International Conference on Data Mining (SDM), pages 10–18. SIAM, 2013.

[26] S. Sintos and P. Tsaparas. Using strong triadic closure to characterize ties in social networks. In *SIGKDD*, 2014.

[27] Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han. Co-author relationship prediction in heterogeneous bibliographic networks. In *International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2011, Kaohsiung, Taiwan, 25-27 July 2011*, pages 121–128, 2011.

[28] J. Tang, T. Lou, and J. M. Kleinberg. Inferring social ties across heterogenous networks. In *WSDM*, 2012.

[29] J. Ugander, L. Backstrom, and J. Kleinberg. Subgraph frequencies: Mapping the empirical and extremal geography of large graph collections. In *WWW*, 2013.

[30] C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo. Mining advisor-advisee relationships from research publication networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 203–212, 2010.

[31] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *ICDM*, 2002.

[32] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.

7 Appendix

7.1 Computing Squares on Large Graphs

Computing the exact number of squares on large graphs can be a challenging problem. This connects to a large literature on counting triangles and other network motifs, but the size of the graph in our problem poses challenges for even the most efficient known techniques. Here, we briefly point out a scalable heuristic that can compute the feature using a randomized algorithm by utilizing HyperLogLog sketches [8]. This approach does not provide good worst case guarantees as it is, but in practice we've observed reasonable performance from the heuristic. We also note that the counts produced by these sketches are fed to a classifier, which provides an additional layer of robustness for error.

Consider a node a with direct neighbors B in G_L, and second degree neighbors C (excluding the nodes distance two away that are in B). We need to compute the squares feature for each $b \in B$. A naive method of computing this would involve forming a list of b's neighbors that are also in C, and doing pairwise intersections between the lists of all b's. We note that one can flip this computation around, with each $c \in C$ having a list of neighbors that is intersected with the set B, as then each pair in the resulting list has a unique square path.

This idea can be implemented efficiently with HyperLogLog sketches that implement intersections [8]. In particular, each c computes a sketch of its neighbors, and each b is annotated with a sketch of B's. Then, for an existing (b, c) edge, the intersection of these sketches is exactly one more than the number of squares that b participates in. This computation can be implemented efficiently in MapReduce [5].

We re-emphasize that in our experiments in this work, we do not use this approximation (since the sample size was tractable with exact computation). We note this algorithm here to demonstrate that the squares feature can be computed on extremely large graphs in an efficient manner.