A Graph-Search Framework for GeneId Ranking (Extended Abstract)

William W. Cohen
Machine Learning Department
Carnegie Mellon University
Pittsburgh PA 15213
wcohen@cs.cmu.edu

1 Introduction

One step in the curation process is *geneId finding*—the task of finding the database identifier of every gene discussed in an article. GeneId-finding was studied experimentally in the BioCreatIvE challenge (Hirschman et al., 2005), which developed testbed problems for each of three model organisms (yeast, mice, and fruitflies). Here we consider *geneId ranking*, a relaxation of geneId-finding in which the system provides a ranked list of genes that might be discussed by the document. We show how multiple named entity recognition (NER) methods can be combined into a single high-performance geneId-ranking system.

2 Methods and Results

We focused on the mouse dataset, which was the hardest for the BioCreatIvE participants. This dataset consists of several parts. The *gene synonym list* consists of 183,142 synonyms for 52,594 genes; the *training data* consists of 100 mouse-relevant Medline abstracts, associated with the MGI geneId's for those genes that are mentioned in the abstract; the *evaluation data* consists of an additional 50 mouse-relevant Medline abstracts, also associated with the MGI geneId's as above; the *test data* consists of an additional 250 mouse-relevant Medline abstracts, again associated with MGI geneId's; finally the *historical data* consists of 5000 mouse-relevant Medline abstracts, each of which is associated with the MGI geneId's for all genes which are (a) associated with the article according to the MGI database, and (b) mentioned in the abstract, as determined by an automated procedure based on the gene synonym list.[1] We also annotated the evaluation-data for NER evaluation.

We used two closely related gene-protein NER systems in our experiments, both trained using Minorthird (Min, 2004) on the YAPEX corpus (Franzén et al., 2002). The *likely-protein extractor* was designed to have high precision and lower recall, and the *possible-protein extractor* was designed to have high recall and lower precision. As shown in Table 1, the likely-protein extractor performs well on the YAPEX test set, but neither system performs well on the mouse evaluation data—here, they perform only comparably to exact matching against the synonym dictionary. This performance drop is typical when learning-based NER systems are tested on data from a statistical distribution different from their training set.

As a baseline for geneId-ranking, we used a string similarity metric called *soft TFIDF*, as implemented in the SecondString open-source software package (Cohen and Ravikumar, 2003), and soft-matched extracted gene names against the synonym list. Table 2 shows the *mean average precision* on the evaluation data. Note that the geneId ranker based on possible-protein performs statistically significantly better[2] than the one based on likely-protein, even though possible-protein has a lower F score.

To combine these two NER systems, we represent all information as a labeled directed graph which in-

[1]The training data and evaluation data are subsets of the BioCreatIvE "devtest" set. The historical data was called "training data" in the BioCreatIvE publications. The test data is the same as the blind test set used in BioCreatIvE.

[2]With $z = 3.1$, $p > 0.995$ using a two-tailed paired test.

	Precis.	Recall	F
mouse eval			
likely-prot	0.667	0.268	0.453
possible-prot	0.304	0.566	0.396
dictionary	0.245	0.439	0.314
YAPEX test			
likely-prot	0.872	0.621	0.725
YAPEX system	0.678	0.664	0.671

Table 1: Performance of the NER systems on the mouse evaluation corpus and the YAPEX test corpus.

	Mean Average Precision (MAP)
mouse evaluation data	
likely-prot + softTFIDF	0.450
possible-prot + softTFIDF	0.626
graph-based ranking	0.513
+ extra links	0.730
+ extra links & learning	0.807

Table 2: Mean average precision of several geneId-ranking methods on the 50 abstracts from the mouse evaluation dataset.

cludes the test abstracts, the extracted names, the synonym list, and the historical data. We then use *proximity in a graph* for ranking. The graph used is illustrated in Figure 1. Nodes in this graph can be either *files*, *strings*, *terms*, or *user-defined types*. Abstracts and gene synonyms are represented as *file* and *string* nodes, respectively. Files are linked to the *terms* (i.e., the words) that they contain, and terms are linked to the files that contain them.[3] File nodes are also linked to *string* nodes corresponding to the output of an NER system on that file. (*String* nodes are simply short files.) The graph also contains *geneId* nodes and *synonym* string nodes created from the dictionary, and for each historical-data abstract, we include links to its associated geneId nodes.

Given this graph, gene identifiers for an abstract are generated by traversing the graph away from the abstract node, and looking for *geneId* nodes that are "close" to the abstract according to a certain proximity measure for nodes. Similarity between two nodes is defined by a *lazy walk process*, similar to PageRank with decay. The details of this are described in the full paper and elsewhere (Minkov et al., 2006). Intuitively, however, this measures the similarity of two nodes by the weighted sum of all paths that connect the nodes, where shorter paths will be weighted exponentially higher than longer paths. One consequence of this measure is that information associated with paths like the one on the left-hand side of the graph—which represents a soft-match between a likely-protein and a synonym—can be reinforced by other types of paths, like the one on the right-hand side of the figure.

[3]In fact, all edges have inverses in the graph.

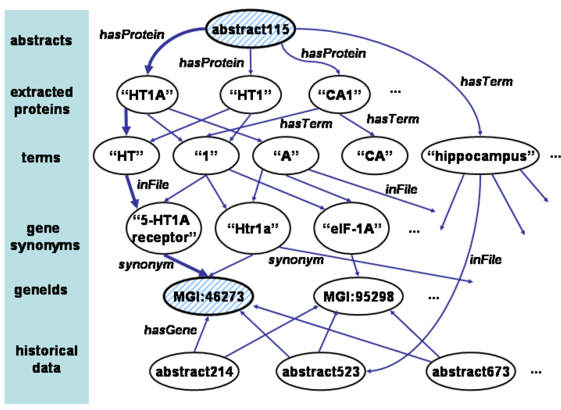

Figure 1: Part of a simplified version of the graph used for geneId ranking.

As shown in Table 2, the graph-based approach has performance intermediate between the two baseline systems. However, the baseline approaches include some information which is not available in the graph, e.g., the softTFIDF distances, and the implicit knowledge of the "importance" of paths from an abstract to a synonym via an NER-extracted string. To include this information, we inserted extra edges labeled *proteinToSynonym* between the extracted protein strings x and comparable synonyms y, and also "short-cut" edges in the graph that directly link abstracts x to *geneId* nodes reachable via one of the "important" paths described above.

As Table 2 shows, graph search with the augmented graph does indeed improve MAP performance on the mouse evaluation data: performance is better than the simple graph, and also better than

	MAP	Avg Max F
mouse test data		
likely-prot + softTFIDF	0.368	0.421
possible-prot + softTFIDF	0.611	0.672
graph-based ranking	0.640	0.695
+ extra links & learning	0.711	0.755

Table 3: Mean average precision of several geneId-ranking methods on the 250 abstracts from the mouse test dataset.

either of the baseline methods described above.

Finally we extended the lazy graph walk to produce, for each node x reached on the walk, a feature vector summarizing the walk. Intuitively, the feature vector records certain features of each edge in the graph, weighting these features according to the probability of traversing the edge. We then use a learning-to-rank method (Collins and Duffy, 2002) to rerank the top 100 nodes. Table 2 shows that learning improves performance. In combination, the techniques described have improved MAP performance to 0.807, an improvement of nearly 80% over the most natural baseline (i.e., soft-matching the dictionary to the NER method with the best F measure).

As a final prospective test, we applied these methods to the 250-abstract mouse test data. We compared their performance to the graph-based search method combined with a reranking postpass learned from the 100-abstract mouse training data. The performance of these methods is summarized in Table 3. The somewhat lower performance is probably due to variation in the two samples.[4] We also computed the maximal F-measure (over any threshold) of each ranked list produced, and then averaged these measures over all queries. This is comparable to the best F1 scores in the BioCreatIvE workshop, although the averaging for BioCreatIvE was done differently.

3 Conclusion

We evaluate several geneId-ranking systems, in which an article is associated with a ranked list of possible gene identifiers. We find that, when used in the most natural manner, the F-measure performance of an NER systems does not correlate well with MAP of the geneId-ranker based on it: rather, the NER system with higher recall, but lower overall performance, has significantly better performance when used for geneId-ranking.

We also present a graph-based scheme for combining NER systems, which allows many types of information to be combined. Combining this system with learning produces performance much better than either NER system can achieve alone. On average, 68% of the correct proteins will be found in the top two elements of the list, 84% will be found in the top five elements, and more than 90% will be found in the top ten elements. This level of performance is probably good enough to be of use in curation.

[4]For instance, the test-set abstracts contain somewhat more proteins on average (2.2 proteins/abstract) than the evaluation-set abstracts (1.7 proteins/abstract).

Acknowledgement

The authors with to thank the organizers of BioCreatIvE, Bob Murphy, Tom Mitchell, and Einat Minkov. The work described here is supported by NIH K25 grant DA017357-01.

References

William W. Cohen and Pradeep Ravikumar. 2003. SecondString: An open-source Java toolkit of approximate string-matching techniques. Project web page, http://secondstring.sourceforge.net.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the ACL*.

Kristofer Franzén, Gunnar Eriksson, Fredrik Olsson, Lars Asker Per Lidén, and Joakim Coster. 2002. Protein names and how to find them. *International Journal of Medical Informatics*, 67(1-3):49–61.

Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. 2005. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6(S1).

2004. Minorthird: Methods for identifying names and ontological relations in text using heuristics for inducing regularities from data. http://minorthird.sourceforge.net.

Einat Minkov, William Cohen, and Andrew Ng. 2006. A graph framework for contextual search and name disambiguation in email. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval*, August. To appear.