leads to successful knowledge-building.

Current MOOC settings do not provide participants (educators and learners) with any kind of analysis of forums' contents. Content analysis aims to describe the attribute of the message or post. An initial step in analysing forums' contents is to identify the topic and the role of the participants. Obviously, this process cannot be performed manually in MOOCs. A variety of techniques can be used to identify the topic of forum posts, which includes clustering, topic detection, or machine learning.

iMOOC processes streaming data using the topic detection module which is responsible for identifying topics in these posts by communicating with the domain ontology. The domain ontology is built based on the course being offered; however, this domain ontology can be expanded to include all existing MOOCs. As a result, students' posts will be grouped into topics ($\text{topic}_1, \text{topic}_2, \ldots, \text{topic}_n$); these topics usually evolve over time, which can be seen as a concept drift. Although this concept drift is not included in the architecture, it is taken into account by the system. After the identification of topics, this information is passed to the clustering module.

The *iMOOC* clustering module groups the topics using hierarchical clustering techniques into ($\text{cluster}_1, \text{cluster}_2, \ldots, \text{cluster}_m$), where n and m need not have to be equal, that is, the number of topics is not necessarily the same as the number of clusters. The third module generates suitable feedback for the students based on the domain ontology, detected topic, and the cluster the post belongs to. In this module, NLP and machine learning are incorporated to generate the feedback and enhance the quality and credibility of the system.

The domain ontology is dynamically changed and enhanced based on the identified topics and students feedback. For students feedback, we assumed that a reasonable command of English exists. Typos will be automatically corrected based on a dictionary module designed for this system. As topics may evolve over time, a concept drift module is embedded in the topic detection module. Also, some topics may fall out of the scope of the course which is detected using outlier clustering methods.

The aforementioned modules work according to the following methodology. We start by building the MOOC ontology; building an ontology is an expensive task in terms of the time and effort involved. Hence, we aim to automate this task by using course text books and text notes as the MOOC domain knowledge. First, we build the term–document matrix for the domain knowledge and then use the most frequent terms as the ontology terms. Next, we find the frequent bigram, trigram, and n-gram expressions to form our MOOC concepts and entity names. The table of contents is used to form the hierarchical representation of the MOOC ontology.

In the topic detection module, we use the entity names obtained in the process of building the domain ontology to construct a deterministic finite automata (DFA). As a result, we have the DFAs state table which is similar to the table used by compilers. Then, the state table is used to parse students' posts (comments, questions, or feedback) and label them. In the case of multiple labels for a post, we consult the hierarchical ontology and get the closest common parent to be the post's label.

In the clustering module, students' posts are clustered based on their labels. A new post is compared to all existing posts in the given cluster to find its semantic similarity to other posts. When a semantic text similarity is found we send the stored feedback to the student. When no similarity is found we send this post to the instructor to assign feedback to it.

Some components of the proposed system have been implemented. The system processes student posts and it identifies content topics and their properties. Course contents were represented using the ontology representation. The system starts by acquiring all course concepts and their relationships. Then for every concept a feedback response is populated in the ontology. After identifying topics and properties, the system sends back the feedback to the student. Experimental results show promising results (Shatnawi, Gaber, & Cocea, 2014).

With this approach, we aim to (a) enhance the learning experience of students using MOOCs by personalising their interaction with the system; (b) provide students with informative feedback by leveraging data mining and NLP techniques; and (c) automate the course ontology building process using data mining and NLP techniques.

7. Summary and future work

MOOCs are new phenomena in e-learning systems and may change the shape of education in the coming few years. Traditional learning management system platforms are not suitable for MOOCs owing mainly to the massive volume of learners. In this paper, techniques that can be used to manage MOOCs and contribute to their success were outlined. Text mining, streaming text mining, and topic detection were discussed, along with how MOOCs can leverage these techniques to personalise students' interactions with MOOC systems. We proposed *iMOOC* to manage MOOCs feedback by using data mining, ontologies, and NLP techniques in order to provide students with automated feedback based on their posts.

The system will be implemented in the future and be validated using real big data obtained from MOOC systems. The proposed system will be tested using real student data obtained from a Coursera offered course in 'Introduction to Databases' with 11,098 students enrolled. Instructors and students posted 21,085 contributions as questions, notes, or peer-feedback (Coursera piazza report for db course, 2013).

A variety of data mining techniques will be used, evaluated, and benchmarked. To personalise students' interactions with *iMOOC*, other information about students' interaction and demographics will be incorporated with

iMOOC to discover a better organisation and management of MOOCs.

Note

1. A greater similarity corresponds to a shorter distance.

References

Aggarwal, C. C. (2011). An introduction to social network data analytics. In C. C. Aggarwal (Ed.), *Social Network Data Analytics* (pp. 1–15). New York, NY: Springer.

Aggarwal, C. C. (2012). Mining text streams. In C. C. Aggarwal & C. Zhai (Eds.), *Mining text data* (pp. 1–10). New York, NY: Springer.

Agrawal, R., & Srikant, R. (1994). *Fast algorithms for mining association rules in large databases*. Proceedings of the 20th international conference on very large data bases. VLDB'94 (pp. 487–499). San Francisco, CA: Morgan Kaufmann.

Aggarwal, C., & Zhai, C. (2012). A survey of text clustering algorithms. In C. C. Aggarwal & C. Zhai, C. (Eds.), *Mining text data* (pp. 77–128). New York, NY: Springer.

Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). *A framework for clustering evolving data streams* (pp. 81–92). Berlin: VLDB Endowment.

Allan, J., Harding, S., Fisher, D., Bolivar, A., Guzman-Lara, S., & Amstutz, P. (2005). *Taking topic detection from evaluation to practice*. Proceedings of the 38th annual Hawaii international conference on system sciences (HICSS'05) – Track 4 (Vol. 4, pp. 101–110). Washington, DC: IEEE Computer Society.

Aslam, J., Pelekhov, E., & Rus, D. (2006). The star clustering algorithm for information organization. In J. Kogan, C. Nicholas, & M. Teboulle (Eds.), *Grouping Multidimensional Data* (pp. 1–23). Berlin Heidelberg: Springer.

Augustson, J., & Minker, J. (1970). An analysis of some graph theoretical cluster techniques. *Journal of the ACM*, *17*(4), 571–588.

Beil, F., Ester, M., & Xu, X. (2002). *Frequent term-based text clustering*. Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining. KDD'02 (pp. 436–442). New York, NY: ACM.

Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77–84.

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

Breslow, L. B., Pritchard, D. E., DeBoer, J., Stump, G. S., Ho, A. D., & Seaton, D. T. (2013). Studying learning in the worldwide classroom: Research into edx's first MOOC. *Research & Practice in Assessment*, *8*, 13–25.

Cooper, S., & Sahami, M. (2013). Reflections on Stanford's MOOCs. *Communications of the ACM*, *56*(2), 28–30.

Coursera piazza report for db course. (2013). Retrieved from https://piazza.com/stats/report/hbtmlzostxhfc.

Cutting, D. R., Karger, D. R., Pedersen, J. O., & Tukey, J. W. (1992). *Scatter/gather: A cluster-based approach to browsing large document collections*. Proceedings of the 15th annual international ACM SIGIR conference on research and development in information retrieval. SIGIR'92 (pp. 318–329). New York, NY: ACM.

Daniel, J. (2012). Making sense of MOOCs: Musings in a maze of myth, paradox and possibility. *Journal of Interactive Media in Education*, *3*(0). Retrieved October 17, 2013, from: http://www-jime.open.ac.uk/jime/article/viewArticle/2012-18/html

Dhillon, I. S. (2001). *Co-clustering documents and words using bipartite spectral graph partitioning* (pp. 269–274). Austin, TX: University of Texas. Retrieved from http://www.ncstrl.org:8900/ncstrl/servlet/search?formname=detail\&id=oai%3Ancstrlh%3Autexas_cs%3AUTEXAS_CS%2F%2FCS-TR-01-05

Gaber, M. M., Zaslavsky, A., & Krishnaswamy, S. (2005). Mining data streams a review. *SIGMOD Record*, *34*(2), 18–26.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *SIGMOD Record*, *29*(2), 1–12.

He, Q., Chang, K., Lim, E. P., & Banerjee, A. (2010). Keep it simple with time: A reexamination of probabilistic topic detection models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(10), 1795–1808.

Hofmann, T. (1999). *Probabilistic latent semantic indexing*. Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval. SIGIR'99 (pp. 50–57). New York, NY: ACM.

Hyman, P. (2012). In the year of disruptive education. *Communications of the ACM*, *55*(12), 20–22.

Kaczmarczyk, L. C. (2013). MOOCs! *ACM Inroads*, *4*(1), 19–20.

Kop, R., Fournier, H., & Mak, J. (2011). A pedagogy of abundance or a pedagogy to support human beings? Participant support on massive open online courses. *International Review of Research in Open and Distance Learning*, *12*(7), 74–93.

Landauer, T. K., Mcnamara, D. S., Dennis, S., & Kintsch, W. (Eds.). (2007). *Handbook of latent semantic analysis*. Mahwah, NJ: Lawrence Erlbaum Associates.

Liu, B. (2007). *Web data: mining exploring hyperlinks, contents, and usage data*. Heidelberg, Berlin: Springer.

Liu, Y. B., Cai, J. R., Yin, J., & Fu, A.W.-C. (2008). Clustering text data streams. *Journal of Computer Science and Technology*, *23*, 112–128.

Luo, C., Li, Y., & Chung, S. M. (2009). Text document clustering based on neighbors. *Data Knowledge Engineering*, *68*(11), 1271–1288. (Including special section: Conference on privacy in statistical databases (PSD 2008) six selected and extended papers on database privacy.)

Mazoue, J. G. (2013). *The MOOC model: Challenging traditional education*. ELI 2013 online spring focus session 2013: Learning and the MOOC, EduCause Review Online. Retrieved from http://www.educause.edu/ero/article/mooc-model-challenging-traditional-education

Murtagh, F., & Contreras, P. (2012). Algorithms for hierarchical clustering: An overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *2*(1), 86–97.

O'Callaghan, L., Mishra, N., Meyerson, A., Guha, S., & Motwani, R. (2002). *Streaming-data algorithms for high-quality clustering*. Proceedings of the 18th international conference on data engineering (pp. 685–694). San Jose, CA.

Parapar, J., & Barreiro, A. (2008). Winnowing-based text clustering. In J. G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D. A. Evans, A. Kolcz, . . . A. Chowdhury (Eds.), *CIKM* (pp. 1353–1354). New York, NY: ACM.

Schleimer, S. (2003). *Winnowing: Local algorithms for document fingerprinting*. Proceedings of the 2003 ACM SIGMOD international conference on management of data 2003 (pp. 76–85). New York, NY: ACM Press.

Shatnawi, S., Gaber, M. M., & Cocea, M. (2014). *Automatic content related feedback for MOOCs based on course domain ontology*. IDEAL. Lecture Notes in Computer Science. Springer. International Publishing AG, Cham. Retrieved from http://link.springer.com/chapter/10.1007%2F978-3-319-10840-7_4#page-1

Steinbach, M., Karypis, G., & Kumar, V. (2000). *A comparison of document clustering techniques* (Technical Report 00-034). University of Minnesota.

Tam, Y. C., & Schultz, T. (2008). Correlated bigram LSA for unsupervised language model adaptation. In D. Koller, D. Schuurmans, Y. Bengio, & L. Bottou (Eds.), *NIPS* (pp. 1633–1640). Vancouver, BC: Curran Associates.

University, C. M. Open Learning Initiative @Online (February 2013).

Vardi, M. Y. (2012). Will MOOCs destroy academia? *Communications of the ACM*, *55*(11), 5–5.

Weller, M. (2011). *The digital scholar: How technology is transforming scholarly practice*. London: A&C Black.

Wallach, H. M. (2006). *Topic modeling: beyond bag-of-words*. Proceedings of the 23rd international conference on machine learning. ICML'06 (pp. 977–984). New York, NY: ACM.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques* (2nd ed.). Morgan Kaufmann Series in Data Management Systems. San Francisco, CA: Morgan Kaufmann.

Zaki, M. J. (2000). Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, *12*, 372–390.

Zhai, C. (2008). Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies*, *1*(1), 1–141.