A Probabilistic Concept Annotation for IT Service Desk Tickets

Ea-Ee Jan
IBM T.J. Watson, USA
ejan@us.ibm.com

Kuan-Yu Chen
IBM T.J. Watson, USA
kychen@iis.sinica.edu.tw

Tsuyoshi Idé
IBM T.J. Watson, USA
tide@us.ibm.com

ABSTRACT

Ticket annotation and search has become an important research subject in the IT service desk delivery. Millions of tickets are created yearly to address business users' IT related problems. In IT service desk management, it is critical to first capture the pain points for a group of tickets to determine root cause; secondly, to obtain the respective distributions in order to layout the priority of addressing these pain points. An advanced ticket analytics system utilizes a combination of topic modeling and clustering to address the above issues and the integration of these features into information architecture will allow for a wider distribution of this technology and progress to a remarkable financial impact for IT industry. Topic modeling has been used to extract topics from given documents; each topic is represented by unigram distributions. However, it is not clear how to interpret the results. Due to the inadequacy to render top concepts, in this paper, we propose a probabilistic framework, which integrates topic models, POS tags, query expansion and so on, for the practical challenge. The rigorously empirical experiments demonstrate the consistent and utility performance of the proposed method on real datasets.

Categories and Subject Descriptors

I.2.7 [Artificial Intelligence]: Natural Language Processing-*Text Analysis*; **I.2.7 [Artificial Intelligence]**: Natural Language Processing-*Language Models*;

General Terms

Algorithms, Performance, Theory

Keywords

IT Ticket, Analysis and Indexing, Probabilistic, Topic Modeling

1. INTRODUCTION

IT Service desk is a tens million dollars business for an enterprise. Millions of IT service desk tickets are created yearly to address business users' IT related problems, e.g., password reset, firewall not working, how to setup mail box, etc. For IT service desk management, it is critical to know what key IT problems have been dealt with. Is there a repeating pattern? How many tickets can be used for an automatic solution? These issues all come down to two key questions: what are the pain points and what are the pain points distributions? Today, IT tickets are managed by the Incident-Problem-Change (IPC) ticket system. Structured and unstructured data are stored in a database for ticket management and analysis. However, due to a combination of factors (such as time pressure and back log), structured data is usually ill populated and the descriptive text or unstructured data elements are written by a human agent in a hurry to address their customers concerns. Typos, spontaneous abbreviation, grammatical errors, templates attached with an agent's conversation, addresses, or over length text cutoff are very common. In addition, the IT service desk tickets contain many domain specific technical terms, properties and product names. These terms and names are seldom addressed in today's web anchors and hyperlinks. Therefore, the web resources are less effective when applying to these tickets.

General search technologies provide an excellent mechanism to extract documents related to a given query. Advanced search technologies offer query suggestions, query completion, spelling correction and query expansion to improve search experiences. However, without knowing specific queries, it is still a challenge to find out what the main topics are buried in those documents. When topic extraction technologies are available, using the topic text as a search query, topic distribution and trend changes can be discovered. After the documents are annotated by topics, the semantic search can be further investigated. In this paper, we explore feasibility of topic annotations and distribution on IT service desk tickets for IT service quality improvement and cost saving. The study is solely based on noisy text due to lack of web resources. Additional analysis can be explored by combining time, locations, proper names and other annotations in the future. The topic modeling, the proposed concepts extraction are first addressed in the paper, followed by proposed search algorithm to extract related tickets for each topic. The experiments setup, evaluation results are then discussed to conclude this paper.

Conference'14, Month 1–2, 2010, City, State, Country.

2. TOPIC MODELING

The vector space model (VSM) [1] is the basis for most IR-related researches. VSM is simple, intuitive, efficient and effective. However, VSM suffers from word usage ambiguity e.g. synonyms and polysemous. To complement the weakness of VSM, the latent semantic analysis (LSA) [9] assumes that there is an implicit semantic structure between words and documents, which can be explored by performing SVD on a word-by-document matrix **A**:

$$\mathbf{A}_{M\times N} \approx \mathbf{U}_{M\times K}\,\Sigma_{K\times K}\,\mathbf{V}^{T}_{K\times N} = \tilde{\mathbf{A}}_{M\times N}, \quad (1)$$

Each element $\mathbf{A}_{wd}$ of **A** is the weighted statistics of word *w* in document *d*. After SVD, each word is uniquely associated with a row vector of matrix **U**, while each document is uniquely associated with a column vector of matrix $\mathbf{V}^{T}$.

After LSA, probabilistic topic models have been proposed as a counterpart of the non-probabilistic methods. The probabilistic Latent Semantic Analysis (pLSA) [7] and the Latent Dirichlet Allocation (LDA) [2, 12] are two well-practiced representatives. For probabilistic topic models, each document *d* is taken as a document topic model, consisting of a set of *K* shared latent topics $\{T_1,\ldots,T_k,\ldots,T_K\}$ associated with the document-specific weights $P(T_k|d)$, where each topic T_k in turn offers a unigram distribution $P(w_i|T_k)$ for observing an arbitrary word of the language. Take pLSA as an example, the probability of a word w_i generated by a document *d* is expressed by:

$$P(w_i \mid d) = \sum_{k=1}^{K} P(w_i \mid T_k) P(T_k \mid d) \quad (2)$$

On the other hand, LDA, having a formula analogous to pLSA for document modeling, is thought of as a natural extension to pLSA. The major difference between LDA and pLSA is in the inference of model parameters: pLSA assumes that the model parameters

are fixed and unknown; while LDA places additional a priori constraint on the model parameters, i.e., thinking of them as random variables that follow Dirichlet distributions [2, 7].

3. THE PROPOSED FRAMEWORK

Most of topic models work using a bag-of-words approach. In this paper, we extend the conventional approach to phrases, meaningful *n*-gram from vocabularies, to represent topics. New methods are developed to handle phrase topic modeling for noisy data. Figure 1 depicts the architecture of our system including the pre-processing, the concept analysis and merge, and the search.

3.1 Text normalization Pre-processing

To handle noisy text from IT service desk tickets, we developed text normalization pre-processors, including xml tag, stop words removal, stemming, punctuations and abbreviation normalization. We also use word length to remove email, http link and other functionless words. The "tab" and "carriage return" markers can be optional preserved to provide additional syntactic information.

3.2 Concept Analysis

Topic modeling can be used to dissect word usage cues in each document. We hence leverage topic models to anatomize a given set of tickets. We assume that each latent topic conveys some ideas which are common to a subset of the input data. To better visualize each topic, we want to represent topics by readable descriptions instead of word distributions given by the topic model. To achieve this goal, each topic is addressed by a phrase. To crystallize this idea, we first generate *n*-gram phrases, followed by filtering by predefined Part of Speech (POS) patterns. The POS patterns clean up *n*-gram phrases effectively. We then determine the most suitable phrase to represent a given topic using:

$$P(Phrase_i|T_k) = \frac{P(T_k \mid Phrase_i)P(Phrase_i)}{P(T_k)} \quad (3)$$

The $P(T_k)$ is ignored since it doesn't affect the ranking result. The prior, $P(Phrase_i)$, is calculated using a background *n*-gram model. The likelihood score, $P(T_k|Phrase_i)$, is computed by EM algorithm. To sum up, the prior of a phrase is used to determine the weights of the phrase in a natural language and the likelihood is used to measure the relevance degree between a pair of topic and phrase. Consequently, each topic is now expressed by a phrase which is friendlier for users to understand the physical meaning of the topic. We can further leverage query expansion [3] to robust the phrase when calculate the likelihood score. In this paper, we simply leverage the Rocchio's method [1, 3] to complement each phrase.

3.3 Search

Different from traditional IR approach, IT ticket search needs to address both the precision and confidence score of each document assigned to a topic. This is due to high penalty of human cost incurred by search errors. Although simple strategies are available, such as normalized query likelihood score with a predefined threshold, it is hard to determine a good threshold; the threshold can be sensitive to total numbers of documents in the data set. To mitigate such problem, we define a score, independent of number of documents, for each pair of document and topic:

$$S(D_j, T_k) = \frac{P(D_j \mid T_k)}{\Sigma_{k'=1}^{K} P(D_j \mid T_{k'})} \approx \frac{P(D_j \mid Phrase_k)}{\Sigma_{k'=1}^{K} P(D_j \mid Phrase_{k'})} \quad (4)$$

The likelihood score here can further be decomposed by [4, 13]:

$$P(D_j \mid Phrase_k) = \prod_{w \in D_j} P(w \mid Phrase_k)^{c(w,D_j)} = \prod_{w \in D_j} \left[P_U(w \mid Phrase_k) + P_T(w \mid Phrase_k) + P_{BG}(w)\right]^{c(w,D_j)} \quad (5)$$

$P_U(w|Phrase_k)$ is the probability of word *w* occurring in *k*-th phrase. It is calculated by ML estimator. $P_T(w|Phrase_k)$ is not computed directly from the frequency of the word occurring in the phrase. It is based on $P(w|T_{k'})$) and the phrase likelihood score of each topic $P(T_{k'}|phrase_k)$). $P_{BG}(w)$ is the background language model. By doing so, both literal matching (i.e., password reset vs. reset password) and concept matching (i.e., purchase pc vs. buy desktop) can be integrated into the score. If the score $S(D_j, T_k)$ is greater than a threshold, the document D_j is assigned to topic T_k. This score nicely fits into single document belongs to multiple topics scenario. As a result, a topic has its own document set, and an IR-like system can be leveraged to rank these documents. The ranking list and the representative phrase of each topic can then be obtained.

4. EXPERIMENTAL SETUP & RESULTS

We use two service desk ticket sets from an enterprise for our experiments. The first ticket set is related to mailbox problems; the second set is related to Applications Portals (AP). These two sets represent two different use cases. Mailbox problems are more specific and AP tickets, which related to many applications, cover boarder spectrum. Each set have approximately 20k tickets. Each ticket contains description and resolution text.

Topics are first extracted using topic modeling module, followed by document clustering for each topic using retrieval module. The retrieval results of each query can be treated as a cluster, respectively. We compared proposed probabilistic framework via LDA and pLSA with Lingo [10]. Lingo is a well-practiced method which proposed a "description-comes-first" approach to clustering. Since we don't have resources to label these tickets, nor perform manual evaluation, the widely used clustering integrity matrices, Dunn index (DI) [5] and Davies-Bouldin index (DBI) [6], have been employed for evaluation in this paper. The DI uses minimum of inter-cluster distance divided by maximum of inter-atom paired distances. It penalizes the worst scenarios and is very sensitive to anomaly. To alleviate the worse scenario (Dunn_1), we also calculate DI based 2nd, 3rd, 4th and 5th minimum and maximum distances. The modified DI, called Dunn_2 to Dunn_5, can illustrate if DIs are skewed by bad clusters. The DBI averages distances difference for all sample pairs and clustering pairs. Each index has its strength and weakness; we decide to use both indexes for better understanding of the performance difference.

It is difficult to interpret DI and DBI when numbers of clusters are different. To achieve the same number of clusters across all approaches, we first perform Lingo algorithm to determine number of topics. The topics are extracted, related documents are clustered and the results are compared for all approaches. This setup is more favorable to Lingo. Figure 2 to 5 show Dunn 1-5 results for AP and mailbox tickets via description and resolution, respectively. Figure 6 shows DBI results. By rules of thumb, larger DI indicates better clustering result. Smaller DBI yields better clustering integrity. These figures clear demonstrate our approach consistently outperform Lingo in both modified DI and DBI. In addition, pLSA is slightly better than LDA, except some Dunn_1. Sometimes, pLSA has 1 worse cluster than LDA.

5. CONCLUSION

We propose a probabilistic concept annotation framework. This approach extracts concepts from phrases, instead of bag of words, and clusters documents using the concepts. This methodology has been applied to noisy texts like IT service desk tickets and the results have been compared to LSA likes method using DI and DBI. Additional semantic insights can be further extracted by incorporated with time, property names, product names and server names annotations.

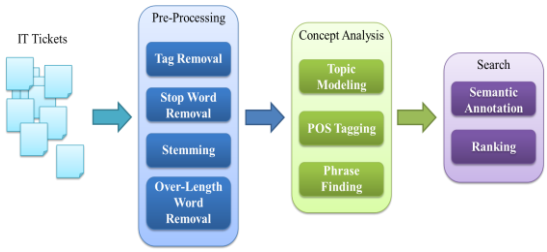

Figure 1: Concept annotation and clustering architecture

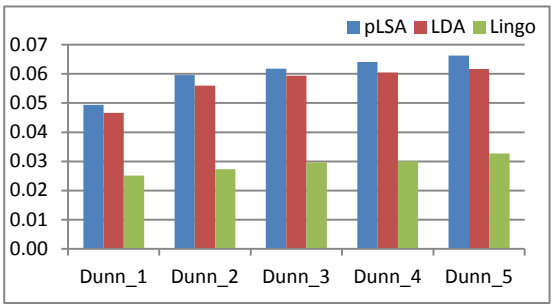

Figure 2: DI comparison for Lingo, LDA and pLSA using description from AP tickets. Notes: larger DI is better.

6. REFERENCES

[1] Baeza-Yates, R. and Ribeiro-Neto, B. 2011 Modern Information Retrieval: The Concepts and Technology behind Search. ACM Press.

[2] Blei, David M. 2012. Probabilistic Topic Models. *Communications of the ACM* 55.4 (April 2012): 77-84.

[3] Carpineto, C. and Romano, G. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *ACM Computing Surveys*, vol. 44, pp.1-56.

[4] Chen, B., Chen, K. Y., Chen, P. N., and Chen, Y. W. 2012. Spoken Document Retrieval with Unsupervised Query Modeling Techniques. *IEEE Transactions on Audio, Speech, and Language Processing*, vol.20, no.9, pp. 2602-2612

[5] Dunn, J. C. 1973. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics*, 3 (3): 32–57.

[6] Davies, David L. and Bouldin, Donald W. 1979. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. PAMI-1 (2): 224–227.

[7] Hofmann, T. 1999. Probabilistic Latent Semantic Indexing. In *Proc. of SIGIR*, pp. 50-57.

[8] Jelinek, F. 1998. Statistical Methods for Speech Recognition. MIT Press, Cambridge, MA, USA.

[9] Landauer, T. K., Peter, W. F., and Darrell, L. 1998. An Introduction to Latent Semantic Analysis. *Discourse processes* 25.2-3: pp. 259-284.

[10] Osinski, S. and Weiss, D. 2005. A Concept-Driven Algorithm for Clustering Search Results. *IEEE Intelligent Systems* 20, 3, pp. 48-54.

[11] Porter, M.F. 1980, An algorithm for suffix stripping, *Program*, 14(3) pp 130−137.

[12] Wei, X. and Croft, W. B. 2006. LDA-based Document Models for Ad-hoc Retrieval. In *Proc. of SIGIR*, pp. 178-185.

[13] Zhai, C. and Lafferty, J. 2001. A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. In *Proc. of SIGIR*, pp. 334-342.

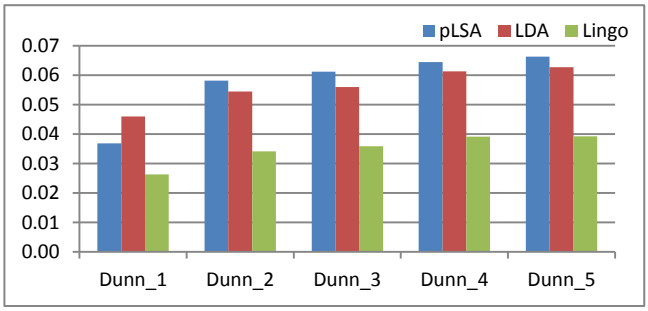

Figure 3: DI comparison for Lingo, LDA and pLSA using resolution from AP tickets. Notes: larger DI is better.

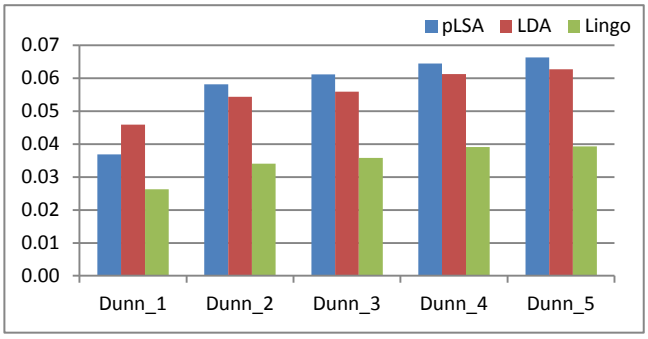

Figure 4: DI comparison for Lingo, LDA and pLSA using description from mail box tickets. Notes: larger DI is better.

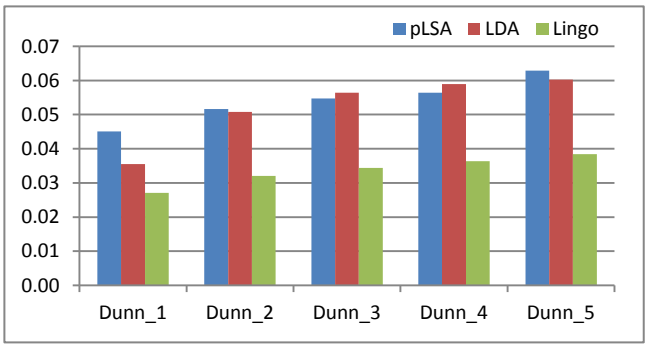

Figure 5: DI comparison for Lingo, LDA and pLSA using resolution from mailbox tickets. Notes: larger DI is better.

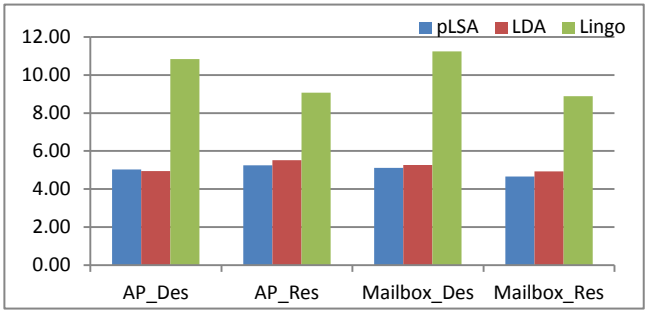

Figure 6: DBI comparison for Lingo, LDA and pLSA using AP and mailbox resolution and resolution tickets. Notes: smaller DBI is better.