Large Scale GPU Based Inference for the Infinite Relational Model

Hansen, Toke Jansen; Mørup, Morten; Hansen, Lars Kai

Publication date:
2010

Document Version
Early version, also known as pre-print

Large Scale GPU Based Inference for the Infinite Relational Model

Toke Jansen Hansen, Morten Mørup, and Lars Kai Hansen
Section for Cognitive Systems
DTU Informatics
Technical University of Denmark
{tjha,mm,lkh}@imm.dtu.dk

Abstract

Mining for associations in large scale graphs is a problem of both theoretical and practical importance, e.g., study of social networks, market basket analysis and collaborative filtering, and in web scale text processing. As these problems mount the need for computationally efficient tools that can accurately predict relations and also provide relevant insight in underlying mechanisms, becomes urgent. The infinite relational model (IRM) has been proposed as a Bayesian generative model for graphs. Generative models can provide accurate predictions and through inference of relevant latent variables they can inform the user about mesoscale structure. The IRM forms a low-rank probabilistic representation of a bipartite network by co-clustering, ie., coordinated grouping of each mode to best model the graph. A benefit of the IRM over existing co-clustering approaches is that the model explicitly exploits the statistical properties of binary graphs and allows the number of components of each mode to be inferred from the data. We present experimental results on much larger IRM's than previously analyzed. This is made possible by the observation that the structure of the bipartite IRM is very well suited for GPU computing. The GPU implementation achieves a speedup of two orders of magnitude relative to conventional CPUs for model estimation in networks with $N = 10^7–10^9$ connections.

1 Introduction

Co-clustering also denoted bi-clustering and two-mode clustering is the simultaneous clustering of rows and columns in a matrix. Co-clustering has already gained popularity in several fields, including bio-informatics for identification of co-regulated genes and gene functional annotation [12], collaborative filtering and market basket analysis for identification of user and product segments [22], text miming [5, 3, 23] for identification of related terms and documents, and social network modeling to find relationships between agents and behavior [1, 7]. Many co-clustering methods have been proposed including the iterative refinement approach of Hartigan [9], heuristics for grouping columns and rows in an adjacency matrix [21], spectral [5] and information theoretic approaches [4], and methods inspired by community detection in complex networks [20, 7]. For reviews on co-clustering, see also [13, 12, 7]. In this contribution we consider the specific case of co-clustering of binary data, i.e., for which co-clustering becomes the analysis of bipartite graphs. This type of relational data is typically represented by the adjacency matrix $\boldsymbol{A} \in \mathbb{B}^{I \times J}$, where $A_{ij} = 1$ if entity i is related to entity j and $A_{ij} = 0$ otherwise. We are interested in approaching co-clustering by generative models. Generative models serve the two main purposes in machine learning namely, to identify predictive relations and, via the inference of relevant latent variables, to provide interpretation and domain insight. Bayesian generative models such as the relational model [11, 23] and the stochastic block model [17, 7] have previously been proposed for analysis of binary relations. The so-called infinite relational model (IRM) for bipartite graphs is based on the decomposition

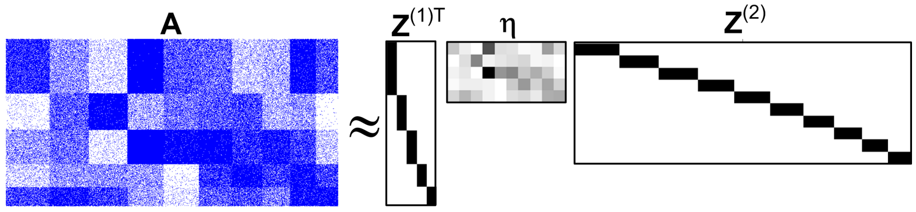

Figure 1: Illustration of the relational model for bipartite graphs. The adjacency matrix $\boldsymbol{A}$ is according to the co-clustering approach decomposed into row and column clusters $\boldsymbol{Z}^{(1)}$ and $\boldsymbol{Z}^{(2)}$ such that η_{lm} gives the extend in which row group l is related to column group m. Contrary to other co-clustering approaches the relational model is specialized for the modeling of binary graph data based on the Bernoulli likelihood as measure of deviation in reconstruction and automatically infers the numbers of row and column clusters from a potential infinite number of clusters.

$\boldsymbol{A} \approx \boldsymbol{Z}^{(1)^\top}\boldsymbol{\eta}\boldsymbol{Z}^{(2)}$, where $\boldsymbol{Z}^{(1)}$ and $\boldsymbol{Z}^{(2)}$ are latent binary clustering assignment matrices while η_{lm} indicates how group l in $\boldsymbol{Z}^{(1)}$ is related to group m in $\boldsymbol{Z}^{(2)}$, see also figure 1. The IRM can be considered a lossy compression/low-rank approach where the bipartite graph is compressed into blocks of homogenous regions in the graph. The IRM can lead to insight in the domain as the latent variables provide for detection of communities, see e.g., the analysis of unipartite graphs by the IRM [11, 14]. A further strong benefit of the IRM over existing co-clustering approaches is that the numbers of row and column clusters L and M are inferred from data. Thus, the IRM has already proven useful in the analysis of a variety of complex networks [11, 23, 14], however, these previous analyzes have been limited to networks of only a few million links. The IRM has not yet been applied to realistic, web scale, networks of say $10^8 - 10^{10}$ links.

In this contribution we investigate the feasibility of GPU based inference for the IRM and find significant performance advantages over traditional CPU calculation, yielding 10^2 speed-ups on a single graphics card. To the best of our knowledge this is the first attempt at large scale co-clustering of bipartite graphs on GPUs. Finally we demonstrate our capability to analyze real-life complex networks that are orders of magnitude larger than previous attempts based on the IRM.

2 Methods

The generative process for the the Relational Model [11, 23] also denoted the Stochastic Block Model [17] is given by:
◇ Sample the row cluster probabilities, i.e. $\boldsymbol{\mu}^{(1)} \sim Dirichlet(\alpha^{(1)}/K^{(1)}\boldsymbol{e}^{(1)})$.
◇ Sample row cluster assignments, i.e. $i = 1, \ldots, I$ $z_i^{(1)} \sim Discrete(\boldsymbol{\mu}^{(1)})$.
◇ Sample the column cluster probabilities, i.e. $\boldsymbol{\mu}^{(2)} \sim Dirichlet(\alpha^{(2)}/K^{(2)}\boldsymbol{e}^{(2)})$.
◇ Sample column cluster assignments, i.e. $j = 1, \ldots, J$ $z_j^{(2)} \sim Discrete(\boldsymbol{\mu}^{(2)})$.
◇ Sample between cluster relations, i.e. $l = 1, \ldots, L$ and $m = 1, \ldots, M$ $\eta_{lm} \sim Beta(\beta^+, \beta^-)$.
◇ Generate links, i.e. $i = I, \ldots, J$ and $j = 1, \ldots, J$ $A_{ij} \sim Bernoulli(\boldsymbol{z}_i^{(1)}\boldsymbol{\eta}\boldsymbol{z}_j^{(2)})$.
Where $K^{(1)}$ and $K^{(2)}$ denote the number of row and column clusters respectively whereas $\boldsymbol{e}^{(1)}$ and $\boldsymbol{e}^{(2)}$ are vectors of ones with size $K^{(1)}$ and $K^{(2)}$.[1]

Blocked Gibbs Sampling in the IRM
As the Dirichlet distribution is conjugate to the Discrete distribution and the Beta distribution is conjugate to the Bernoulli distribution both $\boldsymbol{\mu}^{(1)}$, $\boldsymbol{\mu}^{(2)}$ and $\boldsymbol{\eta}$ can be integrated out analytically (i.e. collapsed). Furthermore, the limits $K^{(1)} \to \infty$ and $K^{(2)} \to \infty$ (i.e., a priori assuming a potential infinite number of clusters in mode one and two) forming the Infinite Relational Model (IRM) has an analytic solution given by the Chinese Restaurant Process (CRP) [23, 11] also denoted the

[1]According to the generative process the probability of observing a link is governed by the class assignment of the i^{th} and j^{th} nodes, i.e. $\boldsymbol{z}_i^{(1)}$ and $\boldsymbol{z}_j^{(2)}$ as well as the inter-group relations $\boldsymbol{\eta}$ where η_{lm} gives the probability of observing a link between class l and class m. Links in the graphs are conditionally independent given their respective node assignments.

Dirichlet Process (DP) [6, 15]. The analytic integration of $\boldsymbol{\mu}^{(1)}$, $\boldsymbol{\mu}^{(2)}$ and $\boldsymbol{\eta}$ introduce however dependence between the nodes within a mode such that the assignments of each node in mode one or two no longer can be updated in parallel. Rather than collapsing parameters of the model blocked sampling can be invoked which potentially also result in better mixing [10, 24]. The CRP/DP can be approximated by the truncated stick breaking construction (TSB), see also [10, 24]. The truncation error becomes insignificant when the model is estimated for large values of $K^{(1)}$ and $K^{(2)}$, see also [10]. The blocked sampling based on TSB result in the following updates for the IRM parameters

$$\begin{aligned}
&P(z_i^{(1)} = l|\boldsymbol{A}, \boldsymbol{Z}^{(2)}, \boldsymbol{\mu}^{(1)}, \boldsymbol{\eta}) \propto \mu_l^{(1)} exp(\textstyle\sum_{m,j} \log(\frac{\eta_{lm}}{1-\eta_{lm}})A_{ij}z_{mj}^{(2)} + \log(1-\eta_{lm})z_{mj}^{(2)}),\\
&P(z_j^{(2)} = m|\boldsymbol{A}, \boldsymbol{Z}^{(1)}, \boldsymbol{\mu}^{(2)}, \boldsymbol{\eta}) \propto \mu_m^{(2)} exp(\textstyle\sum_{l,i} \log(\frac{\eta_{lm}}{1-\eta_{lm}})A_{ij}z_{li}^{(1)} + \log(1-\eta_{lm})z_{li}^{(1)}),\\
&P(\eta_{lm}|\boldsymbol{A}, \boldsymbol{Z}^{(1)}, \boldsymbol{Z}^{(2)}, \beta^+, \beta^-) \sim Beta(N_e(l,m)+\beta^+, N_o(l,m)+\beta^-),\\
&\mu_l^{(1)} = v_l^{(1)} \textstyle\prod_{l'=1}^{l-1}(1-v_{l'}^{(1)}), \quad \text{where} \quad v_l^{(1)} \sim Beta(1+n_l^{(1)}, \alpha^{(1)} + \sum_{l+1}^{K^{(1)}} n_l^{(1)}),\ v_{K^{(1)}}^{(1)} = 1,\\
&\mu_m^{(2)} = v_m^{(2)} \textstyle\prod_{m'=1}^{m-1}(1-v_{m'}^{(2)}), \quad \text{where} \quad v_m^{(2)} \sim Beta(1+n_m^{(2)}, \alpha^{(2)} + \sum_{m+1}^{K^{(2)}} n_m^{(2)}),\ v_{K^{(2)}}^{(2)} = 1.
\end{aligned}$$

where $N_e(l,m) = \sum_{ij} z_{li}^{(1)} A_{ij} z_{mj}^{(2)}$ and $N_o(l,m) = \sum_{ij} z_{li}^{(1)} z_{mj}^{(2)} - N_o(l,m)$ denote the number of links and non-links between group l and m respectively. In [24] blocked sampling based on TSB was found to perform as well as collapsed sampling.

GPU Computing Aspects

Our implementation consists of a CPU part for sampling $\boldsymbol{\eta}$, $\boldsymbol{\mu}^{(1)}$, $\boldsymbol{\mu}^{(2)}$ and a GPU part for sampling $\boldsymbol{Z}^{(1)}$, $\boldsymbol{Z}^{(2)}$. The main reason for not keeping everything in GPU code is that only the sampling of the cluster assignment matrices exhibit an arithmetic intensity sufficient for taking advantage of the GPU architecture, furthermore the limited amount of GPU memory force us to discretize the sampling into suitable sized subproblems where asynchronous operations can be used to hide the associated memory latency when transferring data between the host memory (CPU) and device memory (GPU). The greatest challenge in the development of high performance GPU applications is to keep device memory latency low and thereby all threads occupied with arithmetic operations [18, 19]. Unfortunately sparse matrix representations often result in scattered memory accesses thereby making efficient memory segmentation difficult. Performing sparse matrix operations on CUDA capable GPUs have already been analyzed [2], but since the posterior likelihood of $\boldsymbol{Z}^{(1)}$ and $\boldsymbol{Z}^{(2)}$ contain structure that can be exploited to yield a more efficient memory pattern, no existing implementation seems ideal. To minimize the memory footprint of the cluster assignment matrices we utilize the property that a node can only belong to a single cluster, hence they are encoded as vectors where each entry corresponds to the respective cluster number. With respect to the adjacency matrix $\boldsymbol{A}$, we presort each mode according to node degree and chunk it into aligned blocks of memory where each entry encodes the position of a link. Using the proposed format the calculation of $\boldsymbol{A}\boldsymbol{Z}^{(2)^\top}$ and $\boldsymbol{A}^\top\boldsymbol{Z}^{(1)^\top}$ (constituting the main computational bottleneck in the updates of $\boldsymbol{Z}^{(1)}$ and $\boldsymbol{Z}^{(2)}$ respectively) correspond to incrementing link counts in the dense result matrix directly indexed by our sparse representation. The implemented memory layout minimizes thread divergence and provides coalesced memory transfers in $\boldsymbol{A}$, whereas the scattered accesses in $\boldsymbol{Z}$ can be cached in local memory. Remaining dense matrix operations are handled using the CUBLAS library, whereas we utilize yet another custom CUDA kernel to sample the cluster assignment matrices done by applying inverse transform sampling requiring a stream of uniform random numbers per thread. Since present devices have no built-in random number generator, pseudo random numbers are generated in GPU code using a hybrid of a Linear Congruential Generator (LCG) and a combined Tausworthe generator that has been shown to remove all the statistical defects observed in each separate generator [16].

3 Results and Discussion

In order to evaluate the speedup of GPU computation over traditional CPU computation we analyze synthetically generated graphs, varying the number of nodes, links and clusters[2]. Figure 2 shows the speedup that our GPU implementation[3] achieve over the semantically equivalent CPU implementation. The speedup is positively correlated with the number of links and clusters, since increasing

[2]Hardware configuration; Intel Core i7-920 2.66GHz with 24 GB of memory, NVIDIA C1060 Tesla with 4 GB of memory. Ubuntu 9.10, Linux kernel 2.6.31, NVIDIA CUDA driver version 256.40.

[3]The illustrated results are for floating point precision.

these will improve the overall arithmetic intensity. Increasing in the number of nodes while keeping the number of links fixed yields a negative impact in the speedup, because increasing the sparsity reduces the load on each parallel processing GPU thread. For 512 clusters our implementation reaches more than a 140 times speedup. With respect to numerical precision we have for the synthetical data not observed any significant difference between floating and double precision as measured in terms of AUC scores ($p = 0.05$). However, for even larger problem sizes we foresee that floating point precision might lead to slower convergence as rounding errors accumulate in the inverse transform sampling algorithm. Thus, in terms of speed there may exist a tradeoff between slower convergence in terms of floating point rounding errors and slower calculation in terms of double precision arithmetic.

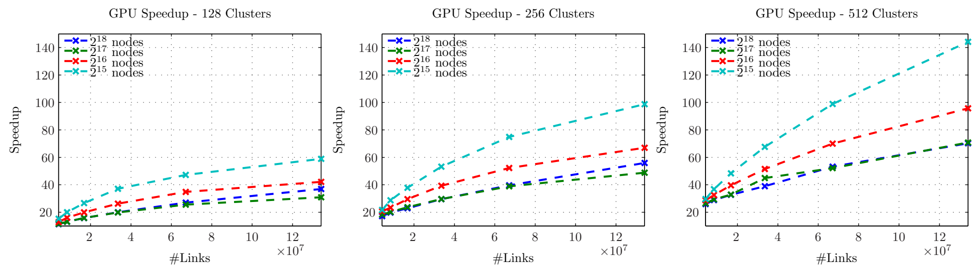

Figure 2: The speedup for GPU over CPU computing. The number of nodes for the two modes are here identical. Left panel; speedup for 128 clusters in each mode, middle panel; speedup for 256 clusters in each mode, right panel; speedup for 512 clusters in each mode.

To demonstrate the GPU implementation on real-life large scale data we turn to topic modeling and consider abstracts from U.S. National Library of Medicine (PubMed) based on the bag of words representation publicly available from [8]. Disregarding the number of occurrences of the words we obtain the adjacency matrix $\boldsymbol{A}$ where $\boldsymbol{A}_{ij} = 1$ if word i occurred in document j and $\boldsymbol{A}_{ij} = 0$ otherwise. The PubMed abstracts contain 140K words and 8 million documents with approximately 500 million word occurrences. We analyzed the dataset based on a maximum of 500 clusters and treated 1 % of links and an equivalent number of non-links as missing at random and analyzed each graph five times with different randomly generated sets of links and non-links treated as missing. Each sampling iteration took less than 80 seconds and the link-predictive performance did not significantly improve after about 100 iterations so the analysis was terminated upon 500 iterations. To investigate the uniqueness of the solution we calculated the normalized mutual information (NMI) measuring how similar the extracted clusters are across the five sampling runs and found that $\text{NMI}^{(1)} = 0.73(0)[0.07(0)]$ whereas $\text{NMI}^{(2)} = 0.67(0)[0.00(0)]$, hence, the extracted clusters all are quite consistent across the sampling runs and much more similar than would be expected by random[4]. Finally, table 1 illustrates three representative clusters extracted by the IRM, relating to various areas of research.

4 Conclusion

Using efficient GPU implementations we demonstrated the first large scale co-clustering results based on the infinite relational model. In terms of scalability the current GPU implementation allows us to go beyond the datasets analyzed here and make inference in even larger bipartite networks, provided that $\boldsymbol{Z}^{(1)}$ can fit in device memory when sampling subsets of $\boldsymbol{Z}^{(2)}$ and vice versa. Furthermore, the discretization that we apply in our current implementation makes it straightforward to utilize systems with multiple GPUs, simply by parallel distribution of the discretized subproblems. Finally, we note that the GPU implementation trivially generalizes to other inference approaches such as variational Bayes [24]. Future work will focus on extending the proposed framework for graphs with temporal dynamics, i.e. online auctions and collaborative content creation such as blogs. We imagine a potential in exploiting random projection methods to efficiently subsample data while still recovering network structure supported by the data within the available timeframe imposed by online systems.

[4]In parenthesis are given the standard deviation on last digit and in brackets the NMI obtained by random.

References

[1] M. J. Barber, M. Faria, L. Streit, and O. Strogan. Searching for communities in bipartite networks. In C. C. Bernido and V. C. Bernido, editors, *Searching for Communities in Bipartite Networks: Proceedings of the 5th Jagna International Workshop*, volume 1021, pages 171–182. AIP Conf. Proc., 2008.

[2] Nathan Bell and Michael Garland. Efficient sparse matrix-vector multiplication on CUDA. NVIDIA Technical Report NVR-2008-004, NVIDIA Corporation, December 2008.

[3] Pavel Berkhin and Jonathan D. Becher. Learning simple relations: Theory and applications. In *In Second SIAM Data Mining Conference*, pages 420–436, 2002.

[4] I. Dhillon, S. Mallela, and D. Modha. Information-theoretic co-clustering. *in Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2003).*, 2003.

[5] Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274, New York, NY, USA, 2001. ACM.

[6] Ferguson, T. S. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[7] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010.

[8] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[9] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.

[10] Hemant Ishwaran and Lancelot F. James. Gibbs sampling methods for stick-breaking priors. *JASA*, (96), 2001.

[11] Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *Artificial Intelligence, Proceedings of the National AAAI Conference on*, 2006.

[12] Sara C. Madeira and Arlindo L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[13] I Van Mechelen, H H Bock, and P De Boeck. Two-mode clustering methods: a structured overview. *Statistical methods in medical research*, 13(5):363–394, October 2004.

[14] M. Mørup, K. H. Madsen, A. M. Dogonowski, H. Siebner, and L. K. Hansen. Infinite relational modeling of functional connectivity in resting state fmri. In *to appear in Neural Information Processing Systems 2010*, 2010.

[15] Radford M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Computational and Graphical Statistics, Journal of*, 9:249–265, 2000.

[16] Hubert Nguyen. *GPU Gems 3*. Addison-Wesley Professional, 2007.

[17] Krzysztof Nowicki and Tom A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

[18] NVIDIA. *NVIDIA CUDA Programming Guide 2.0*. 2008.

[19] NVIDIA. *NVIDIA CUDA Reference Manual 2.0*. 2008.

[20] Jörg Reichardt and Stefan Bornholdt. Clustering of sparse data via network communitiesa prototype study of a large online market. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(06):P06016+, June 2007.

[21] A. Tanay, R. Sharan, and R. Shamir. Biclustering algorithms: A survey. *Handbook of Computational Molecular Biology*, 2004.

[22] Haixun Wang, Wei Wang, Jiong Yang, and Philip S. Yu. Clustering by pattern similarity in large data sets. In *In SIGMOD*, pages 394–405, 2002.

[23] Zhao Xu, Volker Tresp, Kai Yu, and Hans-Peter Kriegel. Learning infinite hidden relational models. *Uncertainity in Artificial Intelligence (UAI2006)*, 2006.

[24] S. Yu K. Yu H.-P. Kriegel Z. Xu, V.Tresp. Fast inference in inifinite hidden relational models. *Mining and Learning with Graphs (MLG'07)*, 2007.

A Supplementary GPU Performance Material

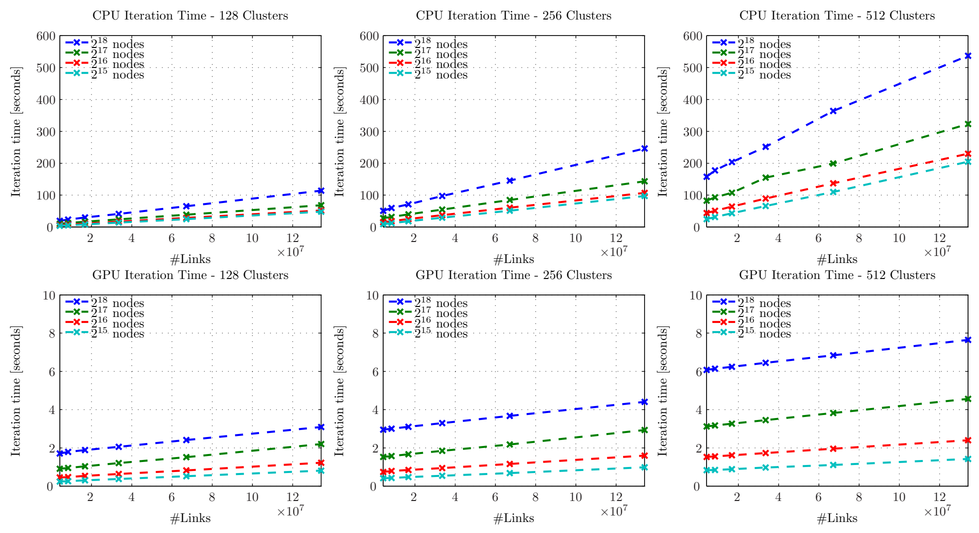

Figure 3: Shows the measured execution times used to obtain the speedup illustrations in figure 2.

B Examples of Discovered Structure in PubMed

∼ Neurology	∼ Cancer	∼ Cardiology
amygdala, ca1, ca3, caudate, cell bodies, circuitry, colliculus, dendrites, dentate gyrus, dorsal root, dorsolateral, entorhinal, gabaergic, ganglion cell, geniculate, innervated, interneuron, lesioned, limbic, mesencephalic, midbrain, motoneuron, myelinated, neocortex, neocortical, nerve fiber, neuropil, perikarya, purkinje, putamen, raphe, rostral, soma, somata, substantia nigra, synapse, tegmental, thalamic, thalamus.	adenocarcinoma, adjuvant, benign, biopsies, biopsy, breast, breast cancer, cancer patient, carcinoma, carcinomas, cell carcinoma, chemotherapy, grade,histologic, histological, histologically, histology, hyperplasia, invasion, invasive, irradiation,lung cancer, lymph node, lymph nodes, malignancies, malignancy, malignant, metastases, metastasis, metastatic, neoplasm, neoplastic, prognostic, prostate, radiation, radiotherapy, recurrence, tumour.	angina, antiarrhythmic, arrhythmia, arrhythmias, atrial, atrial fibrillation, atrioventricular, atrium, av, beat, cardiomyopathy, congestive heart, ecg, echocardiographic, ejection fraction, electrocardiogram, electrocardiographic, electrophysiologic, end-diastolic, endocardial, epicardial, fibrillation, implantable, left atrial, left ventricle, lv, myocardial ischemia, pacemaker, pacing, pectoris, qr, right ventricular, tachycardia,ventricular function, ventricular hypertrophy, ventricular tachycardia.

Table 1: Example of three topic groups given in red, green and blue respectively extracted by the IRM for the PubMed data.