IncBL: Incremental Bug Localization (Appendix)

Zhou Yang, Jieke Shi
Singapore Management University
Singapore
{zyang, jkshi}@smu.edu.sg

Shaowei Wang
University of Manitoba
Canada
shaowei@cs.umanitoba.ca

David Lo
Singapore Management University
Singapore
davidlo@smu.edu.sg

Basically, the incremental update method proposed by Rao et al. in [1] relies on the VSM model, which associates *term frequency*, *document frequency* and *inverse document frequency* for each term to represent documents. In a VSM model, each document is represented as a vector, and the w^{th} value of the vector is the tf-idf weight that is computed by $tf_m(w) \times idf(w)$. Rao et al. tried to find a way to update tf, df and idf incrementally, thus update the VSM model without repetitive computation, but two points are needed to be reinterpreted so that the method can cope with all the update situations correctly.

A. Incremental update for document frequency

The Section IV.B in [1] elaborates how to update the df matrices in VSM models. Assuming the m^{th} file is modified, for the w^{th} term, the $df(w)$ will be updated with the following formula:

$$df^{new}(w) = df^{old}(w) + sign(A_m^{new}(w) - A_m^{old}(w)) \tag{1}$$

where $sign(\cdot)$ returns $+1$ for positive inputs, -1 for negative inputs, and 0 for 0. A^{old} and A^{new} are term-document matrices before and after this change. If the m^{th} file or the w^{th} term are not in A^{new} (or A^{old}), $A_m^{new}(w)$ (or $A_m^{old}(w)$) is set as 0.

Unfortunately the equation 1 only works correctly when $A_m^{new}(w)$ (or $A_m^{old}(w)$) $= 0$ or $A_m^{new}(w) = A_m^{old}(w)$. The $sign(\cdot)$ function is computed by the difference between $A_m^{new}(w)$ and $A_m^{old}(w)$, thus its result is always 1 or -1 in case of $A_m^{new}(w) \neq 0$, $A_m^{new}(w) \neq 0$ and $A_m^{new}(w) \neq A_m^{new}(w)$, in which $df(w)$ should **not** be changed. Following this formula, the df value will be improperly updated to affect the entire model representation.

Let's set an example to explain it more clearly. If 5 documents are involved in a VSM model and the term 'bugs' occurs once in each document, the *document frequency* for term 'bug' is 5, and its *term frequency* for each document is 1. Then the first document is modified and the occurrence of the term 'bugs' in this document is added to 2, which leads that the *term frequency* for term 'bugs' in the first document is changed to 2. If we follow Rao et al's method, the *document frequency* for the term 'bugs' will be $5 + sign(2 - 1) = 6$, while we only have 5 documents, so the formula can not handle the issue when we only modified the existing terms. However, it works when we add new terms or delete terms completely. In the example above, if the first document doesn't have the term 'bugs' in the beginning, *document frequency* for the term 'bugs' is 4, then we modified the document in the same way as above, the updated *document frequency* will be $4 + sign(2 - 0) = 5$, which is perfectly right.

The incremental update formula should be revised as follows:

$$df^{new}(w) = df^{old}(w) + [sign(A_m^{new}(w)) - sign(A_m^{old}(w))] \tag{2}$$

where $sign(\cdot)$ for occurrences of term are computed firstly, then the $sign(A_m^{new}(w)) - sign(A_m^{old}(w))$ contains 4 cases: 1) $0 - 0 = 0$, that means $A_m^{new}(w) = 0$ and $A_m^{old}(w) = 0$; 2) $1 - 0 = 1$, that is in case that $A_m^{new}(w) > 0$ and $A_m^{old}(w) = 0$; 3) $1 - 1 = 0$ in case $A_m^{new}(w) > 0$ and $A_m^{old}(w) > 0$; and 4) $0 - 1 = -1$ in case $A_m^{new}(w) = 0$ and $A_m^{old}(w) > 0$. In this way we can handle the update for df values correctly.

B. Incremental update for inverse document frequency

In Rao et al's work, the *inverse document frequency* is computed by:

$$idf(w) = log(\frac{M}{df(w) + 1}) \tag{3}$$

where M is the number of documents. If M changes, all idf values need to be updated. This means the incremental method can not deal with changes in the number of documents because all model parameters must be recomputed with new idf values.

The equation 3 can be transformed to $log(M) - log(df(w) + 1)$. So one solution we propose to update idf is as follows:

$$idf^{new}(w) = idf^{old}(w) + log(\frac{M + \Delta M}{M}) \tag{4}$$

where ΔM is the change on M. In other words, we use the old idf that is not affected to update the value of idf incrementally rather than computing the idf value from scratch.

References

[1] S. Rao, H. Medeiros, and A. Kak, "An incremental update framework for efficient retrieval from software libraries for bug localization," in *Proceedings - Working Conference on Reverse Engineering, WCRE*, 2013, pp. 62–71.