Web Search Clustering and Labeling with Hidden Topics

Article *in* ACM Transactions on Asian Language Information Processing · August 2009

DOI: 10.1145/1568292.1568295 · Source: doi.acm.org

5 authors, including:

Cam-Tu Nguyen
Nanjing University
29 PUBLICATIONS **399** CITATIONS
SEE PROFILE

Xuan-Hieu Phan
Vietnam National University, Hanoi
62 PUBLICATIONS **1,104** CITATIONS
SEE PROFILE

Quang Thuy Ha
Vietnam National University, Hanoi
90 PUBLICATIONS **626** CITATIONS
SEE PROFILE

Web Search Clustering and Labeling with Hidden Topics

CAM-TU NGUYEN, XUAN-HIEU PHAN, and SUSUMU HORIGUCHI
Tohoku University
and
THU-TRANG NGUYEN and QUANG-THUY HA
Vietnam National University

Web search clustering is a solution to reorganize search results (also called "snippets") in a more convenient way for browsing. There are three key requirements for such post-retrieval clustering systems: (1) the clustering algorithm should group similar documents together; (2) clusters should be labeled with descriptive phrases; and (3) the clustering system should provide high-quality clustering without downloading the whole Web page.

This article introduces a novel framework for clustering Web search results in Vietnamese which targets the three above issues. The main motivation is that by enriching short snippets with hidden topics from huge resources of documents on the Internet, it is able to cluster and label such snippets effectively in a topic-oriented manner without concerning whole Web pages. Our approach is based on recent successful topic analysis models, such as Probabilistic-Latent Semantic Analysis, or Latent Dirichlet Allocation. The underlying idea of the framework is that we collect a very large external data collection called "universal dataset," and then build a clustering system on both the original snippets and a rich set of hidden topics discovered from the universal data collection. This can be seen as a richer representation of snippets to be clustered. We carry out careful evaluation of our method and show that our method can yield impressive clustering quality.

Categories and Subject Descriptors: I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Language models*; *text analysis*

General Terms: Algorithms, Experimentation, Languages

Additional Key Words and Phrases: Latent Dirichlet allocation, hidden topics analysis, Vietnamese, Web search clustering, cluster labeling, collocation, Hierarchical Agglomerative Clustering

This work is supported by the research project QC0706 Vietnamese Named Entity Resolution and Tracking crossover Web Documents and the International Doctoral Program at Tohoku University, Japan.

Author's address: C.-T. Nguyen, Graduate School of Information Sciences, Tohoku University, No 311, Aramaki Aoba 6-3-09, Aoba, Sendai, Miyagi, 980-8579, Japan; email: ncamtu@ecei.tohoku.ac.jp.

ACM Reference Format:
Nguyen, C.-T., Phan, X.-H., Horiguchi, S., Nguyen, T.-T., and Ha, Q.-T. 2009. Web search clustering and labeling with hidden topics. ACM Trans. Asian Lang. Inform. Process. 8, 3, Article 12 (August 2009), 40 pages. DOI = 10.1145/1568292.1568295. http://doi.acm.org/10.1145/1568292.1568295.

1. INTRODUCTION

It has been more than a decade since the first day Vietnam connected to the Internet in 1997. At that time, the Internet served a small group of people but became popular very quickly. In June 2006, VnExpress[1]—one of the most popular electronic newspapers in Vietnamese—appeared in the list of top 100 most accessed sites ranked by Alexa. It has been reported that the number of Internet users has reached 20 million [Vnnic 2008], which accounts for approximately 23% of the population of Vietnam. For efficient access and exploration of such information on the Web, appropriate methods for searching, organizing, and navigating through this enormous collection are of critical need. To this end, there were several emerging Web services such as Baamboo [2008], Socbay [2008], and Xalo [2008], the Web directory Zing [2008] and so on.

Although the performance of search engines is enhanced day by day, it is a tedious and time-consuming task to navigate through hundreds to hundreds of thousands of "snippets" returned from search engines. A study of search engine logs [Jansen et al. 1998] argued that "over half of users did not access result beyond the first page and more than three in four users did not go beyond viewing two pages." Since most search engines display 10 to 20 results per page, a large number of users are unwilling to browse more than 30 results. One solution to manage that large result set is clustering. Like document clustering, search results clustering groups similar "search snippets" together based on their similarity; thus snippets relating to a certain topic will hopefully be placed in a single cluster. This can help users locate their information of interest and capture an overview of the retrieved results easily and quickly. In contrast to document clustering, search results clustering needs to be performed for each query request and be limited to the number of results returned from search engines [Zamir and Etzioni 1999; Ngo 2003]. This adds extra requirements to these kinds of clustering [Zamir and Etzioni 1999]:

—Coherent Clustering: The clustering algorithm should group similar documents together. It should separate relevant documents from irrelevant ones.
—Efficiently Browsing: Descriptive and meaningful labels should be provided to ease user navigation.
—Snippet Tolerance: The method ought to produce high-quality clusters even when it only has access to the snippets returned by the search engines, as most users are unwilling to wait while the system downloads whole documents from the Web.

[1]http://vnexpress.net

These requirements in general and the third one in particular introduce several challenges to clustering. In contrast to normal documents, these snippets are usually noisier, less topic-focused, and much shorter; that is, they contain from a dozen words to a few sentences. Consequently, they do not provide enough shared-context for good similarity measure.

There have been a lot of studies that attempted to overcome this data sparseness to achieve a better (semantic) similarity [Phan et al. 2008]. One solution is to utilize search engines to provide richer context of data [Sahami and Heilman 2006; Bollegala et al. 2007; Yih and Meek 2007]. For each pair of short texts, they use statistics on the results returned by a search engine (e.g., Google) in order to determine the similarity score. A disadvantage is that repeatedly querying search engines is quite time consuming and not suitable for real-time applications. Another solution is to exploit online data repositories, such as Wikipedia[2] or Open Directory Project[3] as external knowledge sources [Banerjee et al. 2007; Schonhofen 2006; Garilovich and Markovitch 2007]. In order to have benefits, the data sources should be in fine structures. Unfortunately, such types of data sources are not available or not rich enough in Vietnamese.

Inspired by the idea of using external data sources mentioned above, we present a general framework for clustering and labeling with hidden topics discovered from a large-scale data collection. This framework is able to deal with the shortness of snippets as well as provide better topic-oriented clustering results. The underlying idea is that we collect a large collection, which we call the "universal dataset," and then do topic estimation for it based on recent successful topic models such as pLSA [Hofmann 1999] or LDA [Blei et al. 2003]. It is worth reminding that the topic estimation needs to be done for a large corpus of long documents (the universal dataset) so that the topic model can be more precise. Once the topic model has been converged, it can be considered as one type of linguistic knowledge which captures the relationships between words. Based on the converged topic model, we are able to perform topic inference for (short) search results to obtain the intended topics. The topics are then combined with the original snippets to create expanded, richer representation. Exploiting one of the similarity measures (such as widely used cosine coefficient), we now can apply any of the successful clustering methods based on similarity such as Hierarchical Agglomerative Clustering (HAC) or K-means [Kotsiantis and Pintelas 2004] to cluster the enriched snippets. The main advantages of the framework include the following points:

—Reducing data sparseness: Different word choices make snippets of the same topic less similar; hidden topics do make them more related than the original. Including hidden topics in measuring similarity helps both reduce the sparseness and make the data more topic-focused.

—Reducing data mismatching: Some snippets sharing unimportant words, which could not be removed completely in the phase of stop word removal,

[2]http://wikipedia.org

[3]http://www.dmoz.org

are likely close in similarity. By taking hidden topics into account, the pairwise similarities among such snippets are decreased in comparison with other pairs of snippet. As a result, this goes beyond the limitation of shallow matching based on word/lexicon.

—Providing informative and meaningful labels: Traditional labeling methods assume that repetitious terms/phrases in a cluster are highly potential to be cluster labels. This is true but not enough. In this work, we use topic similarity between terms/phrases and the cluster as an important feature to determine the most suitable label, thus provide more descriptive labels.

—Adaptable to another languages: The framework is simple to implement. All we need is to collect a large-scale data collection to serve as the universal data and exploit the topics discovered from that dataset as additional knowledge in order to measure similarity between snippets. Since there are not many linguistic resources (Wordnet, Ontology, linguistic processing toolkits, etc.) in Vietnamese (and languages other than English), this framework is an economic and effective solution to the problem of Web search clustering and labeling in Vietnamese (and other Asian languages).

—Easy to reuse: The remarkable point of this framework is the hidden topic analysis of a large collection. This is a totally unsupervised process but still takes time for estimation. However, once estimated, the topic model can be applied to more than one task which is not only clustering and labeling but also classification, contextual matching, etc.

Also, the framework is general enough to be applied to many clustering methods. In this article, we performed a careful evaluation for clustering search results in Vietnamese with the universal dataset containing several hundred megabytes of Wikipedia and VnExpress Web pages and achieved impressive clustering and labeling quality.

The rest of the article is organized as follows. Section 2 summarizes some related studies. Section 3 proposes the general framework for clustering and labeling with hidden topics. Section 4 reviews some of the hidden topic analysis models in which we focus on LDA. Section 5 describes steps for analyzing topics for a universal dataset in Vietnamese. Sample topics and remarks for these datasets are also presented in this section. Section 6 gives more technical details about how to cluster and label Web search results with hidden topics. Section 7 carefully presents our experimental results and the result analysis. Finally, some conclusions are given in Section 8.

2. RELATED WORK

Document clustering in general and Web search results clustering in particular have become an active research topic during the past decade. Based on the relationship between clustering and labeling, we can classify solutions to the problem of Web snippet clustering and labeling into two approaches: (1) perform snippets clustering and then labeling the generated clusters; or

(2) generate significant phrases each of which is a cluster representative, snippets are then clustered based on these cluster representatives. In the following, we will present our survey on the approaches to snippets clustering and labeling as well as the methods to deal with short texts, which is also one major part in our proposal.

2.1 Finding Clusters First

Chen and Dumais [2001] developed a user interface that organizes Web search results into hierarchical categories. To do that, they built a system that achieves the Web pages returned by a search engine and classifies them into a known hierarchical structure such as LookSmart's Web directory. Labels of the categories in the hierarchy are then used as labels of the clusters. Cutting et al. [1992], on the other hand, considered clustering as a document browsing technique. A large corpus is partitioned into clusters associated with their summaries which are frequent words in clusters. Based on the summaries, users navigate through the clusters of interest. These clusters are gathered together to form a subcollection of the corpus. This subcollection is then scattered on-the-fly into smaller clusters. The process of merging and reclustering based on user navigation continues until the generated clusters become small enough. The most detailed (latest) clusters are represented by enumerating individual documents. The system built by Zamir and Etzioni [1999] was the first post-retrieval system, which is designed especially for clustering Web search results. The authors used novel Suffix Tree Clustering (STC) algorithm to group together documents sharing phrases (ordered sequence of words). This algorithm made use of special data structure called suffix tree—a kind of inverted index of phrases for a document collection. Using the constructed suffix tree, "base clusters" are created, each of which is associated with a phrase indexed in the tree. Base clusters with a high degree of overlapping (in their document sets) are combined to generate final clusters. Shared phrases, which appear in many documents of one cluster, are used to convey the content of the documents in that cluster. According to the authors, the advantage of this approach is the ability to obtain overlapping clusters in which a document can occur in more than one cluster. Chi-Lang Ngo used a method based on K-means and Tolerance Rough Set Model to generate overlapping clusters [Ngo 2003]. They then generated cluster labels by adapting an algorithm for n-gram generation to extract phrases from the contents of each cluster. They also hypothesized that phrases which are relatively infrequent in the whole collection but occurs frequently in clusters will be a good candidate for cluster label. Unfortunately, they did not explain how to formalize this hypothesis in practice. Recently, Geraci et al. [2006] performed clustering by means of a fast version of the furthest-point-first algorithm for metric k-center clustering. Cluster labels were obtained by combining intra-cluster and inter-cluster term extraction based on a variant of the information gain measure.

Supposing that clusters are somehow available, several researchers aimed to assigning labels to these clusters. Given document clusters in hierarchy, Popescul and Ungar [2000] presented two methods of labeling document

clusters. The first one is to use a χ^2 test of significance to detect different word usage across categories in the hierarchy. The second method selects words which both occur frequently in a cluster and effectively discriminate the given cluster from the other clusters. Treeratpituk and Callan [2006] labeled document hierarchy by exploiting a simple linear model to combine a phrase's features into a DScore. They used features such as DF (document frequency), TFIDF (term frequency, inverted document frequency), ranking of DF, the difference of these features at the parent and child node, and so on. The coefficients in the DScore model were learned and evaluated using DMOZ.[4]

2.2 Finding Labels First

The second approach to the problem of Web search results clustering is from the idea of finding cluster description first. Vivisimo is one of most successfully commercial clustering engine on the Web. Although most of the algorithm is kept unknown, their main idea is "rather than form clusters and then figure out how to describe them, we only form well-described clusters in the first place." Toward this trend, Osinski [2003] tried to find out labels by a three-phase process: (1) extract the most frequent terms (words and phrases), (2) use Latent Semantic Indexing (LSI) [Deerwester et al. 1990] to approximate term-document matrix, forming concept-document matrix, and (3) select labels for each concept by matching previously extracted terms that are closest to a concept by standard cosine measure. Each concept becomes a cluster in their system; they later used Vector Space Model to determine snippets in clusters and merge clusters by calculating cluster scores. Zeng et al. [2004], on the other hand, extracted and ranked "salient phrases" as labels by using a regression model learned from human labeled training data. The documents were assigned to relevant salient phrases to form cluster candidates, the final clusters were generated by merging these cluster candidates. Ferragina and Gulli [2005] selected (gaped) sentences by a merging and ranking process. This process begins with words, then merges words in the same snippet and within a proximity window into a (longer) gaped sentence. Selected sentences are ranked and the low ranked sentences are discarded. All sentences which have not been discarded are merged with words in the similar manner. The process is repeated until no merge is possible or sentences are formed by eight words (this can be customizable). The results of this process are sentences which form labels for "leaf clusters." These leaf clusters are then merged to achieve higher level clusters based on the sharing of "gaped sentences."

2.3 Dealing with Short Texts

Enriching short texts like snippets has achieved a lot of attentions recently. Banerjee et al. [2007] queried Wikipedia indexed collection for each snippet. They then achieved titles of top Wikipedia pages as additional features for that snippet. Bollegala et al. [2007] proposed a robust semantic similarity measure that uses the information available on the Web to measure similar-

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

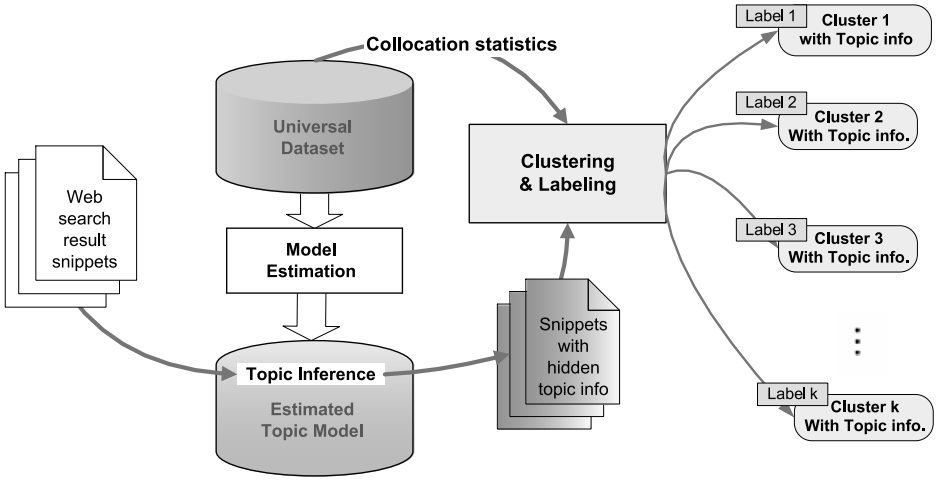
ity between words or entities (Web search results). Not only based on the co-occurance of words in top ranked search results, they also extracted linguistic patterns to measure word semantic similarity. Cai and Hofmann [2003] automatically extracted concepts from a large collection of text using pLSA. They then exploited these concepts for classification with AdaBoost, a boosting technique which combines several weak, moderately accurate classifiers into one highly accurate classifier. Chi-Lang Ngo [2003] provided an enriched representation by exploiting the Tolerance Rough Set Model (TRSM). With TRSM, a document is associated with a set of tolerance classes. In this context, a tolerance class represents a concept that is characterized by terms it contains. For example, {jaguar, OS, X} and {jaguar, cars} are two tolerance classes discovered from the collection of search results returned by Google for the query "jaguar." Ferragina and Gulli [2005] used two databases to improve extracted cluster labels. The first one is an indexed collection of anchor texts extracted from more than 200 millions Web pages. This knowledge base is used to enrich the content of the corresponding (poor) snippets. The second knowledge base is a ranking engine over the Web directory DMOZ[5] which is freely available, controlled by humans and thus of high quality. The fundamental disadvantage of this method when applying to another languages other than English is the requirement of the human-built knowledge base (DMOZ). Recent research [Hu et al. 2008] used a concept thesaurus extracted from Wikipedia to enrich snippets in order to improve clustering performance.

3. GENERAL FRAMEWORK

In this section, we present the proposed framework that aims at building a clustering system with hidden topics from large-scale data collections. The framework is depicted in Figure 1 and consists of six major steps.

Among the six steps, choosing a right universal dataset (a) is probably the most important one. The universal dataset, as its name suggests, must be large and rich enough to cover a lot of words, concepts, and topics that are relevant to the domain of application. Moreover, the vocabulary of the dataset should be consistent with future unseen data that we will deal with. The universal dataset, however, is not necessary in a fine structure like Wikipedia in English or DMOZ. This implies the flexibility of the external data collection in use as well as of our framework. The dataset should also be preprocessed to exclude noise and nonrelevant words, so phase (b) can achieve good results. More details of (a) and (b) steps for a specific collection in Vietnamese will be discussed in the Section 5. Along with performing topic analysis, we also exploit the dataset to find collocations (c) (see Section 6.3.1). The collocations are then used for labeling clusters in (f). One noticeable point is that (a), (b), and (c) are performed offline and with no supervisor. The estimated model can

[5]http://www.dmoz.org



(a) Choosing an appropriate "universal dataset."
(b) Performing topic analysis for the universal dataset.
(c) Finding collocations in the universal dataset.
(d) Performing topic inference for search snippets.
(e) Combining the original snippets with their hidden topics.
(f) Building a clustering/labeling system on the enriched snippets.

Fig. 1. The general framework of clustering Web search results with hidden topics.

be reused as a knowledge base to enrich documents for another tasks such as classification [Phan et al. 2008]. As a result, topic analysis is an economic, extensible, and reusable solution to enrich documents in text/Web mining.

In general, topic analysis for the universal dataset (b) can be performed by using one of the well-known hidden topic analysis models such as pLSA, LDA, DTM, and CTM. It is worthy to notice that there is a tradeoff between the richness of topic information and the time complexity of the system. LDA is chosen in this research because it is a more completely generative model than pLSA but not so complicated. With LDA, we are able to capture important semantic relationships in textual data but keeping time overhead acceptable. More details about topic analysis and LDA will be given in the Section 4.

The result of the step (b) is an estimated topic model including hidden topics and probability distributions of words given those topics (in the case of LDA). Based on this model and a collection of search results, we can perform topic inference (d) for those search snippets. Note that these short, sparse snippets are performed topic inference based on the model of the Universal Dataset, which has already been analyzed and converged. In another words, once the topics has been estimated in a huge dataset, they can be used as a background knowledge for adding more semantic to these search snippets. For each snippet, the output of (d) is the distribution of hidden topics in which high probabilities are assigned to its related topics. For instance, a snippet for the query "ma trận"

(*matrix*) is probably related to topics such as "mathematic" or "movie." How to use this information as rich and useful features for clustering and labeling (e) and (f) depends on the clustering algorithm.

This framework does not confine us to any clustering/labeling approaches. In this research, for simplicity, we applied the "find clusters first" approach and used HAC for the clustering step (see Section 6). However, other method such as K-means can be used for clustering. For K-means, we are able to choose initial centroids as snippets with emerging topics in the collection instead of random selection. Moreover, we can use the "find cluster descriptions first" approach to clustering and labeling in which the topic information is very helpful to achieve "topic-oriented (significant) phrases."

4. HIDDEN TOPIC ANALYSIS MODELS

Representing text corpora effectively to exploit their inherent essential relationship between members of the collections has become sophisticated over the years. Latent Semantic Analysis (LSA) [Deerwester et al. 1990] is a significant step in this regard. LSA uses a singular value decomposition of the term-by-document X matrix to identify a linear subspace in the space of term weight features that captures most of the variance in the collection. This approach can achieve considerable reduction in large collections and reveal some aspects of basic linguistic notions such as synonymy or polysemy. One drawback of LSA is that the resulting concepts might be difficult to interpret [Wikipedia 2008]. For example, a linear combination of words such as *car* and *truck* could be interpreted as a concept *vehicle*. However, it is possible for the case in which the linear combination of *car* and *bottle* to occur. This leads to results which can be justified on the mathematical level, but which have no interpretable meaning in natural language.

Probabilistic Latent Semantic Analysis (pLSA) [Hofmann 1999] was the successive attempt to capture semantic relationship within text. It relies on the idea that each word in a document is sampled from a mixture model, where mixture components are multinomial random variables that can be viewed as representation of "topics." Consequently, each word is generated from a single topic, and different words in a document may be generated from different topics.

While Hofmann's work is a useful step toward probabilistic text modeling, it suffers from severe overfitting problems [Heinrich 2005]. Additionally, although pLSA is a generative model of the documents in the estimated collection, it is not a generative model of new documents. In another words, it is not clear how to assign probability to a document outside the training set [Blei et al. 2003]. The Latent Dirichlet Allocation (LDA) first introduced by Blei et al. [2003], is the solution to these problems. Since topic inference for new documents (based on an estimated topic model) is an important step in our proposal, LDA is a better choice than pLSA for this framework. Not only theoretical analysis, but also careful experiments have been conducted to prove the advantages of LDA over pLSA in Blei et al. [2003].

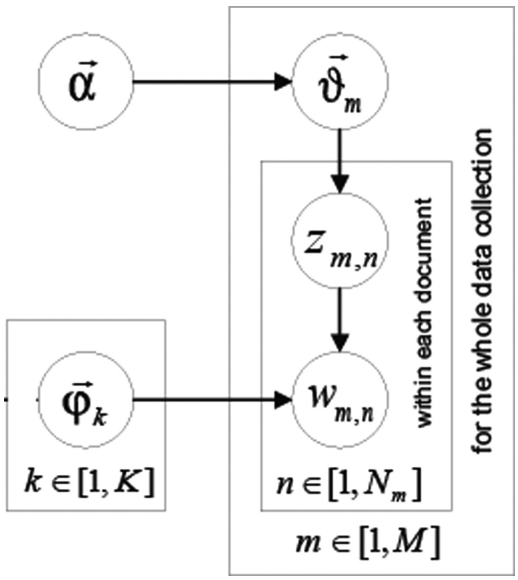


Fig. 2. The generative process of LDA.

There have been some other topic modeling methods proposed recently such as Dynamic Topic Model (DTM) [Blei and Lafferty 2006], Correlated Topic Model (CTM) [Blei and Lafferty 2007], and topical N-gram model [Wang et al. 2007] which can be applied to the process of topic analysis. While still being able to capture rich relationships between topics in a collection, LDA is more simple than these models. For this reason, we choose LDA for the topic analysis step in our proposal. More details about LDA are given in the subsequent sections.

4.1 Latent Dirichlet Allocation (LDA)

LDA [Blei et al. 2003; Heinrich 2005; Phan et al. 2008] is a generative graphical model as shown in Figure 2. It can be used to model and discover underlying topic structures of any kind of discrete data in which text is a typical example. LDA was developed based on an assumption of document generation process depicted in both Figure 2 and Table I. This process can be interpreted as follows.

In LDA, a document $\overrightarrow{w}_m = \{w_{m,n}\}_{n=1}^{N_m}$ is generated by first picking a distribution over topics $\overrightarrow{\vartheta}_m$ from a Dirichlet distribution ($Dir(\overrightarrow{\alpha})$), which determines topic assignment for words in that document. Then the topic assignment for each word placeholder $[m, n]$ is performed by sampling a particular topic $z_{m,n}$ from multinomial distribution $Mult(\overrightarrow{\vartheta}_m)$. And finally, a particular word $w_{m,n}$ is generated for the word placeholder $[m, n]$ by sampling from multinomial distribution $Mult(\overrightarrow{\varphi}_{z_{m,n}})$.

Table I. Generation Process for LDA

for all documents $m \in [1, M]$ **do**
  sample mixture proportion $\overrightarrow{\vartheta}_m \sim Dir(\overrightarrow{\alpha})$
  sample document length $N_m \sim Poiss(\xi)$
  for all words $n \in [1, N_m]$ **do**
    sample topic index $z_{m,n} \sim Mult(\overrightarrow{\vartheta}_m)$
    sample term for word $w_{m,n} \sim Mult(\overrightarrow{\varphi}_{z_{m,n}})$
  end for
end for

Parameters and variables:
- M: the total number of documents to generate (const scalar)
- K: the number of (hidden/latent) topics /mixture components (const scalar)
- V: number of terms t in vocabulary (const scalar)
- $\overrightarrow{\alpha}$: Dirichlet parameters
- $\overrightarrow{\vartheta}_m$: topic distribution for document m
- $\Theta = \{\overrightarrow{\vartheta}_m\}_{m=1}^{M}$: a $M \times K$ matrix
- $\overrightarrow{\varphi}_k$: word distribution for topic k
- $\Phi = \{\overrightarrow{\varphi}_k\}_{k=1}^{K}$: a $K \times V$ matrix
- N_m: the length of document m, here modeled with a Possion distribution with constant parameter ξ
- $z_{m,n}$: topic index of nth word in document m
- $w_{m,n}$: a particular word for word placeholder [m, n]

From the generative graphical model depicted in Figure 2, we can write the joint distribution of all known and hidden variables given the Dirichlet parameters as follows.

$$p\left(\overrightarrow{w}_m, \overrightarrow{z}_m, \overrightarrow{\vartheta}_m | \overrightarrow{\alpha}, \Phi\right) = \prod_{n=1}^{N_m} p\left(w_{m,n} | \overrightarrow{\varphi}_{z_{m,n}}\right) p\left(z_{m,n} | \overrightarrow{\vartheta}_m\right) p\left(\overrightarrow{\vartheta}_m | \overrightarrow{\alpha}\right)$$

And the likelihood of a document $\overrightarrow{w}_m$ is obtained by integrating over $\overrightarrow{\vartheta}_m$ and summing over $\overrightarrow{z}_m$ as follows.

$$p\left(\overrightarrow{w}_m | \overrightarrow{\alpha}, \Phi\right) = \int p\left(\overrightarrow{\vartheta}_m | \overrightarrow{\alpha}\right) \cdot \prod_{n=1}^{N_m} p\left(w_{m,n} | \overrightarrow{\vartheta}_m, \Phi\right) d\overrightarrow{\vartheta}_m$$

Finally, the likelihood of the whole data collection $\mathcal{W} = \{\overrightarrow{w}_m\}_{m=1}^{M}$ is product of the likelihoods of all documents:

$$p\left(\mathcal{W} | \overrightarrow{\alpha}, \Phi\right) = \prod_{m=1}^{M} p\left(\overrightarrow{w}_m | \overrightarrow{\alpha}, \Phi\right) \quad (1)$$

4.2 LDA Estimation with Gibbs Sampling

Parameter estimation for LDA by directly and exactly maximizing the likelihood of the whole data collection in Equation (1) is intractable. One solution is to use approximate estimation methods such as Variational Methods [Blei et al. 2003] and Gibbs Sampling [Griffiths and Steyvers 2004]. Gibbs Sampling is a special case of Markov-chain Monte Carlo (MCMC) [Andrieu

et al. 2003] and often yields relatively simple algorithms for approximate inference in high-dimensional models such as LDA.

Let $\overrightarrow{w}$ and $\overrightarrow{z}$ be the vectors of all words and their topic assignment of the whole data collection W. Gibbs Sampling approach [Griffiths and Steyvers 2004] is not explicitly representing Φ or ϑ as parameters to be estimated, but instead considering the posterior distribution over the assignments of words to topics, $P(\overrightarrow{z}\,|\,\overrightarrow{w})$. We then obtain estimates of Φ and Θ by using this posterior distribution. In order to estimate the posterior distribution, Griffiths et al. used the probability model for LDA with the addition of a Dirichlet prior on Φ. The complete probability model is as follows:

$$\begin{gathered} w_i|z_i,\, \Phi^{(z_i)} \sim \text{Mult}(\Phi^{(z_i)}) \\ \Phi \sim \text{Dirichlet}(\beta) \\ z_i|\Theta^{(d_i)} \sim \text{Mult}(\Theta^{d_i}) \\ \Theta^{(d_i)} \sim \text{Dirichlet}(\alpha) \end{gathered}$$

Here, α and β are hyper-parameters, specifying the nature of the priors on Θ and Φ. These hyperparameters could be vector-valued or scalar. The joint distribution of all variables given these parameters is $p(\overrightarrow{w}, \overrightarrow{z}, \Theta, \Phi|\alpha, \beta)$. Because these priors are conjugate to the multinominal distributions Φ and Θ, we are able to compute the joint distribution $p(\overrightarrow{w}, \overrightarrow{z})$ by integrating out Φ and Θ.

Using this generative model, the topic assignment for a particular word can be calculated based on the current topic assignment of all the other word positions. More specifically, the topic assignment of a particular word t is sampled from the following multinomial distribution.

$$p(z_i = k|\overrightarrow{z}_{\neg i}, \overrightarrow{w}) = \frac{n^{(t)}_{k,\neg i} + \beta_t}{\left[\sum_{v=1}^{V} n^{(v)}_k + \beta_v\right] - 1} \frac{n^{(k)}_{m,\neg i} + \alpha_k}{\left[\sum_{j=1}^{K} n^{(j)}_m + \alpha_j\right] - 1}, \tag{2}$$

where $n^{(t)}_{k,\neg i}$ is the number of times the word t is assigned to topic k except the current assignment; $\sum_{v=1}^{V} n^{(v)}_k - 1$ is the total number of words assigned to topic k except the current assignment; $n^{(k)}_{m,\neg i}$ is the number of words in document m assigned to topic k except the current assignment; and $\sum_{j=1}^{K} n^{(j)}_m - 1$ is the total number of words in document m except the current word t. In normal cases, Dirichlet parameters $\overrightarrow{\alpha}$, and $\overrightarrow{\beta}$ are symmetric, that is, all α_k ($k = 1..K$) are the same, and similarly for β_v ($v = 1..V$).

After finishing Gibbs Sampling, two matrices Φ and Θ are computed as follows.

$$\varphi_{k,t} = \frac{n^{(t)}_k + \beta_t}{\sum_{v=1}^{V} n^{(v)}_k + \beta_v} \tag{3}$$

$$\vartheta_{m,k} = \frac{n^{(k)}_m + \alpha_k}{\sum_{j=1}^{K} n^{(j)}_m + \alpha_j} \tag{4}$$

Fig. 3. Pipeline for data preprocessing and transformation.

4.3 LDA Inference with Gibbs Sampling

Given an estimated LDA model, we can now perform topic inference for unknown documents by a similar sampling procedure as previously [Heinrich 2005]. A new document $\tilde{m}$ is a vector of words $\tilde{\vec{w}}_m$; our goal is to estimate the posteria distribution of topics $\tilde{\vec{z}}$ given the word vector $\tilde{\vec{w}}$ and the LDA model $L(\Theta, \Phi)$: $p(\vec{z}|\vec{w}, L) = p(\tilde{\vec{z}}, \tilde{\vec{w}}, \vec{w}, \vec{z})$. Here, $\vec{w}$ and $\vec{z}$ are vectors of all words and topic assignment of the data collection upon which we estimate the LDA model. The similar reasoning is made to get the Gibbs sampling update as follows:

$$p(\tilde{z}_i = k|\tilde{\vec{z}}_{\neg i}, \tilde{\vec{w}}; \vec{z}_{\neg i}, \vec{w}) = \frac{n_k^{(t)} + \tilde{n}_{k,\neg i}^{(t)} + \beta_t}{\left[\sum_{v=1}^{V} n_k^{(v)} + \tilde{n}_k^{(v)} + \beta_v\right] - 1} \frac{n_{\tilde{m},\neg i}^{(k)} + \alpha_k}{\left[\sum_{z=1}^{K} n_{\tilde{m}}^{(z)} + \alpha_z\right] - 1}, \tag{5}$$

where the new variable $\tilde{n}_k^{(t)}$ counts the observation of term t and topic k in new documents. This equation gives an illustrative example of how Gibbs sampling works: high estimated word-topic association $n_k^{(t)}$ will dominate the multinomial masses in comparison with the contributions of $\tilde{n}_k^{(t)}$ and $n_{\tilde{m}}^{(t)}$, the masses of topic-word associations are propagated into document-topic associations [Heinrich 2005].

After performing topic sampling, the topic distribution of new document $\tilde{m}$ is $\vec{\vartheta}_{\tilde{m}} = \left\{\vartheta_{\tilde{m},1}, ..., \vartheta_{\tilde{m},k}, ..., \vartheta_{\tilde{m},K}\right\}$ where each component is calculated as follows:

$$\vartheta_{\tilde{m},k} = \frac{n_{\tilde{m}}^{(k)} + \alpha_k}{\sum_{z=1}^{K} n_{\tilde{m}}^{(z)} + \alpha_z}. \tag{6}$$

5. HIDDEN TOPIC ANALYSIS OF VIETNAMESE DATASET

5.1 Preprocessing and Transformation

Data preprocessing and transformation are necessary for data mining in general and for hidden topic analysis in particular. Since we target at topic analysis for Vietnamese, it is necessary to perform preprocessing in the consideration of specific characteristics of this language. The main steps for our preprocessing and transformation are described in the following and summarized in Figure 3.

5.1.1 *Segmentation and Tokenization.* This step includes sentence segmentation, sentence tokenization, and word segmentation.

Sentence segmentation is to determine whether a "sentence delimiter" is really a sentence boundary. Like English, sentence delimiters in Vietnamese are full-stop, the exclamation mark and the question mark (.!?). The exclamation mark and the question mark do not really pose the problems. The critical

element is the period: (1) the period can be a sentence-ending character (full stop); (2) the period can denote an abbreviation; (3) the period can be used in some expressions such as URL, e-mail, numbers, etc.; (4) in some cases, a period can assume both (1) and (2) functions. Given an input string, the results are sentences separated in different lines.

Sentence tokenization is the process of detaching marks from words in a sentence. For example, we would like to detach "," or ":" from the previous words, which they are attached to.

Word segmentation. There is no clear word boundaries in Vietnamese since words are written in several syllables separated by white space (thus, we do not know which white space is actual word boundary and which is not). This leads to the task of word segmentation, that is, segment a sentence into a sequence of words. Vietnamese word segmentation is a prerequisite for any further processing and text mining. Though being quite basic, it is not a trivial task because of the following ambiguities:

—Overlapping ambiguity: String abc is called overlapping ambiguity when both ab and bc are valid Vietnamese words. For example: "học sinh học sinh học" (Student studies biology) → "học sinh" (student) and "sinh học" (biology) are found in the Vietnamese dictionary.
—Combination ambiguity: String ab were called combination ambiguity when a, b or ab are possible choices. For instance: "bàn là một dụng cụ" (Table is a tool) → "bàn" (Table), "bàn là" (iron), "là" (is) are found in the Vietnamese dictionary.

For word segmentation, we used Conditional Random Fields approach to segment Vietnamese words [Nguyen et al. 2006] in which F1 measure is reported to be about 94%. After this step, sequences of syllables are joined to form words. For examples, a string like "công nghệ và cuộc sống" will become "công_nghệ và cuộc_sống" (*technology and life*).

5.1.2 *Filters and Nontopic-Oriented Word Removal.* After word segmentation, tokens, which can be word tokens, number tokens and so on, now are separated by white space. Filters remove trivial tokens such as tokens for number, date/time, too-short tokens (of which length is less than two characters). Too short sentences, English sentences, or Vietnamese sentences without tones (The Vietnamese sometimes write Vietnamese text without tone) also should be filtered or manipulated in this phase.

Nontopic-oriented words are those we consider to be trivial for the topic analyzing process. These words can cause much noise and negative effects for our analysis. Here, we consider functional words, too rare or too common words as nontopic-oriented words. The typical categories of functional words in Vietnamese includes classifier noun (similar to articles in English), conjunction (similar to *and*, *or* in English), numeral, pronoun, adjunct, and so on.

Table II. Statistics of the Universal Dataset

The universal dataset
After removing HTML tags, duplicate, too short or navigating pages, doing sentence and word segmentation: **size** ≈ 480M; |**docs**| ≈ 69,371 After filtering and removing non-topic oriented words: **size** ≈ 101M, |**docs**| = 57,691 |**words**| = 10,296,286; |**vocabulary**| = 164,842
Topics assigned by humans in VnExpress Dataset
Society: Education, Entrance Examinations, Lifestyle of Youths **International**: Analysis, Files, Lifestyles **Business**: Business man, Stock, Integration **Culture**: Music, Fashion, Stage, Cinema **Sport**: Football, Tennis **Life**: Family, Health **Science**: New Techniques, Natural Life, Psychology and **Others** ...
Topics assigned by humans in Wikipedia Dataset
Mathematics and Natural Science: geology, zoology, chemistry, meteorology, biology, astronomy, mathematics, physics, etc. **Technologies and Applied Science**: Nano technologies, biologic technology, information technology, Internet, computer science,etc. **Social Science and Philosophy**: economics, education, archaeology, agriculture, anthropology, sociology, etc. **Culture & Arts**: Music, tourism, movie industry, stage, literature, sports, etc. **Religion & Belief**: Hinduism, muslim, buddhism, confucianism, atheistic, etc.

5.2 The Universal Dataset

Choosing a universal dataset is an important step in our proposal. In order to cover many useful topics, we used Nutch[6] to collect Web pages from two huge resources in Vietnamese, which are Vnexpress[7] and Wikipedia.[8] VnExpress is one of the highest ranking e-newspapers in Vietnam, thus containing a large number of articles in many topics in daily life ranging from science, society and business, and many more. Vietnamese Wikipedia, on the other hand, is a huge online encyclopedia and contains thousands of articles which are either translated from English Wikipedia or written by Vietnamese contributors. Although Vietnamese Wikipedia is smaller than the English version, it contains useful articles in many academic domains such as mathematics, physics, etc. We combined two collections to form the universal dataset. The statistic information of the two collections is given in Table II. Note that topics listed here are just for reference and not to be taken into the topic analysis process.

5.3 Analysis Results and Outputs

After data preprocessing and transformation, we obtained 101MB data. We performed topic analysis for this processed dataset using GibbsLDA++.[9] The parameters *Alpha* and *Beta* were set at $50/K$ and 0.1 respectively where K is

[6]http://lucence.apache.org/nutch/
[7]http://vnexpress.net
[8]http://vi.wikipedia.org
[9]http://gibbslda.sourceforge.net

Topic 3	Topic 4	Topic 7	Topic 9	Topic 10	Topic 15
hàm *(function)* **không gian** *(space)* **toán học** *(mathematics)* **định nghĩa** *(definition)* **phần tử** *(elements)* **bài toán** *(problem)* **lý thuyết** *(theory)* **tính toán** *(calculation)* **xác định** *(definite)* **định lý** *(theorem)* **phương trình** *(equation)* **ánh xạ** *(mapping)* **đại số** *(algebra)*	**phần mềm** *(software)* **chương trình** *(programs)* **Windows**$^{+}$ *(Windows)* **phiên bản** *(version)* **Microsoft**$^{+}$ *(Microsoft)* **hệ điều hành** *(operating system)* **ứng dụng** *(applications)* **cài đặt** *(install)* **giao diện** *(interface)* **trình duyệt** *(browser)* **internet*** *(internet)* **server*** *(server)*	**cầu thủ** *(football player)* **HLV**$^{\times}$ *(couch)* **đội bóng** *(football team)* **trận đấu** *(match)* **tiền đạo** *(offensive player)* **bàn thắng** *(goal)* **hậu vệ** *(defensive player)* **thủ môn** *(goalkeeper)* **chấn thương** *(injury)* **trọng tài** *(referee)* **đội hình** *(line-up)* **SLNA**$^{\times}$ *(SLNA*	**máy bay** *(aircraft)* **sân bay** *(airport)* **hàng không** *(airline)* **giao thông** *(traffic)* **tai nạn** *(accident)* **chuyến bay** *(flight)* **quốc tế** *(international)* **khách hàng** *(customer)* **Boeing**$^{+}$ **vận chuyển** *(deliver)* **phương tiện** *(vehicle)* **vận tải** *(transportation)* **đường sắt** *(railway)* **nhà ga** *(station)*	**tác giả** *(author)* **sách** *(book)* **nhà văn** *(writer)* **văn học**$^{\#}$ *(literature)* **truyện** *(stories)* **thơ** *(poem)* **tiểu thuyết** *(novel)* **xuất bản**$^{\#}$ *(publish)* **nhà thơ** *(poet)* **độc giả** *(readers)* **văn chương**$^{\#}$ *(literature)* **nhà xuất bản** *(publisher)* **phát hành**$^{\#}$ *(publish)*	**quốc hội** *(congress)* **tổng thống** *(president)* **dân chủ** *(democratic)* **hội đồng** *(council)* **chính quyền** *(goverment)* **nhân dân** *(people)* **cộng hòa** *(republican)* **nhà nước** *(state)* **hiến pháp** *(constitution)* **lãnh đạo** *(leadership)* **bầu cử** *(election)* **hội nghị** *(meeting)* **đảng cộng sản** *(communist party)*

Fig. 4. Most likely words of some sample topics analyzed from the universal dataset ($K = 60$).

the number of topics. The results of topic analysis with $K = 60$ and $K = 120$ are shown in Figure 4 and Figure 5. The complete results can be viewed online.[10]

Figure 4 and 5 indicate that hidden topic analysis can model some linguistic phenomena such as synonyms or acronyms. For instance, the synonyms "văn học" (*literature*) and "văn chu'o'ng" (*literature*) (Figure 4) are connected by the topic 10. The acronyms such as HLV (Huấn Luyện Viên - *couch*) and SLNA (Sông Lam Nghệ An—name of a famous football club) (Figure 4) were correctly put in the topic of football (topic 7). Furthermore, hidden topic analysis is an economic solution to capture the semantic of new words (foreign words, named entities). For example, words such as "windows", "microsoft", "internet" or "server" (Figure 4), which are not covered by general Vietnamese dictionaries, were specified precisely in the domain of computer (topic 4). Figure 5 demonstrates another interesting situation in which the gap between two ways of writing the word *painter* in Vietnamese ("họa sĩ"—the correct spelling—and "họa sỹ"—the informal spelling but commonly accepted) were bridged by the topic about "painting, art" (topic 82). We will demonstrate how these relationships between words (via topics) can be used to provide good clustering in Section 7.

[10]http://jgibblda.sourceforge.net/vnwiki-120topics.txt

Topic 0	Topic 5	Topic 6	Topic 9	Topic 12	Topic 82
triết học *(phylosophy)* **khái niệm** *(concept)* **nhận thức** (*conceive*) **tri thức** *(knowledge)* **tồn tại** *(existence)* **học thuyết** *(doctrine)* **quan niệm** *(idea)* **bản chất** *(essence)* **quy luật** *(law)* **lý luận** *(reasoning)* **trường phái** *(school of thought)* **tranh luận** *(argument)*	**năng lượng** *(energy)* **điện** *(electricity)* **sóng** *(wave)* **ánh sáng** *(light)* **điện từ** *(electro-magnetic)* **dòng điện** *(electric current)* **vật liệu** *(material)* **tần số** *(frequency)* **tia** *(ray)* **từ trường** *(magnetic field)* **tín hiệu** *(signal)* **công suất** *(electricity power)*	**nhạc** *(music)* **album** *(album)* **ca sĩ** *(singer)* **ban nhạc** *(band)* **Pop** *(pop)* **Madonna** *(Madonna)* **âm nhạc** *(music)* **biểu diễn** *(performance)* **bảng xếp hạng** *(musical chart)* **ca khúc** *(song)* **lưu diễn** *(concert tour)* **Grammy** *(Grammy)* **MTV** *(MTV)* **ghi âm** *(recording)*	**cảnh sát** *(police)* **chết** *(dead)* **nạn nhân** *(victim)* **tai nạn** *(accident)* **phát hiện** *(discovery)* **xác** *(dead body)* **điều tra** *(investment)* **thi thể** *(dead body)* **bắt cóc** *(kidnap)* **súng** *(gun)* **an ninh** *(security)* **bị bắt** *(catch)* **mất tích** *(missing)*	**nguyên tố** *(elements)* **kim loại** *(metal)* **vật liệu** *(material)* **nhiệt độ** *(temperature)* **nhôm** *(aluminium)* **hợp chất** *(compound)* **hóa học** *(chemical)* **tinh thể** *(crystal)* **thép** *(steel)* **ô-xít** *(oxide)* **thủy tinh** *(glass)* **cấu trúc** *(structure)* **hợp kim** *(alloy)*	**nghệ thuật** *(art)* **tranh** *(picture)* **triển lãm** *(exhibit)* **họa sĩ**[#] *(painter)* **mỹ thuật** *(art)* **bảo tàng** *(museum)* **nghệ sĩ** *(artist)* **nhiếp ảnh** *(photograph)* **chân dung** *(portrait)* **hội họa** *(painting)* **bức ảnh** *(image)* **chất liệu** *(material)* **họa sỹ**[#] *(painter)*

Fig. 5. Most likely words of some sample topics analyzed from the universal dataset (K = 120).

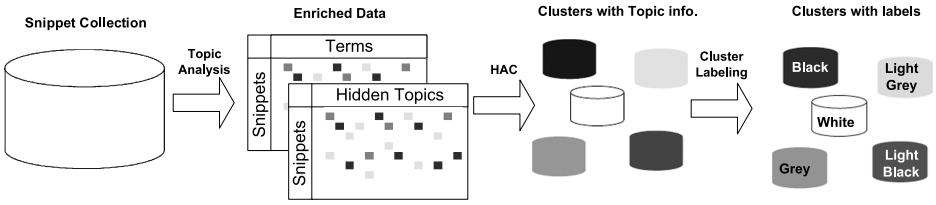


Fig. 6. Clustering and labeling with hidden topics.

6. CLUSTERING AND LABELING WITH HIDDEN TOPICS

Clustering and labeling with Hidden Topics is summarized in Figure 6. Based on the estimated LDA model of the universal dataset (see Section 5), the collection of snippets is cleaned and performed topic analysis (see Section 4.3). This provides an enriched representation of the snippets. A specific clustering method is then applied on the enriched data. Here, we use Hierarchical Agglomerative Clustering (HAC) for the clustering phase. The generated clusters are shifted to the "Cluster Labeling Assignment" step which assigns descriptive labels to these clusters.

6.1 Topic Analysis and Similarity

Similarity between two snippets is fundamental to measure similarity between clusters. This section describes our representation of snippets with hidden topic information, which are inferred based on the topic model of the universal dataset, and presents a method to measure similarity between snippets.

For each snippet d_i, after topic analysis, we obtain the topic distribution $\vec{\vartheta}_{d_i} = \left\{\vartheta_{d_i,1}, ..., \vartheta_{d_i,k}, ..., \vartheta_{d_i,K}\right\}$. Upon this, we are able to build the topic vector $\vec{t}(di) = \{t_1, t_2, ..., t_K\}$ in which the weight t_i of the topic i^{th} is determined with regard to its probability $\vartheta(i)$ as follows:

$$t_i = \begin{cases} \vartheta(i) & \text{if } \vartheta(i) \geq cutoff \\ 0 & \text{otherwise} \end{cases} . \tag{7}$$

Note that K is the number of topics, and *cutoff* is the lower bound threshold for a topic to be considered important. Let V be the vocabulary of the snippet collection, the term vector of the snippet d_i has the following form:

$$\vec{w}(d_i) = \{w_1, ..., w_{|V|}\}$$

Here, the element w_i in the vector, which corresponds to the word/term i^{th} in V, is weighted by using some schema such as TF, TFxIDF. In order to calculate the similarity between two snippets d_i and d_j, the cosine measure is used for the topic-vectors as well as the term-vectors of two snippets.

$$sim_{di,dj}(topic - vectors) = \frac{\prod_{k=1}^{K} t_{i,k} \times t_{j,k}}{\sqrt{\sum_{k=1}^{K} t_{i,k}^2}\sqrt{\sum_{k=1}^{K} t_{j,k}^2}}$$

$$sim_{di,dj}(term - vectors) = \frac{\prod_{t=1}^{|V|} w_{i,t} \times w_{j,t}}{\sqrt{\sum_{t=1}^{|V|} w_{i,t}^2}\sqrt{\sum_{t=1}^{|V|} w_{j,t}^2}}$$

Combining two values, we obtain similarity between two snippets as follows:

$$sim(d_i, d_j) = \lambda \times sim(topic - vectors) + (1 - \lambda) \times sim(term - vectors) \tag{8}$$

Here, λ is a mixture constant. If $\lambda = 0$, the similarity is calculated without the support of hidden topics. If $\lambda = 1$, we measure the similarity between topic vectors of the two snippets without concerning words within them.

6.2 Hierarchical Agglomerative Clustering

Hierarchical Agglomerative Clustering [Ngo 2003] begins with each snippet as a separate cluster and merge them into successively larger clusters. Consequently, the algorithm builds a structure called *dendogram*—a tree illustrating the merging process and intermediate clusters. Cutting the tree at a given height will give a clustering at a selected precision.

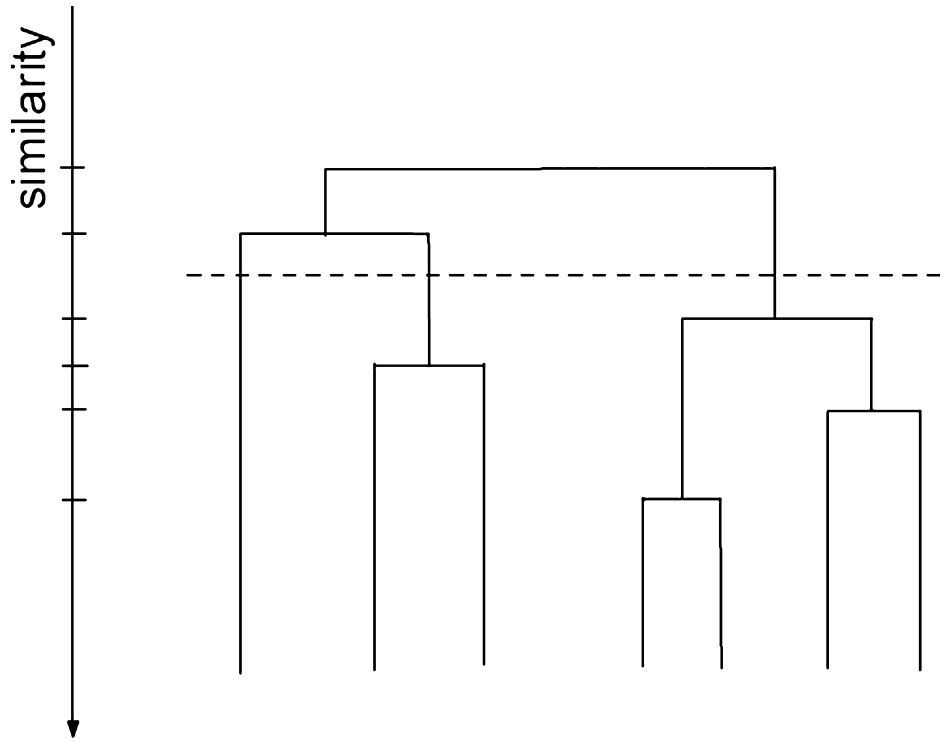


Fig. 7. Dendrogram in Hierarchical Agglomerative Clustering.

Based on similarity between two snippets, similarity between two clusters A & B can be measured as follows:

—The minimum similarity between snippets of each cluster (also called *complete linkage clustering*):

$$\min\{sim(x, y) : x \in A, y \in B\}$$

—The maximum similarity between snippets of each cluster (also called *linkage clustering*):

$$\max\{sim(x, y) : x \in A, y \in B\}$$

—The mean similarity between snippets of each cluster (also called *average linkage clustering*):

$$\frac{1}{|A||B|} \sum_{x \in A} \sum_{y \in B} sim(x, y)$$

6.3 Cluster Label Assignment

Given a set of clusters for a snippet collection, our goal is to generate understandable semantic labels for each cluster. Let $C = \{c_1, c_2, \ldots, c_{|C|}\}$ be a set of

Algorithm 1 Hierarchical Agglomerative Clustering

input: A snippet collection $D = \{d_1, ...d_n\}$, a cluster similarity measure Δ, a **merging threshold** ϵ
output: A set of cluster C

```
C = {initial clusters}            /* each snippet forms an initial cluster */
repeat
    (c1, c2) ← the pair of clusters which are most similar in C
    if Δ (c1, c2) ≥ ε then
        c3 ← c1 ∪ c2
        add c3 into C
        remove c1 and c2 from C
    end
until cannot merge                /* cannot find c1 and c2 with Δ(c1, c2) ≥ ε */
```

$|C|$ clusters, we now state the problem of cluster labeling similarly to the topic labeling problem [Mei et al. 2007] as follows:

—Definition 1: A cluster $c \in C$ in a text collection has a set of close snippets, each cluster is characterized by an *expected topic distribution* ϑ_c, which is the average of topic distributions of all snippets in that cluster.

—Definition 2: A *cluster label* or a *label* l for a cluster $c \in C$ is a sequence of words which are semantically meaningful and best describe the latent meaning of c.

—Definition 3 (Relevance Score): The relevance score of a label l to a cluster c, which is denoted as $s(l, c)$, measures the semantic similarity between the label and the cluster. Given that both l_1 and l_2 are meaningful label candidates, l_1 is a better label for c than l_2 if $s(l_1, c) > s(l_2, c)$

With these definitions, the problem of cluster labeling can be defined as follows: Let $L_i = \{l_{i1}, l_{i2}, ..., l_{im}\}$ be the set of label candidates for the cluster i^{th} in C. Our goal is to rank label candidates and select the most relevant labels for each cluster.

6.3.1 *Label Candidate Generation.* The first step in cluster label assignment is to generate phrases as label candidates. We extract two types of label candidates from the collection of search snippets. The first one includes unigrams (single words except for stop words); and the second one consists of meaningful bigrams (a meaningful phrase of two words—or bigram collocation). While extracting unigrams does not cause many issues, the difficulties lie in meaningful bigram extraction. The problem is how to know a bigram is a meaningful phrase or not. One method is based on hypothesis testing in which we extract phrases from n consecutive words (n-gram) and conduct statistical tests to know whether these words occurs together often than by chance. The null hypothesis usually assumes that "the words in a n-gram are independent," and different statistic testing methods have been proposed to test the significance of violating the null hypothesis. The process of generating label candidates for clusters are summarized in Algorithm 2. Although we only

Algorithm 2 Label Candidate Generation

input: Set of snippets $D = \{d_1, d_2, ..., d_n\}$
Set of clusters $C = \{c_1, ..., c_{|C|}\}$
A frequency threshold **lblThreshold**
An "external collocation list" EC
A collocation threshold **colocThreshold**
output: Label candidates for clusters $LC = \{LC_1, LC_2, ..., LC_{|C|}\}$

```
extract and do statistics for all unigrams and bigrams from D
for each c_i ∈ C do
    LC_i ← θ
    for each unigram u do
        if frequency of u in c_i ≥ lblThreshold then
            if u not a stop-word then LC_i ← LC_i ∪ u
        end
    end
    for each bigram b do
        if frequency of b in c_i ≥ lblThreshold then
            t ← t–score of b in D            /* according to Eqn. 9 */
            if EC contains b or t ≥ colocThreshold then
                LC_i ← LC_i ∪ b
            end
        end
    end
end
```

use n-grams ($n \leq 2$) as label candidates of clusters, the experiments show that this extraction is quite good for Vietnamese due to the fact that Vietnamese word segmentation (see Figure 5) is able to also combine named entities (like "Hồ Chí Minh"–the name of the famous former president in Vietnam) and some other frequently used combination (like "hệ điều hành" (operating system)). Longer phrases can be constructed by concatenating bigrams and unigrams.

A famous hypothesis testing method showing good performance on phrase extraction is Student's T-Test [Manning and Schutze 1999; Banerjee and Pedersen 2003]. Suppose that the sample is drawn from a normal distribution with mean μ, the test considers the difference between the observed and expected means, which are scaled by the variance of the data, and generates the probability of getting a sample of that mean and variance. We then compute the t statistic to specify the probability of getting our sample as follows:

$$t = \frac{x - \mu}{\sqrt{\frac{s^2}{N}}}, \tag{9}$$

where x is the sample mean, s^2 is the sample variance, N is the sample size, and μ is the mean of the distribution. We can reject the null hypothesis if the t statistic is large enough. By looking up the table of the t distribution, we

t-score	C(w^1 w^2)	C(w^1)	C(w^2)	w^1	w^2
31.45	995	2130	1708	Điện thoại (*Phone*)	Di động (*Mobile*)
30.49	992	5223	4664	Thị trường (*Market*)	Chứng khóan (*Stock*)
21.24	469	2854	3713	Công nghệ (*Technology*)	Thông tin (*Information*)
19.1	365	2033	447	Vốn (*Capital*)	Điều lệ (*Charter*)
19.05	363	1278	860	Hội đồng (*Board*)	Quản trị (*Director*)
18.44	340	1492	2434	Đội tuyển (*Team*)	Quốc gia (*National*)
16.88	285	764	972	Vũ khí (*Weapon*)	Hạt nhân (*Nuclear*)
15.49	246	860	4005	Quản trị(*Administration*)	Kinh doanh (*Business*)
15.09	228	560	1021	Hệ điều hành (*OS*)	Windows (*Windows*)
13.82	191	409	1940	Nhà cung cấp (*Supplier*)	Dịch vụ (*Services*)
13.65	204	3432	3094	Trung tâm (*Center*)	Thương mại (*Trade*)
2.65	7	356	349	Khủng hoảng (*Crisis*)	Tiền tệ (*Money*)
2	4	238	407	Ứng cử viên (*Candidate*)	Nghiêm túc (*Serious*)
1.78	5	937	1373	Ủng hộ (*Support*)	Bà (*Her)*
1.73	3	1448	200	Chuẩn bị (*About to*)	Quảng bá (*Advertise*)
1.42	2	658	48	Người sử dụng (*User*)	Tra cứu (*Look up*)
1	1	1040	167	Đặc biệt (*Particularly*)	Yêu mến (*Love*)
1	1	3	2230	Nghiệm thu (*Check*)	Xây dựng (*Construction*)
0	3	5363	379	Chương trình (*Program*)	Cần thiết (*Necessary*)

Fig. 8. Collocations and noncollocations specified from the universal dataset. Here, $C(s)$ is the frequency of the string s in the dataset, and s can be a word or a bigram. The bigrams with t value greater than 2.576 (the confident value of 99.5%) are collocations. All the collocations are extracted into a list called the "external collocation list."

can find out how much confident for us to reject that hypothesis with a predefined threshold. Based on this t test, we now can examine whether a bigram is a collocation or not. Indeed, we find collocations in two situations (using JNSP[11]). The first one is to find collocations (in advance) from the universal dataset. This is performed (offline) to produce what we called the "external collocation list." Examples of collocations and non-collocations drawn from the universal dataset is shown in Figure 8. The second situation is to determine collocations for each snippet collection to be clustered. Extracting collocations from the universal dataset is to obtain common used noun phrases such as "thị_truòng chúng_khoán" (*stock market*) or "điện_thoại di_động" (*mobile phone*) which probably has not enough statistic information in the snippet collection to be verified as a collocation. On the other hand, finding collocations from the snippet collection is able to achieve specific phrases such as named entities which may not occur in the external collection.

6.3.2 *Relevance score.* Given a set of clusters C and their label candidates, we need to measure the relevance between each cluster $c \in C$ and each label candidate l. In this work, we considered the relevance score as a linear combination of some specific features of l, c, and other clusters in C as following

$$relevance(l, c, C) = \sum_{i=1}^{|F|} \alpha_i \times f_i(l, c, C) + \gamma. \quad (10)$$

[11]http://jnsp.sourceforge.net/

Here, α_i and γ are real-value parameters of the relevance score;—F—is the number of features in use, and each feature $f_i(l, c, C)$ is a real-value function of the current label candidate l, current cluster c and the cluster set C. We considered five types of features ($|F| = 5$) for labeling clusters with hidden topics:

—Intra-cluster topic similarity: Topic similarity between the label candidate l and the expected topic distribution of the cluster c (TSIM). If the label candidate l and the cluster c have some common topic with high probability, the two are likely related. We measure TSIM as the cosine of the two topic distribution vectors

$$TSIM(l, c) = \cos(\vartheta_l, \vartheta_c)$$

—Cluster document frequency: Number of snippets in the cluster c containing the phrase l (CDF).

—T-score: The t-score of the phrase l in the snippet collection. If l is a unigram, its TSCORE is assigned to 2 (long phrases are preferred only if they are meaningful phrases).

—Inter-cluster topic similarity: The sum of intra-topic similarity of the label candidate l and other clusters

$$OTSIM(l, c, C) = \sum_{c' \in C, c' \neq c} TSIM(l, c')$$

—Inter-cluster document frequency: The sum of CDF in other clusters

$$OCDF(l, c, C) = \sum_{c' \in C, c' \neq c} CDF(l, c')$$

The label candidates of a cluster are sorted by its relevance in descending order and the most relevant candidates are then chosen as labels for the cluster. The inclusion of topic related features is a remarkable aspect of our proposal in comparison with previous work in cluster labeling (Section 2).

7. EXPERIMENTS

7.1 Experimental Data

We evaluated clustering and labeling with hidden topics on two datasets:

—*Web dataset* consists of *2357* snippets in *9* categories (business, culture and arts, health, laws, politics, science - education, life style and society, sports, technologies). These categories can be used as key clusters for later evaluation. Since this dataset contains the general categories, it can be used for evaluating the overall performance of clustering across domains as well as the quality of topic models (which topic model best describe the categories).

—*Query dataset* includes query collections. We collected this dataset by submitting 20 queries to Google and obtaining about 150 distinguished snippets in key clusters (but ignore minor clusters) for each query (query collection). The search queries are listed in Table III. The reason for choosing these

Table III. Queries Submitted to Google

Types	Query
General Terms	Bảo hiểm (*Insurance*), Công nghệ (*Technology*), Du lịch (*Tourism*), Hàng hóa (*Goods*),Thị tru'ò'ng (*Market*), Triển lãm (*Exhibition*), Đầu tư' (*Investment*), Tài khoản (*Account*), Dân gian (*Folk*), Địa lý (*Geography*), Xây dự'ng (*Construct)*, Tết (*Tet Holiday*)
Ambiguous Terms	Táo (*Apple, Constipation, Kitchen God*), Chuột (*Mouse*), Cử'a sổ (*Windows*), Không gian (*Space*), Ma trận (*Matrix*), Hoa hồng (*Commission, Rose*)
Named Entities	Hồ Chí Minh (*Ho Chi Minh*), Việt Nam (*Vietnam*)

queries is that they are likely to occur in multiple subtopics, so we will benefit more from clustering search results. Since this dataset is sparse, it is much closer to realistic data that the search clustering system need to deal with. We used key clusters in each query collection to evaluate both clustering and labeling with hidden topics.

7.2 Evaluation

7.2.1 *Clustering evaluation.* For evaluation, we need to compare the generated clusters with the key clusters. To do that, we used BCURED scoring method [Bagga and Baldwin 1998], which originally exploited for evaluating entity resolution but also used for clustering evaluation [Bollegala et al. 2007]. This scoring algorithm models the accuracy of the system on a per-document basis and then build a more global score. For a document i, the precision and recall with respect to that document are calculated as follows:

$$P_i = \frac{\text{number of correct documents in the output cluster containing } document_i}{\text{number of documents in the output cluster containing } document_i}$$

$$R_i = \frac{\text{number of correct documents in the output cluster containing } document_i}{\text{number of documents in the key cluster containing } document_i}$$

Here, given a document i, the document j is correct if it is in the same key cluster as the document i. The final precision and recall numbers are computed by the following two formulae:

$$FinalPrecision = \sum_{i=1}^{N} 1/N \times P_i \quad \text{and}$$

$$FinalRecall = \sum_{i=1}^{N} 1/N \times R_i.$$

Usually, precision and recall are not used separately, but combined into F_β measure as following:

$$F_\beta = (1 + \beta^2) \times (precision \times recall)/(\beta^2 \times precision + recall). \tag{11}$$

For clustering evaluation, we used $F_{0.5}$ (or $\beta = 0.5$) to weight precision twice as much as recall. This is because we are willing to have average-size clusters but high precision than merging them into a large cluster for higher recall but low precision (thus, low coherence within clusters).

7.2.2 *Labeling evaluation.* We performed label candidate generation for fixed key clusters in the query dataset. After this step, we had a list of label candidates for each key cluster. We manually assigned "1" to appropriate labels and "0" to inappropriate ones. These scores were used for estimating parameters for the relevance score as well as for evaluation. As mentioned earlier, the label assignment is to rank label candidates for each cluster using relevance score and select the first-rank label. So, we measured the qualityof the relevance score (or the ranking quality) by calculating precision (P) at top N label candidates in the generated ranking list:

$$P@N = \frac{\text{Number of correct label candidates}}{N}. \tag{12}$$

Here, correct label candidates of a given cluster are the ones with the score of "1". In the following experiments, we use P@5, P@10, P@20 for evaluating our labeling method.

7.3 Experimental Settings

We conducted topic analysis for the universal dataset using Latent Dirichllet Allocation with different number of topics (K = 20, 60, 80, 100, 120, 160, 180 topics). The topic models are exploited for experiments hereafter. In the following experiments, we refer to clustering (using HAC) without hidden topics as baseline and clustering (using HAC) with K-topic model (K = 20, 60, etc.) as HT*K*.

The default parameters are specified in Table IV. These default parameters are basically unchanged in our experiments except for lambda which is changed in one specific experiment. The other parameters are changed more often, such as the merging threshold ϵ for clustering (see Algorithm 1), the number of hidden topics (K) for the universal dataset. The parameters of relevance score for labeling, on the other hand, is learned from the query dataset (see Section 7.4.3). By keeping some parameters unchanged and varying others, we measured the influence of the main parameters on the clustering and labeling performance.

7.4 Experimental Results and Analysis

7.4.1 *Clustering performance.* The comparison between baseline and HT*K* (K = 20, 60, 80, etc.) in the Web dataset is demonstrated in Figure 9. Using the categories of the dataset as key clusters, we evaluated clustering performance with precision, recall, and F0.5 as described in the previous section. By taking the maximum value of F0.5 (among different merging thresholds), we compare the performance of baseline and HT*K* (K = 20, 60, 80, etc.) in Figure 9 . As depicted in the figure, clustering with hidden topics in most cases (other than

Table IV. Default Parameters for Clustering and Labeling with Hidden Topics. The parameters are basically set as in the following table. Note that the rare word threshold is set differently for "Web dataset" and "query collections" (in "query dataset." This is because "Web dataset" is much larger than any "query collection" and removing rare words can help to reduce the computational time.

Parameters	Values	Explanation
Clustering Parameters		
Term weighting method	TF	Term Frequency
Lambda	0.35	Mixture constant in the similarity formula between two snippets (Equation 8).
Cluster similarity	Average Linkage	The mean similarity between elements of each clusters.
Frequency threshold	30%	Terms/topics occur more frequent than this rate will be cut off.
Rare threshold	2 or 6	Terms occur less than this threshold will be removed. This threshold is set to 6 for "Web dataset" and to 2 for "query collections."
Topic Cutoff	0.02	Topics with probability less than this value will not be used for enriching snippets.
Labeling Parameters		
Collocation Threshold	2	A bigram with t score calculated in a snippet collection larger than this value is probably used as a label candidate. This is set by looking up the t-score table (for infinite degree of freedom and the confidence value of 97.5%).
Label threshold	2	Phrases with the frequency (in a cluster) less than this value will not be chosen as label candidates for that cluster.

the 20-topic model) improve clustering performance. The bad performance of HT20 (9.74% worse than in the baseline) indicates that the number of topics for analysis should be suitable to reflect the topics in the universal dataset. Once the number of topics is large enough (like larger than 60 topics), the F0.5 is quite stable. It can also be observed that the 100-topic model best describes these general categories. As a result, $K \approx 100$ is probably the suitable number of topics for the universal dataset.

We showed the results of the baseline and clustering using the 100-topic model with lambda of 0.2 (HT100-0.2) in Figure 10(a). From the figure, we can see that HT100-0.2 can provide significant improvement over the baseline. The maximum value of F0.5 in HT100-0.2 is 62.52% which is nearly 16% better than the baseline. When merging threshold is zero, all the snippets are merged into one cluster. That explains why HT100-0.2 and the baseline have the same starting value of F0.5. In addition, the inclusion of hidden topics increases similarity among snippets. As a result, when merging threshold is small, HT100-0.2 does not show an advantage over the baseline. When merging threshold is large enough, on the other hand, we can always obtain better results with HT100-0.2.

In order to evaluate the influence of lambda in clustering performance, we conducted similar experiments to the one in Figure 10(a) but with different lambda (0.2 to 1.0). The maximum values and average values of F0.5 (when

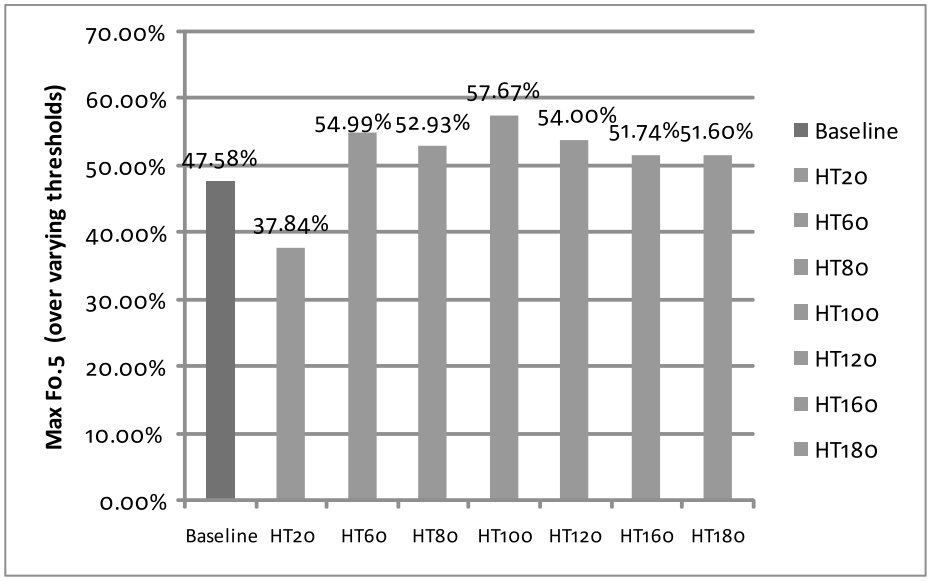
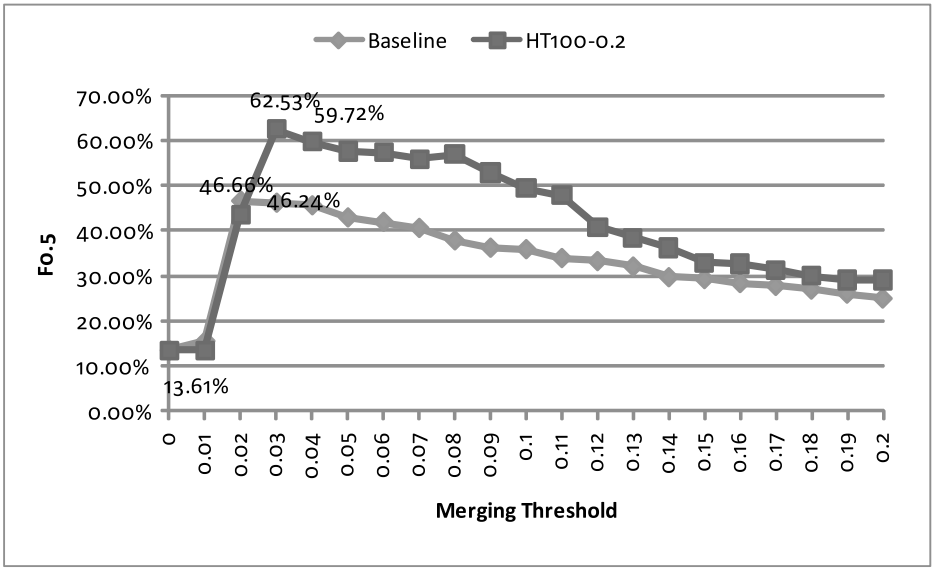


Fig. 9. Performance of clustering using HAC (in baseline) and HAC with different topic models in Web dataset. For each clustering setting (without or with hidden topic models), we changed merging threshold and obtained the maximum F0.5 for comparison.

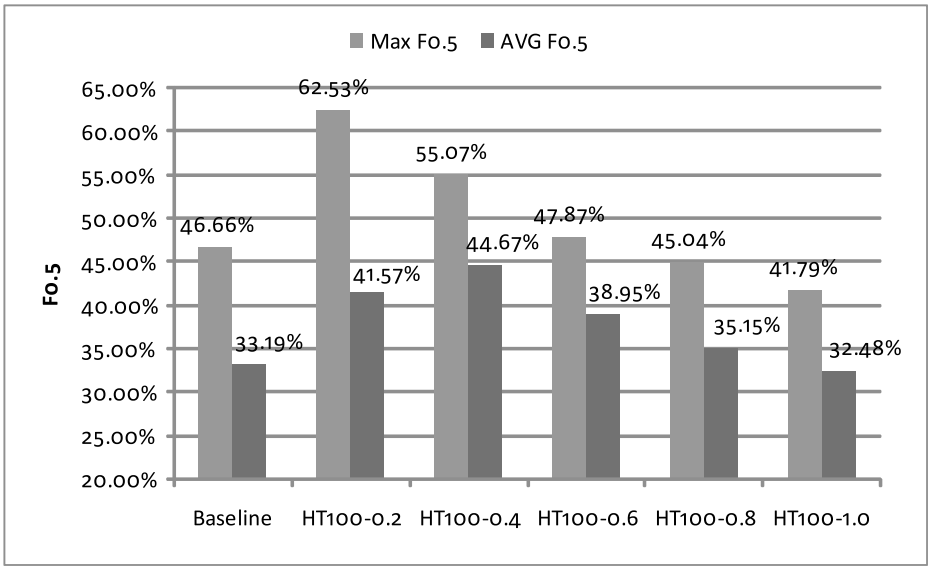
merging threshold is changed from 0 to 0.2) were obtained for comparison in Figure 10(b). As you can see from the figure, HT100-0.2 (lambda = 0.2) and HT100-0.4 (lambda = 0.4) provides the most significant improvements. This means lambda should be chosen from 0.2 to 0.4.

Since the Web dataset is large and much more condensed than real search results, the above evaluation cannot give us a closer look at the performance of the real system. For this reason, we evaluated clustering performance using query dataset which are collected from search results for some sample queries. For each query collection in the dataset, we conducted eight experiments (clustering without hidden topics (the baseline) and with seven different topic models). Taking the maximum F0.5 (and the corresponding precision and recall), we averaged these measures of the same experiment across query collections and summarized in Table V and Figure 11. According to the table, HT20 is still fail to provide an improvement (3.09% worse than the baseline) but the situation is not as bad as in the Web dataset (9.09% worse than the baseline). Clustering with hidden topic models (other than HT20) provides significant improvements in both precision and recall. F0.5 reaches its peak in HT80 with 8.31% better than the baseline. As in the Web dataset, the value of F0.5 changes slightly over different hidden topic models. This supports the previous observation that clustering with hidden topics outperforms the baseline when the number of hidden topics is large enough.

7.4.2 *Detailed analysis.* We considered two cases in which hidden topics can be helpful toward clustering/labeling. The first one is the diverse of word



(a)



(b)

Fig. 10. Baseline vs. HT100 in the Web dataset: (a) Baseline vs. HT100 and lambda = 0.2 (HT100-0.2) and (b) Merging threshold is varied from 0 to 0.2 like in (a). We compared the maximum and average values of F0.5 among clustering with different settings. Note that HT100-*X* (*X* is from 0.2 to 1) means clustering with 100 hidden topic model and lambda = *X*.

choices in the same domain (also the sparseness of snippets). This is not only caused by the large number of words in one domain, but also by a variety of linguistic phenomena such as synonyms, acronyms, new words, and words

Table V. Baseline vs. clustering with different topic models in the query dataset: For each clustering setting, the maximum value of F0.5 for each query collection is obtained. We then average these maximum values across query collections for comparing clustering settings.

	AVG Max F0.5	AVG Precision	AVG Recall
Baseline (HAC)	**65.35%**	**76.86%**	**45.77%**
HT20	62.26%	74.49%	39.97%
HT60	72.72%	80.41%	54.31%
HT80	**73.60%**	**82.76%**	**53.58%**
HT100	72.58%	81.56%	53.90%
HT120	72.19%	81.25%	52.62%
HT160	72.95%	82.07%	51.68%
HT180	72.41%	81.57%	53.45%

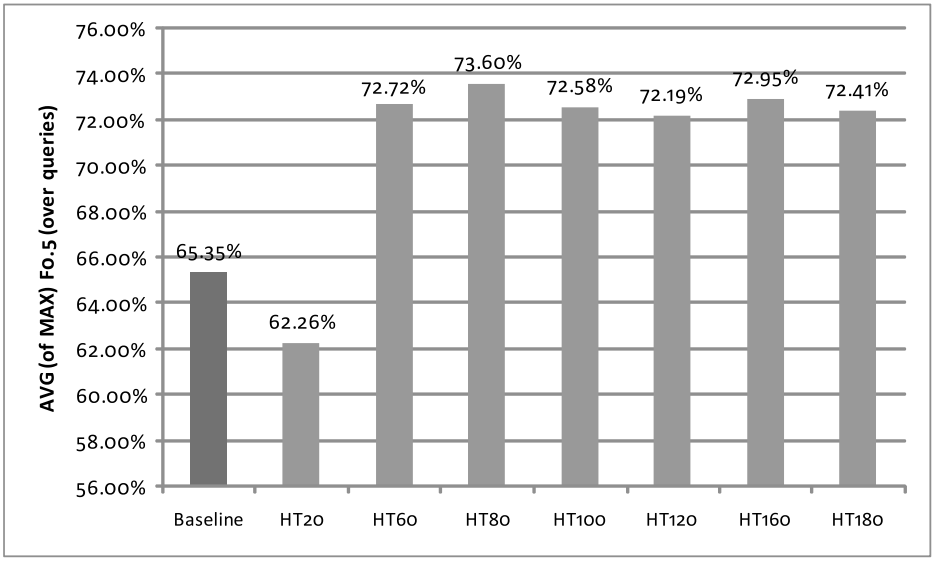
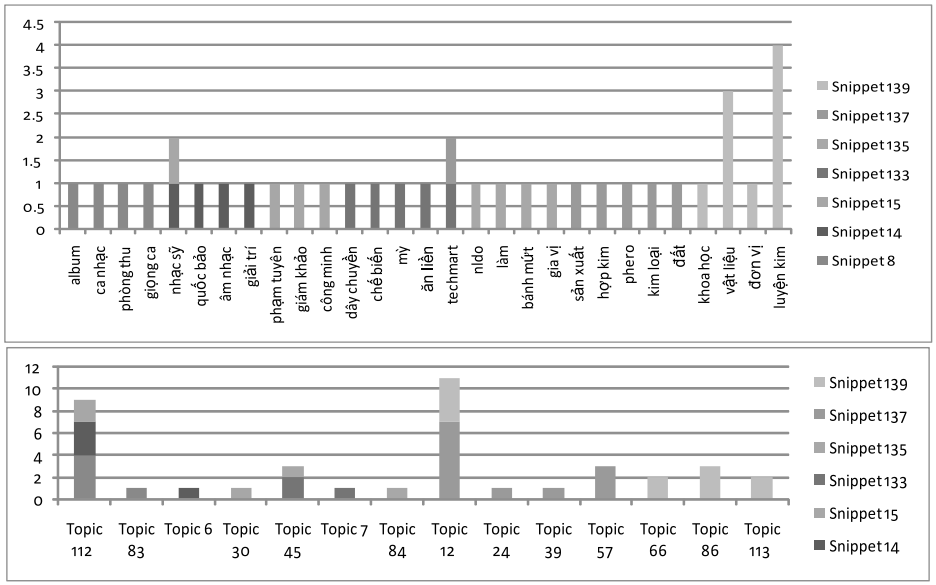


Fig. 11. Baseline and clustering with different topic models on the query dataset.

originating from foreign languages which are probably not covered by dictionaries, and different writing ways such as "color" and "colour." As described in 5, hidden topics from the universal dataset can help us to bridge the semantic gap between these words. As a result, when taking hidden topics into account, the snippets in the same domain but with different word choice can be more similar. The second case is the existence of trivial words but with high frequencies. Although we eliminate stop words before clustering, it is impossible to totally get rid of them.

To better understand the reasons why our proposal works better than the baseline, we analyze one example (Figure 12) to see how hidden topics can be used to reduce data sparseness and mismatching. Figure 12 reveals that snippet 133 and snippet 135 are about the "food industry" but have no term in common. Similarly, snippet 137 and snippet 139 should be in the cluster of "material production" but share no term. Snippet 8, snippet 14, and



Preferred Clusters:

Music Activities: Snippet 8, Snippet 14, Snippet 15

Food Industry: Snippet 133, Snippet 135

Material Production: Snippet 137, Snippet 139

Shared words across snippets:

Nhạc sỹ (musician); Techmart (name of a multiple field website)

Shared topics across snippets:

Topic 112: hát (*sing*), ca sĩ (*singer*), nhạc (*music*), nhạc sĩ (*musician)*, bài hát (*song*), …

Topic 45: món(*dish*), ngon (*delicious*), bánh (*cake*), nhà hàng (*restaurant*), nấu (*cook*), …

Topic 12: nguyên tố (*elements*), kim loại (*metal*), sắt (*iron*), nhiệt độ (*temprature*), vật liệu (*material*), …

Fig. 12. Illustration of the important contributions of hidden topics toward achieving better clustering/labeling.

snippet 15 about "music activities" share only one term "nhạc sĩ" (*musician*) and not close enough for good clustering. This is due to different word choices or the sparseness of the snippets. On the other hand, although snippet 133 and snippet 137 are in totally different topics—the first one is about "food industry" while the second one is about "material production", they share the term "techmart"—the name of the Web site from which two snippets extracted—which is a trivial word here. Since the term-based similarity only makes use of frequencies, and treats words equally, it does not reflect contextual similarity among the snippets. By taking topics into account, snippet 133 and snippet 137 (bridged by the topic 45) are closer in similarity. The same effect happens to the pair of snippet 137 and snippet 138 (bridged by the topic 12), and the triple of snippet 8, snippet 14, and snippet 15 (bridged by the topic 112). Snippet 133 and snippet 137, however, have no topic in common. As a result, the similarity between snippet them decreases in relative to the other pairs in the collection.

Table VI. Testing and Training Data for Cluster Labeling

	#Queries	#Clusters	#Label Candidates
Testing data	4	27	797
Training data	16	119	3113

7.4.3 *Labeling performance.* As mentioned earlier, the query dataset consists of several query collections, each of which include snippets returned by Google for a specific query. We manually partitioned each query collection into key clusters. We then fixed these key clusters and generated "label candidates" for each of them. We also associated each key cluster with a list of scored label candidates (label candidates are assigned "1" if appropriate and "0" otherwise). Based on these specified clusters and their scored label candidates, we used linear regression to learn parameters for relevance score. To do that, we split the query dataset into two parts: (1) the testing data containing query collections of four queries {"tài khoản" (*account*), "táo" (*apple*), chuột (*mouse*) and "ma trận" (*matrix*)}; (2) the training data containing the rest of query collections. Some statistics about the training and testing sets are provided in Table VI.

The training data was put into the module linear regression of Weka[12] to learn parameters for relevance score. We tested two set of features: (1) the full set containing all five feature types as described in the Section 6; and (2) the partial set which exclude features associated with topics of the universal dataset. After the learning process, we achieved the relevance scores as shown in the following:

—Learning with the full set of features: Relevance Score with the 120-topic model of the universal dataset (RS-HT120)

$$\begin{aligned}\text{RS-HT120} &= 0.4963 \times TSIM + 0.5903 \times CDF \\ &- 0.0755 \times TSCORE - 0.3312 \times OTSIM \\ &- 0.064 \times OCDF - 0.2722\end{aligned}$$

—Learning with the partial set of features: Relevance score without hidden topics (RS-base)

$$\begin{aligned}\text{RS-base} &= 0.6389 \times CDF - 0.0866 \times TSCORE \\ &- 0.4177 \times OCDF + 0.891\end{aligned}$$

As we can see from the formula of RS-HT120, TSIM is the second important feature after the most significant one: CDF. The inter-cluster document frequency (OCDF) is quite important in RS-base (with the weight absolute of 0.4177) but less important than inter-cluster topic similarity (OTSIM) in RS-HT120. In both relevance scores, TSCORE does not have much effect on ranking label candidates.

Based on two relevance scores, we ranked label candidates in key clusters in the testing data. We then compared P@5, P@10, and P@20 of two scores

[12]http://www.cs.waikato.ac.nz/ml/weka/

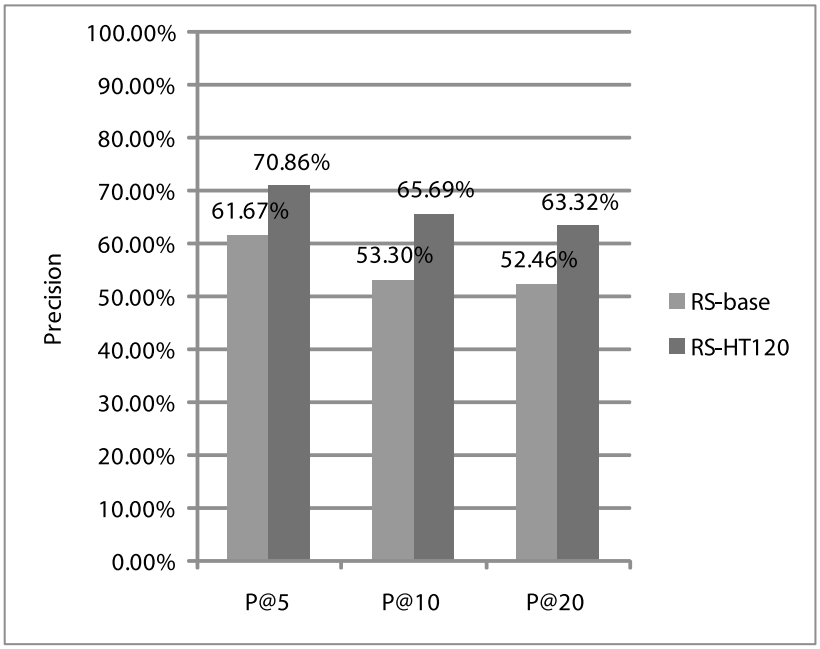


Fig. 13. Comparison of the baseline (labeling without hidden topics) and labeling with 120 topics in the testing collection.

in Figure 13. As observable in the figure, labeling with hidden topics can improve nearly 10% precision on average in the testing dataset. This showed the effective of hidden topics in label assignment.

Figure 14 shows the difference between labeling without and with hidden topics for some key clusters in the testing dataset. For the same cluster "điện thoại" (*mobile phone*) of the query "tài khoản" (*account*), four out of five label candidates in labeling with RS-HT120 are related to "phone" while there are only three good candidates out of five in labeling with RS-base (the first and fifth ones are inappropriate). The same situations occur in the other key clusters of the queries "chuột" (*mouse*), "táo" (*apple*) and "ma trận" (*matrix*). Moreover, better ranking was obtained in labeling with RS-HT120. It can be observed that the first ranking positions of the cluster "điện thoại" (*mobile phone*) (of the query "tài khoản" (*account*)) and the cluster "y tế" (*health services*) (of the cluster "chuột" *mouse*) in labeling with RS-base are "tiền" (*money*) and "dùng" (*take*) repectively which are not as much related to the content of the clusters as "tài khoản điện thoại" (*phone account*) and "thuốc" (*medicine*) in labeling with RS-HT120.

7.4.4 *Computational time analysis.* We compared the computational time between the baseline and clustering and labeling with HT120 in Figure 15. Since topic estimation of the universal dataset is conducted offline, the phase, which requires online computation, is the topic inference for snippets.

Query/Cluster	Labeling without RS-base	Labeling with RS-HT120
Tài khoản/Điện thoại *(Account/Mobile phone)*	**Tiền** *(Money)* **Tài-khoản điện-thoại** *(Phone Account)* **Tài-khoản di-động** *(Mobile Account)* **Điện-thoại di-động** *(Mobile Phone)* **Việt-nam** *(Vietnam)*	**Tài-khoản điện-thoại** *(Phone Account)* **Tiền** *(Money)* **Tài-khoản di-động** *(Mobile Account)* **Điện-thoại di-động** *(Mobile Phone)* **SIM** *(SIM card)*
Chuột/Y tế *("Mouse" or in "Cramp"/Health Services)*	**Dùng** *(Take)* **Thuốc** *(Medicine)* **Bệnh** *(Disease)* **Chữa** *(Cure)* **Loại** *(Type)*	**Thuốc** *(Medicine)* **Dùng** *(Take)* **Chữa** *(Cure)* **Bệnh** *(Disease)* **Chứng chuột-rút** *(The Cramp Trouble)*
Táo/Đồ ăn *("Apple" or "Name of a company", .../Food)*	**Bánh táo-nướng** *(Baked Apple Cake)* **Trái-táo** *(a fruit of Apple)* **Thử một-số** *(Try some)* **Thay-vì ăn** *(Insead of eating)* **Ăn bánh** *(Eating cake)*	**Bánh táo-nướng** *(Baked Apple Cake)* **Trái-táo** *(a fruit of Apple)* **Bột** *(Flour)* **Ăn bánh** *(Eating cake)* **Muối** *(Salt)*
Ma trận/Âm nhạc *(" name of a movie" or "a music album"/Music)*	**Ca-sĩ trẻ** *(Young Singers)* **Hình tượng** *(Image)* **Phuongthanhfc** *(phuongthanhfc)* **Thời-sự** *(Current Events)* **POP** *(POP Music)*	**Nhạc-sỹ** *(Musician)* **Ca-sỹ trẻ** *(Young Singers)* **Âm-nhạc** *(Music)* **POP** *(POP Music)* **Ca-sỹ** *(Singers)*

Fig. 14. Examples of labeling without hidden topics and labeling with 120 topics in the testing collection. Note that the "cluster" in Query/Cluster column is the key cluster label assigned manually.

However, it seems to be acceptable when the number of snippets is around 200 snippets; the default number of snippets to be clustered in Vivisimo [Vivisimo 2008]. Additionally, using hidden topics enables us to remove more rare words than without hidden topics. The point is rare words, for example ones occurring only twice in the snippet collection, sometimes play an important role in connecting snippets. Suppose that we can divide a set of snippets about "movie" into two separated parts: those contains the word "actor" and those includes "director." If we have two snippets in two parts containing the same word such as "movie" which occurs only two times, we can join two parts into one coherent cluster. However, using hidden topics, you can remove such rare words without losing that connection because they all share the topic about "movie." This leads to significant reduction in the size of term vectors; and an improvement is obtained in computational time.

7.4.5 *Query examples.* We obtained four real query collections from Google for four other queries "sản phẩm" (*products*), "Hồng So'n" (*a common name*), "ngôi sao" (*star*), "khủng hoảng" (*crisis*) which are not in the query dataset. In comparison with the query collections in the query dataset, these collections are not cleaned by the fact that we do not exclude minor clusters from them.

We then conducted clustering and labeling with 120-hidden topics and the baseline. The default parameters were set like in IV and the merging threshold of 0.18. Other parameters for the experiments were set according to Table IV. We also submitted the queries to Vivisimo [2008] in order to obtain clustering results. We compared the clusters generated for the queries in

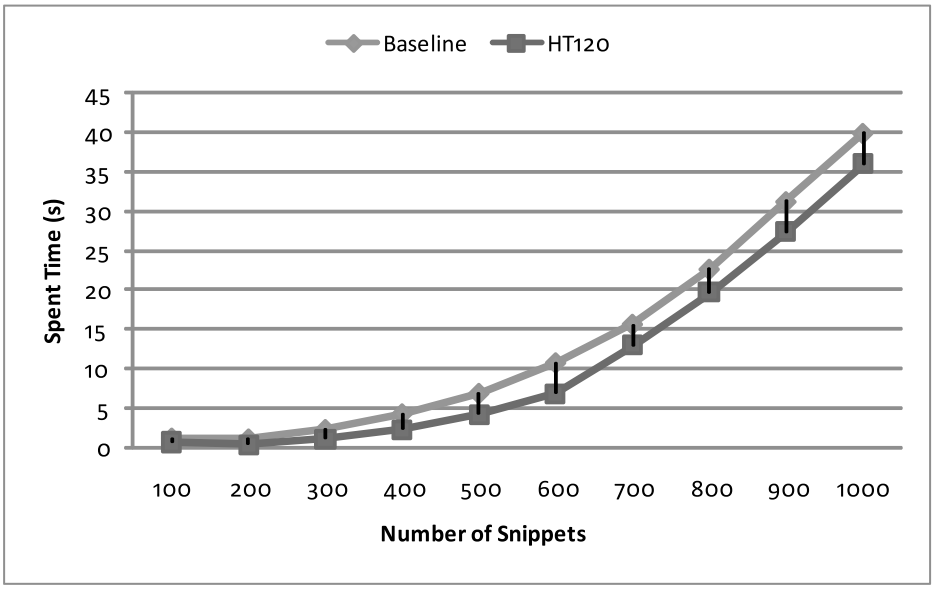


Fig. 15. Computational time of HAC with hidden topics compared to HAC without hidden topics.

clustering/labeling with 120-hidden topic model, in the baseline, in Vivisimo in Figure 16 and Figure 17. The number of snippets in each cluster is written in the bracket next to the cluster label. Note that the query collections, which Vivisimo used, is different from the collection used in the baseline and clustering/labeling with hidden topics.

It can be observable from Figure 16 and Figure 17 that our proposal can provide better clustering/labeling results in comparison with Vivisimo and the baseline. Since Vivisimo is not optimal for Vietnamese, the clustering results are totally unsatisfactory. One obvious example is the cluster label "chính, khủng hoảng tài" of the query "khủng hoảng" (*crisis*). This phrase should be "khủng-hoảng tài-chính" in which "khủng hoảng" (*crisis*) is one valid word and "tài chính" is another valid word with two syllables in Vietnamese. Because word segmentation is not performed in Vivisimo, the two syllables "tài" and "chính" can not be joined to form the correct word. In comparison with the baseline, the clusters generated by our proposed method are better and assigned with more descriptive labels. Considering the query "sản phẩm" (*products*), for example, it is clear that the clusters in the baseline (introduction, news, vietnam) are either two vague or two general in comparison with the clusters in our proposed method (software product, mobile phone, insurance product, etc.). Another example is that the cluster of "singer, music stars" (of the query "star") should be a major cluster, which is recognized in our method, but are not generated in the baseline. For the query "Hồng So'n", the cluster "môn phái" (*martial art group*) in our method actually corresponds to the cluster "Vietnam" in the baseline but the label in our method is much more descriptive.

Hồng Sơn *(A personal name)*		
Clustering and Labeling with HT120	**The Baseline**	**Vivisimo**
Bác sĩ Phạm Hồng Sơn (14) *Doctor Pham Hong Son (14)* **Thủ môn Dương Hồng Sơn (8)** *Goal Keeper Duong Hong Son* **Diễn viên (8)** *Actor/Actress* **Đạo diễn Vũ Hồng Sơn (5)** *Director Vu Hong Son* **Ca sỹ (5)** *Singer* **Môn phái (5)** *Martial Art Group* **Nghệ an, ngôi đền (4)** *Nghe An, Temple* **Xã (4)** *Commune*	**Bác- sĩ Phạm Hồng Sơn (13)** *Doctor Pham Hong Son* **Thủ môn Dương Hồng Sơn (10)** *Goal Keeper Duong Hong Son* **Diễn viên (8)** *Actor/Actress* **Đạo diễn Vũ Hồng Sơn (5)** *Director Vu Hong Son* **Nguyễn Hồng Sơn (4)** *Nguyen Hong Son* **Xã (4)** *Commnune* **Việt Nam (4)** *Vietnam* **Ca sỹ (3)** *Singer*	**Nam, Việt (50)** *Nam, Viet* **Vietnamese (37)** *Vietnamese* **Phạm Hồng Sơn (20)** *Pham-Hong-Son* **Công (23)** *Cong* **Quang (13)** *Quang* **Dương Hồng Sơn (12)** *Duong Hong Son* **Thông tin (11)** *Information* **Dân trí (8)** *Dan tri*
Khủng hoảng *(Crisis)*		
Clustering and Labeling with HT120	**The Baseline**	**Vivisimo**
Ngân hàng (31) *Banks* **Khủng hoảng lương thực (18)** *Food Crisis* **Nền kinh tế (14)** *Economic* **Săn tiền vệ, khủng hoảng nhân sự (10)** *Hunt Players, Human Resource Crisis* **Doanh nghiệp Việt nam (10)** *Vietnam companies* **Xử lý khủng hoảng (9)** *Crisis management* **Giáo dục Việt nam (7)** *Vietnam Education* **Nhà đất (6)** *Real Estate* **Khủng hoảng chính trị (6)** *Political Crisis*	**Tín dụng Mỹ (22)** *United State Credit* **Thế giới (21)** *World* **Tài chính (17)** *Finance* **Vietnam (15)** *Vietnam* **Chính trị (8)** *Politics* **Nhân sự (7)** *Human Resource* **Giáo dục (7)** *Education* **Xử lý khủng hoảng (6)** *Crisis Management* **Thực phẩm thế giới (6)** *World Food*	**Chính, khủng hoảng tài (49)** *the phrase "Financial Crisis" but in the wrong order* **Việt Nam (43)** *Vietnam* **Vietnam(30)** *Vietnam* **Khủng hoảng kinh (21)** *Part of the phrase "Economic Crisis"* **Thông tin (19)** *Information* **Doanh (15)** *in "***Doanh nghiệp***" (Companies)* **Vietnamnet (12)** *Vietnamnet* **Thực, khủng hoảng lương (8)** *the phrase "Food Crisis" but in the wrong order*

Fig. 16. Clustering using HAC with HT120 and labeling with RS-HT120 in new query collections.

7.5 Discussion

Analysis of clustering results affirmed the advantages of our approach. All in all, the main points having been discussed so far include:

—Clustering snippets with hidden topics: It is able to overcome the limitation of different word choices by enriching short, sparse snippets with hidden topics of the universal dataset. This is particularly useful when dealing with Web search results—small texts with only a few words and having less context-sharing. The effective of exploiting hidden topics from the universal dataset is expressed in two aspects: (1) increase similarity

Ngôi sao*(Star)*		
Clustering and Labeling with HT120	**The Baseline**	**Vivisimo**
Blogger (13) *Blogger* **Ca sĩ, ngôi sao ca nhạc (12)** *Singer, Music Stars* **Ngôi sao trẻ (9)** *Young Stars* **Ngôi sao phim (9)** *Film Stars* **Sân Mỹ Đình (9)** *My Dinh Stadium* **Mặc đẹp (8)** *Nice wearing* **Vũ trụ (8)** *Universe* **Tiền vệ (5)** *Half-back*	**Miley Cyrus (10)** *Miley Cyrus* **Thế giới (9)** *World* **Blogger (9)** *Blogger* **Công ty TNHH (9)** *LLC Companies* **Mặc đẹp (8)** *Nice Wearing* **Người lớn (8)** *Adult* **Đầu tiên, vũ trụ (8)** *First, Universe* **Vn, Tiền vệ (7)** *Vn, Half-back*	**Việt, Nam (37)** *Viet, Nam* **Phim (21)** *Film* **Những ngôi sao (14)** *Stars* **Nhac (11)** *Music* **Star (11)** *Star* **Lên, Đang (8)** *Len, Dang* **Bóng (5)** *in "***bóng đá***" (football)* **May mắn (4)** *Lucky*
Sản phẩm *(Products)*		
Clustering and Labeling with HT120	**The Baseline**	**Vivisimo**
Sản phẩm phần mềm (20) *Software Product* **Doanh nghiệp (13)** *Companies* **Doanh số bán (11)** *Sell turnover* **Chất lượng sản phẩm (11)** *Product Quality* **Dịch vụ (9)** *Services* **Điện thoại di động (9)** *Mobile Phone* **Sản phẩm bảo hiểm (9)** *Insurance Product* **Intel, sản phẩm máy tính (8)** *Intel, Computer Products*	**Dịch vụ (13)** *Services* **Giới thiệu, Thái lan (10)** *Introduction, Thailan* **Chất lượng (9)** *Quality* **Tin tức, trang chủ (8)** *News, Homepage* **Thông tin (8)** *Information* **Công nghệ (8)** *Technology* **Việt nam (7)** *Vietnam* **Mã (6)** *Code*	**Giới, thiệu (30)** *Two syllables of the word "***giới thiệu***" (Introduction)* **Dịch (27** *The first syllable of the word "***dịch vụ***" (services)* **Vietnam(26)** *Vietnam* **Tính (17)** *The last syllable of the word "***máy tính***" (computer)* **Mới (17)** *New* **Mua bán (15)** *Selling* **Công ty TNHH (15)** *LLC Companies*

Fig. 17. Clustering using HAC with HT120 and labeling with RS-HT120 in new query collections.

between two snippets having common topics but using different words; and (2) decrease similarity between two snippets sharing non-topic oriented words (including trivial words) which may not be removed completely in the phase of preprocessing. As a result, good clustering is achieved when we are able to assure the "snippet-tolerance" condition, an important feature for a practical clustering system. We conducted evaluation on two datasets—the Web dataset and query dataset—and showed significant improvement of our proposal.

—Labeling clusters using hidden topic analysis: By exploiting hidden topic information, we can assign clusters with more topic descriptive labels. Since

snippets sharing topics are also gather in our method, there are not many repeating words in such clusters. Consequently, word frequency is not enough to determine labels for clusters generated by our method because. In this aspect, phrases sharing topics with most of the snippets in the cluster should be considered significant. Thank to the complete generative model of Latent Dirichlet Allocation, we have a coherent way to map snippets, clusters, and label candidates into the same topic space. As a result, similarity in terms of topics between these clusters, snippets, label candidates are easy to be formalized by using some typical similarity measures such as cosine measure. For evaluation, we split the query dataset into two parts (training data and testing data). We learned two relevance scores from the training data (RS-base, in which we do not consider hidden topic information, and RS-HT120, in which we take topics from the 120-topic model of the universal dataset into account). We then conducted labeling and measured ranking performance (P@5, P@10, and P@20) for two relevance scores in the testing data and showed that labeling with hidden topics can provide better performance.

—Finding collocations in the universal dataset: Using the universal dataset helps to find out meaningful phrases such as "điện_thoại di_động" (*mobile phone*), "thị_trường chứng_khoán" (*stock market*) as labels for clusters. For labeling, we need to extract label candidates and then rank them with regards to some specific conditions. In order to obtain meaningful phrases as label candidates, we find collocations (two or more words commonly used together as fix phrases) using hypothesis testing. Due to the fact that the universal dataset is much larger than snippet collections but snippet collections contain query-oriented text, we find collocations both in the universal dataset and snippet collections. This helps to find out both common noun phrases such as "công nghệ thông tin" (*information technology*), which probably have not enough statistics in snippet collections to be verified as collocations, and named entities or specific phrases which may not occur in the universal dataset such as "Doctor Phạm-Hồng-Sơn" in the snippet collection "Hồng Sơn" (*a common name*).

—Computational time vs. performance: This is an important aspect to consider in any practical applications. Hidden topics bring improvement to clustering process but add extra computational time caused by the analysis process and the usage of topic vectors. For the analysis process, we use Gibbs sampling based on the estimated model. Once the model is converged in the estimation process, 30-50 sampling iterations is quite enough for topic analysis for each snippet collection. So, the complexity of the additional time caused by this step is O(n) in which n is the number of snippets in the collection. However, since the size of these topic vectors are fixed (because the number of topic is fixed) while the number of rare words can be removed without losing the connections between snippets are increased (as analyzed in the previous section), term-vectors of snippets can be reduced in size. This helps us to obtain good clustering performance while decreasing the additional time.

—Flexibility and simplicity: These are advantages of the framework which have been pointed out in our proposal. Here, all we need is to gather a large collection and use it for several phases in our framework. Analysis of the large collection is totally unsupervised, it requires small effort of humans for preprocessing the collection. This is particularly useful when dealing with languages lacking knowledge bases and other linguistic processing toolkits. As a result, this solution works well for Vietnamese and similar languages. The flexibility of our framework is also shown by the fact that the framework does not limit to any topic model or clustering algorithm. We can use CTM or topical *n*-gram model with K-means for to obtain better results while optimizing clustering/labeling time complexity.

8. CONCLUSION

This article presented a framework for clustering and labeling with hidden topics, which (to the best of our knowledge) is the first careful investigation of this problem in Vietnamese. The main idea is to collect a large dataset and then estimate hidden topics for the collection based on one of the recent successful topic models such as pLSI, LDA, CTM. Using this estimated model, we can perform topic inference for snippet collections which need to be clustered. The old snippets are then combined with hidden topics to provide a richer representation of snippets for clustering and labeling. It has been shown that this integration helps overcome the sparseness of snippets returned by search engines and improve quality of clustering. By using hidden topics for labeling clusters, we can assign more descriptive and meaningful labels to the clusters. We have evaluated the quality of the framework via a lot of experiments. Also, through examples and analyzing clusters, we have proved that our approach is somewhat satisfies the three requirements of Web search clustering (high quality clustering, effective labeling and snippet-tolerance) in Vietnamese.

Although, it is not concerned in this paper, hidden topics can be used to obtain overlapping clusters in which a snippet with multiple topics should be put in multiple clusters. Moreover, it is able to re-ranking snippets within generated clusters in which similarity in topics between snippets and the cluster containing them can be used as significant ranking criteria. Thus, in future studies, we will focus on overlapping clusters, re-ranking snippets within clusters as well as generating tree-based instead of flat clustering results.

ACKNOWLEDGMENTS

We would like to express our gratitude to all of the students at the Satellite Laboratory of Knowledge Discovery and Human Computer Interaction, College of Technology, Vietnam National University, Hanoi, who helped us a lot in preparing data and evaluation. We also want to thank the unknown reviewers for their very helpful comments.

REFERENCES

ANDRIEU, C., FREITAS, N., DOUCET, A., AND JORDAN, M. 2003. An introduction to mcmc for machine learning. *Mach. Learn. 50*, 5–43.

BAAMBOO. 2008. Vietnamese search engine. http://mp3.baamboo.coms.

BAGGA, A. AND BALDWIN, B. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 17th International Conference on Computational Linguistics (ACL'98)*. 79–85.

BANERJEE, S. AND PEDERSEN, T. 2003. The design, implementation and use of the ngram statistics. In *Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics*. 370–381.

BANERJEE, S., RAMANATHAN, K., AND GUPTA, A. 2007. Clustering short texts using wikipedia. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*.

BLEI, D. AND LAFFERTY, J. 2006. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*.

BLEI, D. AND LAFFERTY, J. 2007. A correlated topic model of science. *Ann. Appl. Stat. 1*, 17–35.

BLEI, D., NG, A., AND JORDAN, M. 2003. Latent Dirichlet allocation. *J. Mach. Learn. Res. 3*, 993–1022.

BOLLEGALA, D., MATSUO, Y., AND ISHIZUKA, M. 2007. Measuring semantic similarity between words using Web search engines. In *Proceedings of the International World Wide Web Conference (WWW'07)*. 757–766.

CAI, L. AND HOFMANN, T. 2003. Text categorization by boosting automatically extracted concepts. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'03)*.

CHEN, H. AND DUMAIS, S. 2001. Bringing order to the Web: Automatically categorizing search results. In *Proceedings of the International Conference on Human Factors in Computing Systems (CHI'01)*. 145–152.

CUTTING, D. R., KARGER, D. R., PEDERSEN, J. O., AND TOKEY, J. W. 1992. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 318–329.

DEERWESTER, S., FURNAS, G., AND LANDAUER, T. 1990. Indexing by latent semantic analysis. *J. Amer. Soc. Inform. Sci. 41*, 391–407.

FERRAGINA, P. AND GULLI, A. 2005. A personalized search engine based on Web-snippet hierarchical clustering. In *Proceedings of the International World Wide Web Conference (WWW'05)*. 801–810.

GARILOVICH, E. AND MARKOVITCH, S. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'07)*.

GERACI, F., PELLEGRINI, M., MAGGINI, M., AND SEBASTIANI, F. 2006. Cluster generation and cluster labeling for Web snippets: A fast and accurate hierarchical solution. Lecture Notes in Computer Science, vol. 4209, 25–36.

GRIFFITHS, T. AND STEYVERS, M. 2004. Finding scientific topics. *Natl. Acad. Sci. 101*, 5228–5235.

HEINRICH, G. 2005. Parameter estimation for text analysis. Tech. rep., University of Leipzig and vsonix GmbH.

HOFMANN, T. 1999. Probabilistic lsa. In *Proceedings of the Conference on Uncertainly in Artificial Intelligence (UAI'99)*.

HU, J., FANG, L., CAO, Y., ZENG, H.-J., LI, H., AND CHENG, Q. Y. Z. 2008. Enhancing text clustering by leveraging wikipedia semantics. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. 179–186.

JANSEN, B. J., SPINK, A., BATEMAN, J., AND SARACEVIC, T. 1998. Real life information retrieval: A study of user queries on the Web. *SIGIR Forum. 32*, 1, 5–17.

KOTSIANTIS, S. AND PINTELAS, P. E. 2004. Recent advances in clustering: A brief survey. *WSEAS Trans. Inform. Sci. Appl. 1*, 1, 73–81.

MANNING, C. D. AND SCHUTZE, H. 1999. *Foundations of Statistic Natural Language Processing*. MIT Press.

MEI, Q., SHEN, X., AND ZHAI, C. 2007. Automatic labeling of multinomial topic models. In *Proceeding of the Knowledge Discovery and Data Mining Conference (KDD'07)*.

NGO, C.-L. 2003. A tolerance rough set approach to clustering Web search results. Master's thesis, Warsaw University.

NGUYEN, C.-T., NGUYEN, T.-K., PHAN, X. H., NGUYEN, L. M., AND HA, Q. T. 2006. Vietnamese word segmentation with CRFs and SVMs: An investigation. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Compuation (PACLIC'06)*. 215–222.

OSINSKI, S. 2003. An algorithm for clustering Web search result. Master's thesis. Poznan University of Technology, Poland.

PHAN, X. H., NGUYEN, L. M., AND HORIGUCHI, S. 2008. Learning to classify short and sparse text and Web with hidden topics from large-scale data collections. In *Proceedings of the International World Wide Web Conference (WWW'08)*.

POPESCUL, A. AND UNGAR, L. 2000. Automatic labeling of document clusters. http://www.cis.upenn.edu/~popescul/Publications/popesculcolabeling.pdf.

SAHAMI, M. AND HEILMAN, T. 2006. A Web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the International World Wide Web Conference (WWW'06)*.

SCHONHOFEN, P. 2006. Identifying document topics using the wikipedia category network. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'06)*. 456–462.

SOCBAY. 2008. Vietnamese search engine. http://www.socbay.com.

TREERATPITUK, P. AND CALLAN, J. 2006. Automatically labeling hierarchical clusters. In *Proceedings of the International Conference on Digital Government Research (DGRC'06)*.

VIVISIMO. 2008. Clustering engine. http://vivisimo.com/.

VNNIC. 2008. Vietnam Internet Center. http://www.thongkeinternet.vn.

WANG, X., MCCALLUM, A., AND WEI, X. 2007. Topical n-grams: Phrase and topic discovery with an application to information retrieval. In *Proceedings of the 7th IEEE International Conference on Data Mining (DM'07)*. 697–702.

WIKIPEDIA. 2008. Latent semantic analysis. http://en.wikipedia.org/wiki.

XALO. 2008. Vietnamese search engine. http://xalo.vn.

YIH, W. AND MEEK, C. 2007. Improving similarity measures for short segments of text. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'07)*.

ZAMIR, O. AND ETZIONI, O. 1999. Grouper: A dynamic clustering interface to Web search results. *Comput. Netw. 31*, 11-16, 1361–1374.

ZENG, H. J., HE, Q. C., CHEN, Z., MA, W. Y., AND MA, J. 2004. Learning to cluster Web search results. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'04)*.

ZING. 2008. Vietnamese Web site directory. http://directory.zing.vn.

Received September 2008; revised January 2009, April 2009; accepted May 2009