HULTECH at the NTCIR-10 INTENT-2 Task: Discovering User Intents through Search Results Clustering

Jose G. Moreno
Normandie Univ, France
UNICAEN, GREYC, F-14032 Caen, France
CNRS, UMR 6072, F-14032 Caen, France
jose.moreno@unicaen.fr

Gaël Dias
Normandie Univ, France
UNICAEN, GREYC, F-14032 Caen, France
CNRS, UMR 6072, F-14032 Caen, France
gael.dias@unicaen.fr

ABSTRACT

In this paper, we describe our participation in the Subtopic Mining subtasks of the NTCIR-10 Intent-2 task, for the English language. For this subtask, we experiment a state-of-the-art algorithm for search results clustering, the $HISGK$-means algorithm and define the users' intents based on the cluster labels following a general framework. From the Web snippets returned for a given query, our framework allows the discovery of users' intents without (1) using the query logs databases provided by the organizers and (2) accessing any external knowledge base. Our best run outperforms the other participants' submissions in terms of D-nDCG@10 and achieves high position in the general ranking.

Team Name

Hultech (Human Language Technology Team - GREYC)

Subtasks

Subtopic Mining (English)

Keywords

Search Results Clustering, Users' Intents Discovery, Web Snippets Mining

1. INTRODUCTION

In NTCIR-10, we participated in the Subtopic Mining subtask of the Intent-2 task for the English language [8], which goal is to retrieve relevant aspects (or subtopics) of ambiguous or multifaceted queries. Within that context, we experiment a state-of-the-art post-retrieval algorithm called $HISGK$-means [3], which provides labeled clusters of Web snippets results for a given query. In particular, the $HISGK$-means has shown competitive results when compared to state-of-the-art algorithms in the context of Search Results Clustering (SRC) [5]. The strongest characteristic of this algorithm is the integration of the labeling process into the clustering step, thus avoiding an extra-step processing and providing meaningful results. In this paper, we propose to evaluate this algorithm for the purpose of users' intent discovery. As a consequence, the followed framework corresponds to a classical SRC problem. But for a given query, the output is not a set of clusters of Web snippets, but instead a set of labels ranked by clusters importance, which may be used as users' intents.

The second important characteristic of the $HISGK$-means algorithm is the fact that it only relies on the analysis of Web snippets. Indeed, any SRC algorithm [1] could be used in this general framework. However, the $HISGK$-means has recently shown similar results to the new state-of-the-art SRC algorithm, called TOPICAL [6], which relies on external knowledge resources [5]. Moreover, unlike other approaches used for users' intents discovery in Intent-2, our methodology avoids the use of query logs/suggestions databases and as such enables the process of unseen or uncommon queries.

In the remainder of this paper, Section 2 describes our framework for processing the Web search results with any SRC algorithm. Section 3 briefly presents the $HISGK$-means algorithm. Section 4 describes the configurations, submissions and results for the Subtopic Mining subtask in Intent-2. Finally, we draw our conclusions in Section 5.

2. GENERAL FRAMEWORK

Our approach consists in a general framework based on the organization of Web search results by SRC algorithms. Within this competition, we used the Hierarchical InfoSimba-based Global K-means ($HISGK$-means) algorithm, which only relies on Web snippet analysis to generate a hierarchical representation of Web search results. In particular, clusters are hierarchically grouped following an iterative process and automatically labelled within the clustering process. Finally, only the cluster labels are kept and ordered by relevance to the query[1]. It is important to notice that any SRC algorithm could be tuned into this framework and the ranking strategy still would be applicable. The procedure to get the users' intents based on SRC algorithms is thus defined in algorithm 1.

Algorithm 1 Framework for Users' Intent Discovery

Input: $TextQuery$, $Algorithm_{SRC}$
Output: $UserIntents$
1. $RankedList = getWebResults(TextQuery)$
2. $ClusterSet = Algorithm_{SRC}.getClusters(RankedList)$
3. $OrderedClusterSet = Sort(ClusterSet)$
4. For each element $cluster_i$ in $OrderedClusterSet$
5. $ClusterName = getClusterName(cluster_i)$
6. $UserIntents_i = TextComb(TextQuery, ClusterName)$
7. return $UserIntents$

In our experiments, (1) the $Algorithm_{SRC}$ is the $HISGK$-means algorithm, (2) $TextComb(.,.)$ is a function returning the concatenation of $TextQuery$ and $ClusterName$ or only

[1]Any ordering function can be used.

$ClusterName$[2] depending of the submission configuration and (3) $Sort(.)$ is the function that ranks the labelled clusters by the number of Web snippets they contain.

3. THE HISGK-MEANS ALGORITHM

The main goal of SRC algorithms is to organize Web snippets into a compact taxonomy, guaranteeing that at each level of the hierarchy, the most suitable number of clusters is found. Then, meaningful labels are assigned to each cluster (i.e. a small set of representative words). Within this context, the $HISGK$-means is a hierarchical top-down divisive hard clustering algorithm, which recursively splits a set of Web snippets based on a variant of the Global K-means algorithm (GK-means) [4] combined with the simplified InfoSimba informative similarity measure, which we call the InfoSimba-based Global K-means ($ISGK$-means). The procedure is defined in algorithm 2.

Algorithm 2 The $HISGK$-means algorithm

Input: A set of Web snippets S and a stop criterion C
Output: A hierarchy
Initialize the root h_0 of the hierarchy to S
Initialize the level of the hierarchy to 1 i.e. $l = 1$
Initialize the number of representative words for the centroid to 2 i.e. $p = 2$
Apply $ISGK$-means at level h_0
Retrieve K_0 clusters $h_{1,1}, \ldots, h_{1,K_0}$
Link all clusters $h_{1,k}$ to their parent h_0
Label all clusters $h_{1,k}$ and h_0 based on their p-sized centroids
$l = l + 1$
$p = p + 1$
for Each cluster $h_{l-1,k}$ and C is true **do**
  Apply $ISGK$-means at level h_{l-1}
  Retrieve K_l clusters $h_{l,1}, \ldots, h_{l,K_l}$
  Link all clusters $h_{l,k}$ to their parent h_{l-1}
  Label all clusters $h_{l,k}$ and h_{l-1} based on their p-sized centroids
  $l = l + 1$
  $p = p + 1$
end for

The main advantages of the $HISGK$-means algorithm are (1) language independence, (2) threshold freedom, (3) cluster labelling included in the clustering process and (4) compactness. In fact, this compactness is mainly due to the use of the simplified InfoSimba informative similarity measure.

3.1 Web Snippet Similarity

While existing methodologies, both polythetic or monothetic, evaluate the similarity between Web snippets based on the exact match of constituents, the InfoSimba similarity measures proposes that two Web snippets are highly related if both share highly related (eventually different) constituents. So, similarity is not any more based on the exact match of constituents but on related words. Indeed, it is clear that both sentences (1) and (2) are similar although they almost do not share any word.

(1) *The jaguar is a wild animal.*

(2) *The pantera onca in huge natural habitats.*

This situation can easily be understood as *jaguar* from sentence (1) is highly correlated to *pantera*, *onca* etc. from sentence (2). The InfoSimba similarity measure proposed in [2] models this phenomenon in an elegant way. Within the polythetic strategy, each Web snippet is represented by a vector of its most relevant words i.e. the set of the best p words selected based on a given score. So, given two Web snippets X_i and X_j, their similarity is evaluated by the simplified InfoSimba measure defined in Equation 1 where $S(.,.)$ is any symmetric similarity measure and each W_{ij} corresponds to the word at the j^{th} position in the vector X_i and X_{ij} corresponds to the weight of the word W_{ij}.

$$ISs(X_i, X_j) = \frac{1}{p^2} \sum_{k=1}^{p} \sum_{l=1}^{p} X_{ik}.X_{jl}.S(W_{ik}, W_{jl}). \quad (1)$$

In our experiments, we use the Symmetric Conditional Probability association measure $SCP(.,.)$ proposed in [7] and defined in Equation 2 to evaluate the correlation between two word vector constituents i.e. $S(.,.)$, where $P(.,.)$ is the joint probability of two words appearing in the same Web snippet and $P(.)$ is the marginal probability of any word appearing in a Web snippet.

$$S(.,.) = SCP(x,y) = \frac{P(x,y)^2}{P(x) \times P(y)}. \quad (2)$$

3.2 The Algorithm

In the particular context of Web snippets clustering, the K-means algorithm needs to be adapted in order to use the InfoSimba similarity measure. Indeed, a Web snippet is not defined by a numerical vector but by a set of p words (i.e. a word context vector of size p) over which a proximity coefficient is defined, in this case, the simplified InfoSimba $ISs(.,.)$ defined in Equation 1. In particular, all words contained in the word context vector are given a score of 1. As a consequence, we define the objective function Q_{IS} to maximize during the clustering process in Equation 3.

$$Q_{IS} = \sum_{k=1}^{K} \sum_{x_i \in \pi_k} ISs(x_i, m_{\pi_k}). \quad (3)$$

It is important to note that a cluster centroid m_{π_k} is now defined by a p-context vector of words $(w_1^{\pi_k}, \ldots, w_p^{\pi_k})$. As a consequence, a way to update cluster centroids must be defined in such a way that Q_{IS} increases at each step of the clustering process[3]. The choice of the best p words representing each cluster is a way of assuring convergence. For that purpose, the procedure UPDATE(π_k) is defined, which consists in selecting p words from the global vocabulary V in such a way that Q_{IS} is improved. The global vocabulary is the set of all words, which appear in any context vector[4]. So, for each word $w \in V$ and any proximity coefficient PC (in this case, the $SCP(.,.)$), its interestingness $\lambda^k(w)$ is computed as regards to cluster π_k as defined in Equation 4 where $s_i \in \pi_k$ is any Web snippet from cluster π_k and only select the p words with higher interestingness value to

[2]More strategies could be explored in an extended version.

[3]It is the theoretical constraint of the K-means algorithm.

[4]Notice that V can be high.

construct the cluster centroid. We can easily show that Q_{IS} is maximized in such a way.

$$\lambda^k(w) = \frac{1}{p} \sum_{s_i \in \pi_k} \sum_{w_q^i \in s_i} PC(w_q^i, w). \quad (4)$$

So, the adaptation of the K-means within the context of Web snippets clustering is straightforwardly defined in algorithm 3 and called the InfoSimba-based K-means (ISK-means).

Algorithm 3 The ISK-means algorithm

Input: Number of K, a set of Web snippet X, List of Centroids L_{in}
Output: K partitions, List of Centroids L_{out}
Initialize K cluster centers in X, randomly and/or using L_{in}
while convergence is not obtained **do**
  Assign each Web snippet $s_i \in X$ to its nearest cluster using $ISs(.,.)$
  Update each cluster center by computing its centroid using UPDATE(π_k)
end while

Now that the well-known K-means has been adapted to the case of Web snippet clustering, we introduce the GK-means clustering algorithm [4], which is at the basis of the overall $HISGK$-means algorithm. The GK-means constitutes a deterministic effective global clustering algorithm for the minimization of the clustering error that employs the K-means algorithm as a local search procedure. The algorithm proceeds in an incremental way. As such, to solve a clustering problem with M clusters, all intermediate problems with $1, 2, ..., M-1$ clusters are sequentially solved. The basic idea underlying the proposed method is that an optimal solution for a clustering problem with M clusters can be obtained using a series of local searches using the classical K-means algorithm. At each local search, the $M-1$ cluster centers are always initially placed at their optimal positions corresponding to the clustering problem with $M-1$ clusters. The remaining M^{th} cluster center is initially placed at several positions within the data space. Since for $M = 1$ the optimal solution is known, it is possible to iteratively apply the above procedure to 2^{nd} optimal solutions for all K-clustering problems $K = 1, ..., M$. In addition to effectiveness, the method is deterministic and does not depend on any initial conditions or empirically adjustable parameters. Moreover, its adaptation to the specific case of Web snippet clustering is direct as shown in algorithm 4. We call this algorithm the InfoSimba-based Global K-means ($ISGK$-means), which is the core of the $HISGK$-means.

Algorithm 4 The $ISGK$-means algorithm

Input: Number of K, a set of Web snippets X
Output: K partitions, List of Centroids L_{out}
Run ISK-means(1, X, [])
$Lcentroids_1 \leftarrow$ centroid of ISK-means(1, X, [])
for Each $k = 2$ to $k = K$ **do**
  Run ISK-means(k, X, $Lcentroids_{k-1}$)
  $Lcentroids_k \leftarrow$ centroids of ISK-means(k, X, $Lcentroids_{k-1}$)
end for

4. EXPERIMENTS

In this section, we present the data preparation process, four different setups and the results obtained for different evaluation metrics proposed by the track organisers[5].

4.1 Data Preparation

First, we collected the returned Web snippets results for the 50 queries provided by the Intent-2 organizers from a commercial Web search engine API[6]. For each query, we used only the top relevant 50 Web snippets for obvious operational constraints from the commercial provider. Some examples of the collected snippets are presented in Table 1.

4.2 Subtopic Mining Runs

Following the Intent-2 restrictions of the Subtopic Mining subtask, we submitted 4 different runs for the English language only. All submission runs are based on the labels obtained by the $HISGK$-means over the 50 Web snippets returned for each query. In particular, the $HISGK$-means is using the complete set of words within each Web snippet as defined in [3] and is coupled to a specific methodology to discover the "best" number of clusters based on the definition of a rational function, which models Q_{IS}[7].

So, the differences between our runs correspond to the use of different levels of the hierarchy produced by the $HISGK$-means. Indeed, each cluster receives a different label depending on its hierarchy level. For example, for the query "Computer Programming", the $HISGK$-means produces 4 clusters at the first level of the hierarchy and the corresponding labels ordered by relevance are "program degree", "language programmer", "introduction subject" and "coding shortened". At the second level, the cluster "program degree", is divided into 2 other sub-clusters ordered by relevance and repectively labeled "skills science software" and "training programmer jobs". This situation similar for all clusters. So, given the hierarchical structure of the $HISGK$-means for each query, our four runs are constructed such as to (1) analyse the effect of the combination of different level labels and (2) understand the effect of the concatenation of the labels with the original query. Each of our four runs is described as follows. As explained in section 2, we remind that all intents are sorted by the importance of the cluster they belong to, i.e. by the number of Web snippets contained in each (sub-)cluster.

hultech-S-E-1A (*Run#1*): Each (sub-)cluster label (independently of its level) is concatenated to the original query.

hultech-S-E-2A (*Run#2*): All first level cluster labels are concatenated to the original query. The labels of leaf subclusters are added without concatenating the query. Note that leaf clusters in the first level give rise to two different intents: with and without the query.

[5]Detailed results of all participants are referred in [8].

[6]In our experiments, we used the Google API, but any other API could be used as well as any (stand-alone) search engine.

[7]As this function is still unpublished work, it is not detailed in this paper. However, similar results can be obtained by experimentally defining the "best" number of clusters as proposed in [6].

Query	*Url* – Title | Snippet
403b	*http://en.wikipedia.org/wiki/403(b)* 403(b) Wikipedia, the free encyclopedia | account holder, but this advantage ceased to exist after the October 2007 Bankruptcy Abuse Prevention and Consumer Protection Act extended bankruptcy protection to 403b
	http://www.irs.gov/retirement/article/0„id=172430,00.html IRC 403(b) Tax-Sheltered Annuity Plans | A 403(b) tax-sheltered annuity (TSA) plan is a retirement plan, similar to a 401(k) plan, offered by public schools and certain 501(c)(3) tax-exempt
	http://www.ehow.com/info_8061335_403b.html What Is a 403B? . eHow.com | A 403B is the Internal Revenue Service tax code used to describe a Tax Sheltered Annuity (TSA) plan. The TSA is frequently called a "403B" plan, which nonprofits
Computer Programming	*http://www.ehow.com/computer-programming/* Computer Programming - How To Information . eHow.com | Get essential tips and useful Computer Programming info on eHow. Learn about everything from Visual Basics Programming, Learn Programming, Recover Deleted Files, and
	http://www.bls.gov/ooh/Computer-and-Information-Technology/Software-developers.htm Software Developers : Occupational Outlook Handbook : U.S. Bureau ... | Software developers usually have a bachelor's degree in computer science and strong computer-programming skills. Pay. The median annual wage of applications software
	http://suite101.com/computerprogramming Computer Programming . Suite101.com | Top Tips for File Handling in C - Avoid Potential Pitfalls! Opening a File and Reading From It In C : C Programming Tutorial. Comma Separation Using strtok in C: How

Table 1: Examples of Web snippets for the queries "403b" and "Computer Programming".

hultech-S-E-3A (*Run#3*): Each cluster label, at the first level only, is concatenated to the original query.

hultech-S-E-4A (*Run#4*): Each leaf cluster label is concatenated to the original query.

In order to better understand all runs, we present the obtained intents for the query "Computer Programming" in Table 2. Note that the run "hultech-S-E-1A" (*Run#1*) gives the best average results over all our other submissions. In particular, it includes the complete hierarchy without label repetitions and integrates the hierarchy particularities in the ranking of the predicted intents. The overall information of our submissions is presented in Table 3.

Run #	Number of Intents	Avg Intents per query
1	526	10.52
2	619	12.38
3	378	7.56
4	240	4.80

Table 3: Overall information of our submissions.

Note that the average number of submitted users' intents for *Run#1* is closer to the evaluation setups, which propose to only evaluate the first 10 intents of each run. Indeed, on one hand, *Run#3* and *Run#4* obtain, in average, less intents than the number that could be evaluated. On the other hand, *Run#2* over-estimates the number of intents. In fact, this situation can be addressed in different ways. In order to fit as much as possible to the evaluation framework, different strategies can be proposed: (1) tuning the SRC algorithm to retrieve a set of relevant 10 clusters, (2) tuning the number of input Web snippets so that more or less intents can be found or (3) propose different combinations of labels to express users' intents. However, this could be seen as tuning the system towards a given evaluation task. We prefered to propose a more realistic solution, which does not depend on the evaluation setups. As such, no specific tuning was performed for the Intent-2 task.

4.3 Results

The evaluation framework, proposed in the Intent-2 task, uses the following metrics to access the performance of each run: I-rec@10, D-nDCG@10 and D#-nDCG@10. The results presented in Table 4 show the performance of our four runs under theses setups.

Run #	I-rec@10	D-nDCG@10	D#-nDCG@10
1	0.3680	**0.5368** ▲^{2,3}	**0.4524**▲^{2}
2	0.2697	0.2986	0.2841
3	0.3045	0.3345	0.3195
4	**0.3688**	0.4807▲^{2}	0.4248 ▲^{2}

Table 4: Results for our submissions. The best result for each metric is marked in bold.

It is important to notice that these results correspond to the "revised" version reported in [8][8]. In particular, significance of the results is verified using the two-sided randomized Tukey's HSD (Honestly Significant Difference) at $\alpha = 0.05$ test. The symbol ▲^{i} denotes significant improvements of the current run with respect to *Run#i*.

[8]A complete summary of the results can be found there.

Run Id	Intents ordered by relevance
hultech-S-E-1A (*Run#1*)	computer programming program degree computer programming program degree skills science software computer programming language programmer computer programming introduction subject computer programming coding shortened computer programming program degree training programmer jobs
hultech-S-E-2A (*Run#2*)	computer programming program degree program degree skills science software language programmer computer programming language programmer introduction subject computer programming introduction subject coding shortened computer programming coding shortened program degree training programmer jobs
hultech-S-E-3A (*Run#3*)	computer programming program degree computer programming language programmer computer programming introduction subject computer programming coding shortened
hultech-S-E-4A (*Run#4*)	computer programming program degree skills science software computer programming language programmer computer programming introduction subject computer programming coding shortened computer programming program degree training programmer jobs

Table 2: Examples of our submissions for the query "Computer Programming" (QueryID = 416).

On the one hand, the results clearly show that *Run#1* outperforms all other runs in almost all the evaluated metrics. On the other hand, the worst run is *Run#2*, which does not necessarily include the initial query. This information is interesting as it shows that most of the systems for query intent discovery propose to keep the initial query in the description of the intent. This issue should be discussed as one may think that the general intent is given by the query, which the user keeps in mind. As such, we could expect that the intent would not depend so much on the initial query. Additionally, the results of *Run#2* for D-nDCG@10 and D#-nDCG@10 are inferior, with statistical relevance, compared to *Run#1* and *Run#4*, which are the best two runs overall. The second important interpretation of the results is the fact that using the different levels of the cluster hierarchy drastically improves the performance of discovery. Indeed, *Run#3* is the second worst run.

In order to better understand the behaviour of each run, we propose to compare the results of *Run#2*, *Run#3*, *Run#4* to *Run#1* over each query. As such, Figure 1 should be interpreted as follows. On the X-axis, the queries are sorted by relevance to *Run#1*, i.e. the first query on the X-axis is the one for which *Run#1* got its best result, the second query is the one for which *Run#1* got its second best result, and so on and so forth. The illustration clearly shows that only *Run#4* can compete with *Run#1* in some cases.

4.4 Comparison to Other Teams

A total of 8 teams[9] participated in the Subtopic Mining subtask organized by Intent-2. Each team submitted a maximum of 5 runs for a total of 34 submissions. Table 5 presents the results of the top-3 competitors for each one of the three evaluation metrics. Our top submission, *Run#1*, outperforms the other participants in terms of D-nDCG@10. However, in terms of I-rec@10 and D#-nDCG@10, the results show lower performance compared to the other top-3 teams[10]. But the statistical analysis shows no relevant difference between all runs of all top-3 teems. In particular, the low results for the I-rec@10 metric can be explained by the fact that our framework is using a few number of Web snippets. As such, it is likely that the methodology misses relevant users' intents as they may be expressed in Web pages, which do not belong to the 50 retrieved results[11].

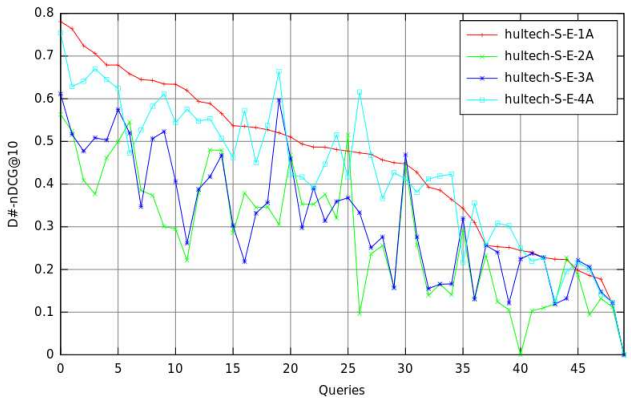

Figure 1: Comparison between our four runs in terms of query intent for D#-nDCG@10.

[9]7 research teams and 1 team as baseline.

[10]Detailed description of the results can be found in [8].

[11]Additional experiments must be conducted to verify the variations with the use of more Web snippets.

Team	I-rec@10	D-nDCG@10	D#-nDCG@10
THUIR	0.4364	0.5062	**0.4713**
THCIB	**0.4431**	0.4657	0.4544
hultech	0.3680	**0.5368**	0.4524

Table 5: Comparison with best run of top-3 teams. The best result for each metric is marked in bold.

In order to better understand the differences between our best run (*Run#1*) and all other runs submitted to Intent-2, we compare its results to the best and worst results obtained for any of the overall 34 submissions. The results are illustrated in Figure 2. On the X-axis, the queries are sorted by relevance to *Run#1* (exactly the same way as for Figure 1) and the Y-axis represents the best and worst results of all 34 runs for the given query. Finally, the average performance for a given query is computed as the arithmetic mean of the results obtained for the 34 runs. The results clearly show that our best submission improves over the average performances. However, it is possible to find a better run for almost any of the query, although this run can be any of the 34 proposed ones.

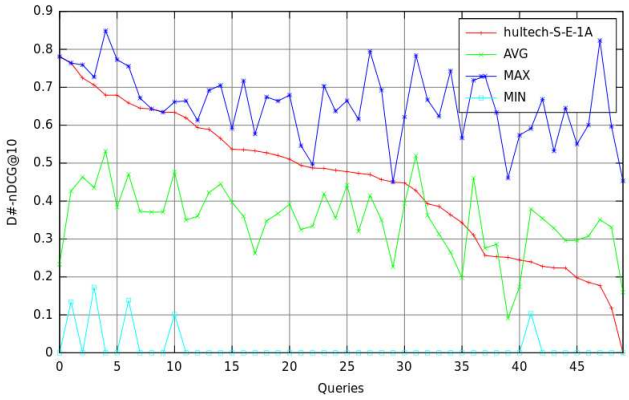

Figure 2: Comparison between *Run#1* and all other submissions in terms of query intent for D#-nDCG@10.

As a consequence, we propose a different illustration to compare *Run#1* to the two other top-3 runs (i.e. *THUIR* and *THCIB*) in Figure 3. Each line corresponds to the results of each one of the top-3 runs ordered by decreasing performance and the MAX line would correspond to the "hypothetical" best configuration i.e. the best results obtained by any of the 34 runs ordered also by decreasing performance. The results show that (1) none of the best three runs reaches the maximum possible results, (2) the top-3 runs have a similar average behaviour although the individual results (i.e. for a given query) differ rather drastically (see Figure 4) and (3) our framework shows worst results for less succeeded intents, which indicates that some queries are rather difficult to deal with. One possible explanation for this last case is the fact that only 50 Web snippets are used for each query, but of course further work needs to be carried out to confirm this intuition.

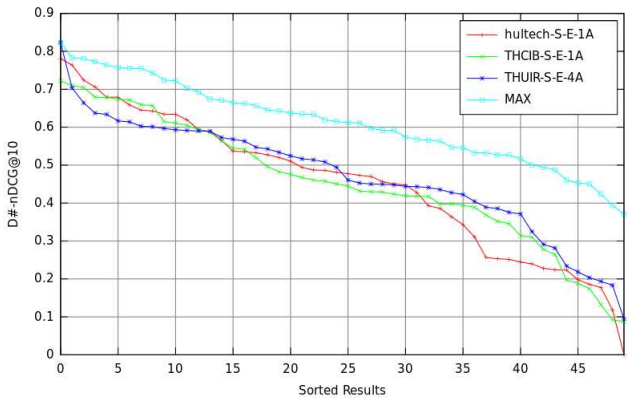

Figure 3: Sorted comparison between *Run#1*, *THUIR* and *THCIB* in terms of query intent for D#-nDCG@10.

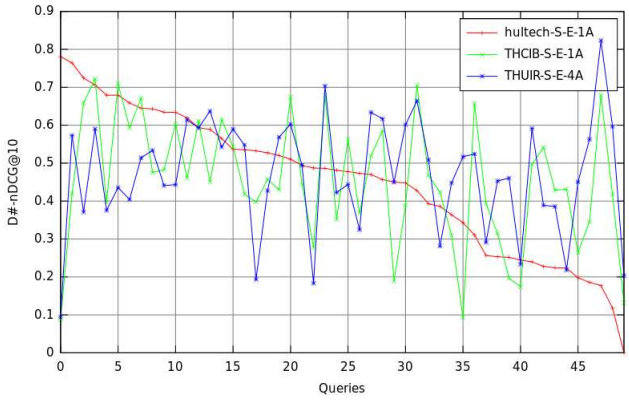

Figure 4: Comparison between *Run#1*, *THUIR* and *THCIB* in terms of query intent for D#-nDCG@10.

5. CONCLUSIONS

In the context of the Subtopic Mining subtask of the NTCIR-10 Intent-2 task, we proposed a general framework based on SRC algorithms to identify query intents from Web search results. Experiments were performed with a state-of-the-art SRC algorithm, the *HISGK*-means and promising results were achieved. In terms of D-nDCG@10, the "revised" results showed that our top submission run outperformed all other 34 runs and high scores were also achieved in terms of D#-nDCG@10. Particularly, the top submission (i.e. *THUIR*) outperformed our top results by 5% of improvements but without statistical significance.

6. REFERENCES

[1] C. Carpineto and G. Romano. Mobile information retrieval with search results clustering : Prototypes and evaluations. *Journal of the American Society for Information Science*, 60:877–895, 2009.

[2] G. Dias. Information digestion, 2010. HDR Thesis, University of Orléans (France).

[3] G. Dias, G. Cleuziou, and D. Machado. Informative polythetic hierarchical ephemeral clustering. In *International Conference on Web Intelligence*, pages 104–111, 2011.

[4] A. Likasa, V. N., and J. Verbeek. The global k-means clustering algorithm. *Pattern Recognition*, 36:451–461, 2003.

[5] J. G. Moreno and G. Dias. Using text-based web image search results clustering to minimize mobile devices wasted space interface. In *34th European Conference on Information Retrieval*, pages 532–544, 2013.

[6] U. Scaiella, P. Ferragina, A. Marino, and M. Ciaramita. Topical clustering of search results. In *5th ACM International Conference on Web Search and Data Mining*, pages 223–232, 2012.

[7] J. Silva, G. Dias, S. Guilloré, and J. Lopes. Using localmaxs algorithm for the extraction of contiguous and non-contiguous multiword lexical units. In *9th Portuguese Conference in Artificial Intelligence*, pages 113–132, 1999.

[8] T.Sakai, Z. Dou, T. Yamamoto, Y. Lui, M. Zhang, and R. Song. Overview of the ntcir-10 intent-2 task. In *NTCIR-10*, 2013.