Investigation of Zhihu Tag/Doc Data

June 3, 2017

Leaving aside the dependency between document feature and tags, this document only focuses on the Tag Graph (TG) data and the Tag-Doc Bipartite (TDB) data. The ultimate objective of this investigation is to understand, at least at an intuitive level, whether and to what extent, the structure of TG may be useful for document prediction.

The TG should first be cleaned by removing skip connections. For each of the quantities below (with exceptions noted below), we are looking to obtaining its mean, standard deviation, maximum/minimum and histogram (each taking one line to get in matlab).

Exceptions:

1. For the items labelled with "constant", just give the numbers.
2. For the items labelled with "scatter", give a scatter plot.

1 Quantities on TG

1. (constant) The number of verties
2. (constant) The number of source (i.e., most senior) vertices
3. (constant) The number of terminal (i.e., most junior) vertices
4. For vertex v, let $\pi(v)$ be the parents of v. Get $|\pi(v)|$.
5. For vertex v, let $\chi(v)$ be the children of v. Get $|\chi(v)|$.
6. For each vertex v, let $\mathcal{T}(v)$ be the set of all terminal vertices t for which there is a path from v to t. Get $|\mathcal{T}(s)|$ for every source vertex s.
7. For each vertex v, let $\mathcal{S}(v)$ be the set of all source vertices s for which there is a path from s to v. Get $|\mathcal{S}(t)|$ for every terminal vertex t.
8. For each pair (u, v) of vertices, let $\mathcal{P}(u, v)$ be the set of all paths from u to v or from v to u. Get $|\mathcal{P}(s, t)|$ for every source-terminal vertex pair (s, t).
9. For each vertex v, let $\mathcal{D}(v)$ be the set of all descendants of v. For each source vertex s, get $|\mathcal{D}(s)|$.
10. For each vertex v, let $\mathcal{A}(v)$ be the set of all descendants of v. For each terminal vertex t, get $|\mathcal{A}(t)|$.
11. For any two sets, let $J(A, B)$ denote the Jaccard coefficient/index between A and B. For every two source vertices s and s', get $J(\mathcal{D}(s), \mathcal{D}(s'))$
12. For every two terminal vertices t and t', get $J(\mathcal{A}(t), \mathcal{A}(t'))$

13. For any two vertices u and v for which there is path from u to v or from v to u, let $\overline{d}(u,v)$ the length of the longest path between u and v. For each vertex v, let $\overline{\ell}_{\max}(v) := \max_{s\in\mathcal{S}(v)} \overline{d}(s,v)$. Get $\overline{\ell}_{\max}(t)$ for each terminal vertex t.

14. For each vertex v, let $\overline{\ell}_{\min}(v) := \min_{s\in\mathcal{S}(v)} \overline{d}(s,v)$. Get $\overline{\ell}_{\min}(t)$ for each terminal vertex t.

15. (scatter) For each terminal vertex t, get $(\overline{\ell}_{\max}(t), \overline{\ell}_{\min}(t))$.

The above quantities have not involved statistics related to the depths of vertices. But the notion of depth can be tricky to define since a vertex have multiple source ancestors (and terminal descendants). The following is my idea of defining depth, noted here to further explore if needed.

First, for each vertex v, we need to define a notion of importance for v. Such a notion of importance should be related to the frequency at which v is used to tag a document. Then based on this notion of importance, for each vertex v, we can somehow find a tree (or backward tree) that includes v and is most important. Then we can define the depth for v relative to this tree.

2 Quantities on TDB

3 Quantities joining TG and TDB