Dense Vector Representations of Proteins Based on Gene Ontology Annotations Using Word and Sentence Embedding Tools

JAMES ZHANG[1], Chelsea Ju[2], Muhao Chen[2], Dat Duong[2], Yunsheng Bai[2], Wei Wang[2]

1 BIG Summer Program, Institute for Quantitative and Computational Biosciences, UCLA

2 Department of Computer Science, UCLA

1. Introduction

Recent advances of machine learning algorithms and accumulation of large datasets make possible a wide assortment of new predictive tasks in biological and medical settings. In particular, accurate prediction of protein-protein interactions (PPIs) is vital to our understanding of mechanistic pathways within the human body as well as drug design.

However, in order to be analyzed via computational means, proteins must first be embedded into numerical vectors. Recently, the Gene Ontology (GO), a database of biological terms arranged in a hierarchical graph structure, has been considered a potential source of extracting these vector representations of proteins.

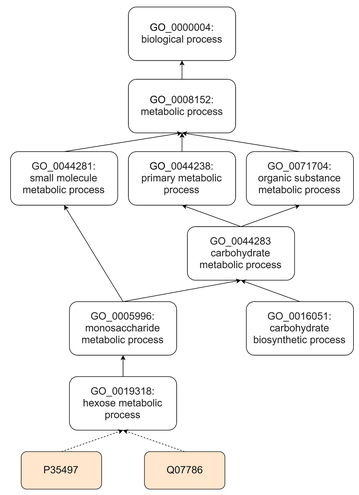
Here, we utilize two natural language processing (NLP) methods to embed proteins into dense vector representations. In the first method, dubbed Onto2Vec[4], the structure of the GO graph, with accompanying protein annotations, is described in a series of sentences. Word embeddings of each protein are generated using Word2Vec. In the second method[2], sentence embeddings of GO term definitions are inputted into a graph convolutional network (GCN) trained on entailment relationships of GO terms. Protein embeddings are calculated from GO term embeddings taken from the embedding layer.

Our code is available at our GitHub repository:
https://github.com/jameszha/ProteinEmbedding

2. The Gene Ontology

The GO database is comprised of nearly 50,000 GO terms that each describe biological functions of genes. The GO is organized into a hierarchical tree structure, with each GO term occupying a node and each superclass/subclass relationship forming an edge.

Proteins may be annotated to one or more of these GO terms. Our methods seek to capture the structure of the GO within each protein embedding.



3. Methods

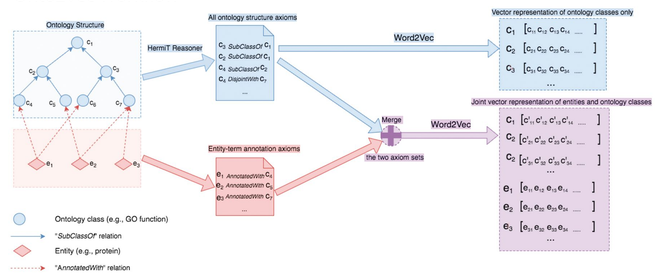
The datasets used are all publicly available online. The GO (2019-07-01 version) was downloaded in .owl format from the OBO Foundry Permanent URLs (PURLs). Yeast protein annotations were downloaded in .gaf format from the Gene Ontology Annotation (GOA) Database. The binary PPI dataset used was the benchmark yeast PPI dataset[1], which is widely used in many modern methods. PPI types were downloaded from the STRING database. EC numbers were downloaded from ExPASy.

As a baseline, sparse binary embeddings were generated from the protein annotations simply by setting each vector element to 1 if the protein was annotated to the corresponding term or to 0 otherwise.

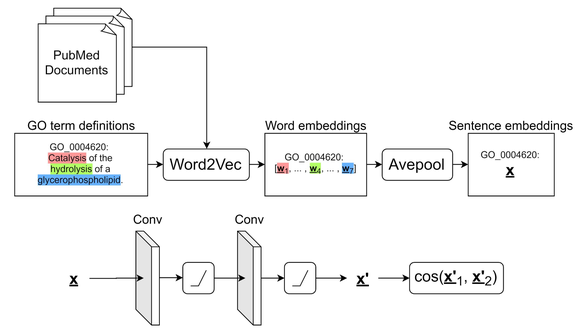
The Onto2Vec and GCN method workflows are depicted to the right. Notably, the GCN method produces GO term embeddings. Protein embeddings were taken to be the sum or mean of the term embeddings to which the proteins were annotated. A similar process was also repeated for Onto2Vec term embeddings for sake of comparison.

The embeddings were evaluated in three different tasks (multiclass Enzyme Commission (EC) number classification, binary PPI prediction, and multiclass PPI type classification), using four classifiers from the scikit-learn Python package: Logistic Regression, Random Forest (default sklearn hyperparameters), SVM (linear kernel, C=1), and MLP (hidden layers of 800 and 200).

Onto2Vec Workflow



GCN Workflow



4. Results

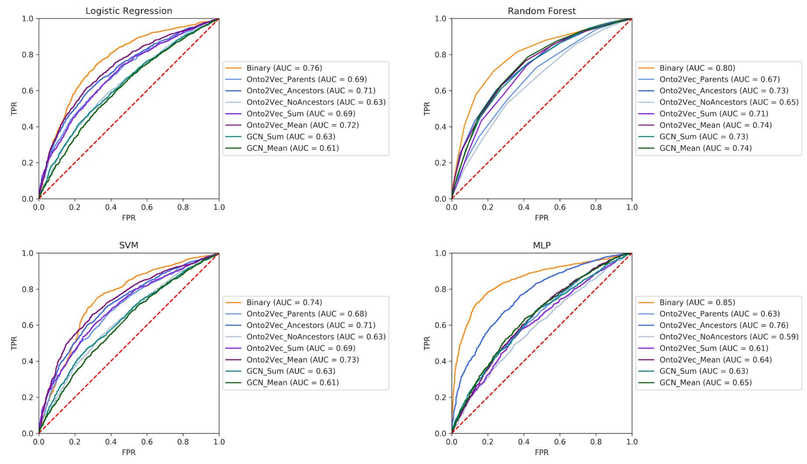
The binary embeddings generated were of dimension 5609. The dense embeddings generated via the other methods were all of dimension 200. From the datasets used, we generated
938 training samples and 402 test samples for EC classification, 7518 training samples and 3222 test samples for binary PPI prediction, and 7662 training samples and 3284 test samples for PPI types classification.

We found that for multiclass classification of single proteins into top-level EC categories, Onto2Vec embeddings with annotation propagation up the entire GO performed very well across each classifier. The ROCs achieved AUCs of up to 0.92 with an MLP classifier and matched or surpassed those of the sparse binary embeddings in all classifiers types.

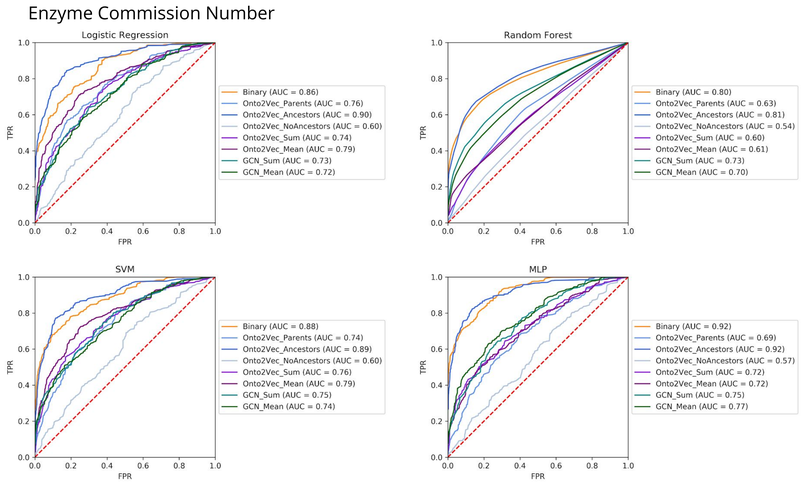
We found that the dense embeddings were less successful in protein-protein interaction tasks, performing worse than the binary embeddings, especially in binary protein-protein interaction prediction. Of the dense embeddings, Onto2Vec with propagation up the entire GO with an MLP classifier performed best, achieving an ROC AUC of 0.76. However, this is significantly lower than the ROC AUC using binary embeddings with an MLP classifier of 0.85.

In general, protein embeddings generated from taking the sum or mean of GO term embeddings to which they are annotated performed the most poorly. As a result, the GCN methods were largely unsuccessful.

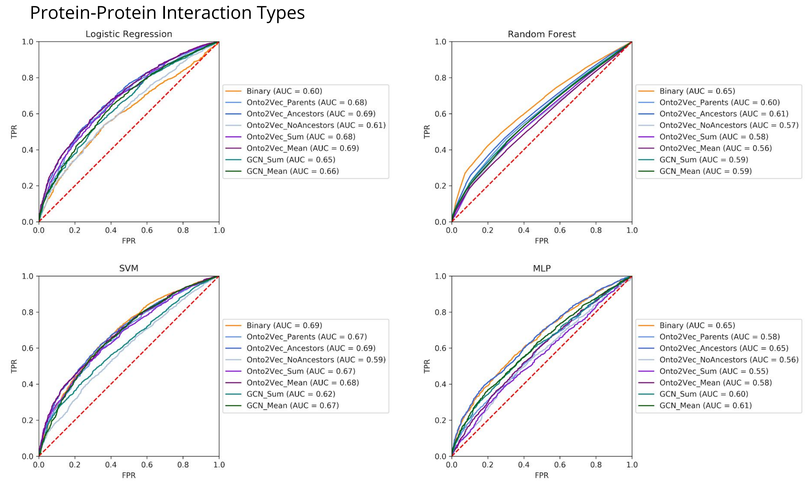
Binary Protein-Protein Interaction



Enzyme Commission Number



Protein-Protein Interaction Types



5. Discussion

We explored several methods of embedding proteins into dense vector representations utilizing NLP word and sentence embedding tools.

Despite failure of the dense protein embeddings to match the performance of the sparse binary embeddings in PPI tasks, we hypothesize that dense embeddings may show promise in situations of data scarcity. Due to the high dimensionality of the binary embeddings, a similarly high training sample set size is required. When training data is limit, as was the case for EC number classification, our dense embeddings outperformed the sparse binary embeddings. To expand on this project, classifier performance may be examined at varying training set sizes.

Furthermore, two advantages of dense embeddings we observed are storage requirements and classifier runtime. Higher dimensionality naturally requires proportionally greater storage space and dramatically increases runtime. Further work is needed to quantify these tradeoffs.

Future work may also involve applying GCN methods to a supergraph containing the GO tree along with protein nodes and annotation edges. In particular, recently developed methods like GraphSAGE, which produces node-level embeddings, may serve to yield protein embeddings directly, without need for taking the sum or mean of GO term embeddings.

6. References

1. Chen, M., Ju, C. J., Zhou, G., Zhang, T., Chen, X., Chang, K., . . . Wang, W. (2018). Lasagna: Multifaceted Protein-Protein Interaction Prediction Based on Siamese Residual RCNN. doi:10.1101/501791
2. Duong, D., Ahmad, W. U., Eskin, E., Chang, K., & Li, J. J. (2017). Word and sentence embedding tools to measure semantic similarity of Gene Ontology terms by their definitions. doi:10.1101/103648
3. Hamilton, W. L., Ying, R., & Leskovec, J. (2018). Inductive Representation Learning on Large Graphs. Doi:1706.02216
4. Smaili, F. Z., Gao, X., & Hoehndorf, R. (2018). Onto2Vec: Joint vector-based representation of biological entities and their ontology-based annotations. Bioinformatics, 34(13), I52-I60. doi:10.1093/bioinformatics/bty259