Community Detection and Link Prediction for Visual Genome Image Object-to-Object Relationships

Michelle Lam[1], Vivian Nguyen[1]

[1]Stanford University, Department of Computer Science

Abstract

This project explores various network representations of the Visual Genome annotated image dataset. The relational structure of the items within the dataset's images lends itself well to a network representation consisting of three node types—*entities*, *predicates*, and *attributes*—and we perform community detection and relationship / link prediction on this tripartite graph. Relationship prediction aims to identify whether an edge exists between two nodes; to this end, we developed feature vectors based on node properties and proximity measurements between nodes, and we applied four different machine learning models (SGD, logistic regression, SVM, and random forest). Furthermore, we incorporated Wikipedia corpus textual features and GloVe vectors into our community detection and link prediction tasks, and we found that the addition of text data was advantageous for both of these tasks. Overall, we achieved high accuracy, precision, and recall (around 90% or higher) for a significant portion of the link prediction, with the highest accuracy at 99.2%.

Introduction

Problem/Motivation

For our project, we propose to investigate Visual Genome – a densely-annotated image dataset – as a network connecting objects and attributes to model relationships. This dataset in its original form can be visualized as a graph network and thus lends itself well to graph analysis. In addition, we would like to augment the analysis of this graph by incorporating text features from the Wikipedia corpus and GloVe vectors derived from this corpus. These natural language features can be added as node attributes to enrich the object nodes in community detection, and similarities between the GloVe word vectors can be used to inform relationship prediction between object nodes. Further, since Visual Genome is an image-derived knowledge base and Wikipedia is a text-centric knowledge base, these two modes may complement each other to unearth new patterns in object relationships and attributes. Visual Genome is also rich in relationship (edge) information, but poor in object (node) information, so the addition of Wikipedia text features can help to extend the object (node) information while potentially providing additional relational information.

First, we would like to explore the *clusters* or *"communities"* present within object networks. Given a set of objects and their relations with respect to other objects in Visual Genome, how might we discover latent topical or thematic clusters (ex: food, transportation, zoo)? How are objects commonly associated with attributes and actions and activities in this large image base?

Next, we would like to investigate whether we can *predict relationships* between objects. Given a set of objects in a source image, can we predict (1) *which* objects have a relation and (2) *how* these objects are related?

Related Work

In their 2013 work, Yang et al. investigate the community detection problem through a new lens by identifying a gap in the prior work in traditional community detection algorithms and clustering algorithms: while communities and graphs seem to intuitively arise from both the network structure and the attributes of the nodes themselves, community detection algorithms are largely concerned with the network structure, and clustering algorithms typically focus on the node attributes [13]. While it is challenging to combine these two sources of information to determine communities in networks, it is important to consider both of these aspects to make accurate and robust classifications. To achieve this goal, this work introduces the Communities from Edge Structures and Node Attributes (CESNA) model for community detection on networks with node attribute information. Since the model accounts for the dual factors of network structure and node attributes, it results in improved accuracy for community detection, and the presence of node information also aids in the subsequent interpretation of these communities.

This paper presents very exciting implications for the accuracy, scalable performance, and interpretability of community detection algorithms. However, it would be nice to extend this work to support additional attribute types beyond binary-valued attributes to model more complex knowledge about a given node.

Meanwhile, the domain of image clustering could also benefit from community detection techniques. Papadopoulos et al., driven by the information-overload present on photo-sharing applications, investigate the use of community detection to cluster similar images into visually- and conceptually-cohesive groups [6]. Existing work in image clustering involved high computational cost, and since many techniques were supervised, they weren't well-suited to unlabeled data such as user-contributed images. Thus, they turned to the graph-based method of community detection, which is efficient; this method also only groups together images that are related to each other such that outliers are not included, which improves the resulting cluster precision. Visual and text-based image features were used to determine the weight of edges between node images (these features were tested independently and in combination). Visual features included SURF local feature descriptors, histogram intersections, and affine transformation estimates; the text features were generated using tag co-occurrence. They found that this method was effective in generating clusters that had geospatial coherence and semantic relevance.

While this work presented a novel approach to the image-clustering domain using graph-based community detection algorithms, it could be improved with the additional use of node attributes. The application of the community detection algorithm here makes use of the relations between images (network structure), but does not incorporate the core properties of each of the individual images that comprise a potential community/cluster. Perhaps by using a hybrid model such as CESNA this work could improve the accuracy of its images clusters.

Recognizing that domains in the real world are often relational, meaning that entities can be different types and have different attributes, Taskar et al. set out to predict what types of links would exist between objects within these domains [10]. In an earlier paper by Taskar et al., it was pointed out that data is not "flat" and thus realistically should not always have the same attributes [9]. This fact is particularly relevant with our project given that we are working with image objects with unique atttributes. Taskar et al. used the strategies of a relational Markov network (RMN) framework and the application of probabilistic models to the link graph, and the results were significantly more accurate than through flat classification. After testing on web data obtained from three different universities and on social network data, two meaningful patterns within subgraphs emerged: similarity between entities and links that share a graph property, and transitivity which relates trios of entities and links that are organized into a triangle.

A drawback of this paper is that the work relied on running probabilistic inference on dense Markov networks, thereby making it difficult to attain an accurate inference. Therefore, Taskar et al. had suggested in a previous paper the usage of belief propagation, which is a method used to approximate the gradient for gradient descent, but even this method presents a problem when the graph becomes complex because it will not converge. Thus, while this paper presents a very unique idea that incorporates relational data into the graph structure, there is room for improvement. Despite this, though, the idea of different types of entities and links will prove to be useful in our project because it more closely resembles the natural world. Within our daily lives, there are many different types of entities and links between them, and thus this paper allows us to begin to think about how we could represent our dataset as more than one type of node with many different possible relationships between them.

Data

Visual Genome

We will be using the Visual Genome dataset which consists of 108,077 images along with all of their content annotations [2]. These annotations have three different types: 3.8 million **object** (entity) instances (e.g. *person, building, table*), 2.8 million **attributes** (e.g. *blue, red, large*), and 2.3 million **relationships** (predicates) (e.g. *wearing, behind, in front of*). The object-to-object relationships alone total 1,531,448 instances. In the original dataset, each image can be represented as a scene graph in which objects, attributes, and relationships are connected based on region descriptions (sentences that describe a particular region of the image). For clarity, we refer to what Visual Genome calls "objects" as "entities" (so as to distinguish them from "objects" in the grammatical sentence-structure sense). Thus, there are three main types of relationships that are derived from the region description sentences: (1) entity $\leftrightarrow$ attribute, (2) subject $\rightarrow$ predicate, and (3) predicate $\rightarrow$ object.

Therefore, we will be representing entities (objects), attributes, and relationships (predicates) as three different types of nodes within the graph, creating a tripartite graph. Directed edges between them will represent the three above associations between the different node types.

Wikipedia

We then gathered data on Wikipedia articles from the *Wikispeedia* dataset, which includes lists of Wikipedia article titles, article category hierarchies, article links, and the full

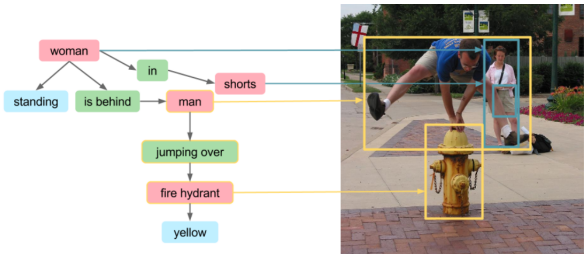

Figure 1: Example of graphical representation of an image. Image from Krishna et al.

text of each article [11]. This dataset has over 4,600 articles and over 119,000 links between articles. This data was collected in order to augment the information at each of our entity nodes with additional textual features. We also gathered word vectors (from GloVe) that had been pre-trained on the Wikipedia dataset to further enrich the information at entity, attribute, and relationship nodes.

Method

Constructing the Graph Network

We first must reformulate the data so that it resembles a directed graphical network. We will visualize this as a tripartite graph with three types of nodes: *entity* nodes, *predicate* nodes, and *attribute* nodes. The original data is represented as two JSON files: one listing relationships between entities per image and one listing attributes of each entity per image. The relationships listed form the set of predicate nodes for the graph, and every predicate can have a subject entity and an object entity. This directed relationship between entity and predicate nodes is represented through directed edges. This type of linear relationship doesn't exist between attributes and entities, so we represent the edges between attributes and entities as bidirectional edges.

A problem that we took into account when constructing the graph network was that the data consists of over 100,000 images. There would be a lot of nodes that would be repeated; for instance entities like "tree" or "sky" would be a common occurrence. Thus, we implemented our own system of only keeping track of unique nodes, each with its own unique id. Furthermore, each node has an attribute *Type*, which indicates whether it's an *entity*, *predicate*, or *attribute*. Because the graph represents a large number of images, many of which will have similar characteristics, a multigraph was formed. Each node can have multiple edges between them due to the scale of the image base.

The resulting graph consists of 221,318 nodes, of which 103,718 are *entity* nodes, 37,342 are *predicate* nodes, and 80,258 are *attribute* nodes. The graph has highly similar in- and out-degrees for the three node types as shown in Table 1.

Node Type	In-Deg	Out-Deg
Entity	44.9	44.9
Predicate	62.0	62.0
Attribute	29.2	29.2

Table 1: In- and Out-Degrees for the Visual Genome Graph Node Types

Community Detection

For the community detection task, we aim to identify overlapping communities among the nodes of our tripartite graph.

Text features While we were interested in communities that might arise from the Visual Genome graph structure alone, we also felt that the nodes themselves lacked some richness. We thus turned towards textual datasets to augment our existing network; we decided on the Wikipedia dataset because it provided a large text corpus and a variety of well-defined entities (article topics). Using this dataset, we experimented with two text-based features to augment our node attributes for the prior community detection methods.

- **GloVe** We use GloVe (Global Vectors) trained on the Wikipedia dataset as node attributes for any nodes in the network for which we have a word vector [8].The GloVe model is trained on a global word-word cooccurrence matrix such that it generates a vector space with meaningful substructure; this enables the word vectors to excel in tasks such as analogies and word similarities. Since we aim to bridge the *visual-spatial, image-based* relationships between entities and attributes with a *semantic, language-based* taxonomy of the world, GloVe seemed very fitting as a method to increase our language-oriented insights in community detection.

- **TF-IDF Vectors** In order to better utilize the encyclopedic information available for entities that are article topics, we also generated TF-IDF (term frequency–inverse document frequency) vectors for all Wikipedia articles from the *Wikispeedia* dataset. These vectors encode the degree of importance of words to each document in a corpus.

Community Detection Algorithms We investigated several community detection algorithms to compare their effectiveness on different graph formulations.

- **Link Clustering** We first wanted to investigate a community detection method based on network links. Ahn et al. find that many communities display hierarchical structures, but often they have so many overlapping communities that it becomes difficult to determine community

structure using global hierarchies [1]. Instead, they use link communities to encapsulate this overlap and at the same time reveal hierarchical structures. This algorithm first uses Jaccard similarity to determine the similarity between link pairs that share a node. Then, hierarchical clustering is performed based on this single-link hierarchy, which creates a dendrogram of links. The dendrogram is then cut according to thresholds (like the maximum partition density D, shown below) to produce final link communities.

$$D = \frac{2}{M} \sum_c m_c \frac{m_c - (n_c - 1)}{(n_c - 2)(n_c - 1)}$$

where $C =$ the number of subsets formed by the partition, $M =$ the total number of links in the graph, $m_c =$ number of links in a partitioned subset of the graph, and $n_c =$ number of nodes touched by those links.

- **BIGCLAM** Next, we applied the BIGCLAM (Cluster Affiliation Model for Big Networks) algorithm to our community detection task. This algorithm is scalable to very large networks and is built upon the insight that overlaps between communities are often more densely connected than non-overlapping parts; it follows from this observation that the more communities shared between a pair of nodes, the higher the likelihood that these nodes are connected [12]. The algorithm creates a bipartite affiliation network that connects nodes to the communities to which they belong; the aim is to find an affiliation factor matrix $\hat{F}$ that is most likely for the underlying network G.

$$\hat{F} = arg \max_{\geq 0} l(F)$$

where

$$l(F) = \sum_{(u,v) \in E} log(1 - exp(-F_u F_v^T)) - \sum_{(u,v) \notin E} F_u F_v^T$$

- **Link Clustering extended with textual similarity features** We then built upon the link clustering method by incorporating text-based features into the algorithm's similarity metric. We chose to augment the link clustering method rather than the node clustering method because our vector-based text features were amenable to distance calculations and could directly build upon similarity comparisons between nodes (as are performed for link clustering). For each node, we retrieved word vectors and TF-IDF vectors. Then, depending on which vector was used to augment the community detection method, we calculated the cosine similarity between the vectors belonging to nodes connected by an edge, as shown below.

$$cos(a, b) = \frac{A \cdot B}{||A||_2 ||B||_2}$$

This score is linearly combined with the standard similarity metric for the link clustering algorithm, Jaccard Similarity.

Relationship Prediction

The goal of relationship prediction is to determine whether two nodes, u and v, in graph $G = (V, E)$ would have a connecting edge. If (u, v) have a connecting edge, then $(u, v) \in E$; otherwise, $(u, v) \notin E$. In order to determine whether (u, v) contains an edge, we use (1) the nodes' characteristics provided to us by the inherent graph structure and (2) the proximity metrics between the two nodes. These values thus form rich feature vectors that can represent the nodes in machine learning models.

Before applying machine learning methods, the graph data needs to first be divided into two sets: the training set (X_{train}: a set of node pairs represented as feature vectors, and y_{train}: the corresponding labels where 0 means $(u, v) \notin E$ and 1 means $(u, v) \in E$) and the test set (X_{test} and y_{test}). Thus, we assigned the data so that the training set consists of 20,000 node pairs: 10,000 with an edge between them and 10,000 without. The test set consists of 2,000 node pairs: 1,000 with an edge between them and 1,000 without.

After defining our training and test sets, the next steps are to compute the feature vectors and then to train various models on the training set in order to perform the link predictions on the test set.

Feature Selection There are many possibilities for which features to include in the feature vectors. An option is to consider properties of the nodes in the network, such as the degrees of the nodes. The work of Liben-Nowell et al. addresses the link prediction problem by measuring the proximity of the nodes through various metrics, such as the number of common neighbors between the two nodes [3]. Thus, we decided to approach our problem by comparing two "types" of feature vectors: (1) node-based properties and (2) proximity-based measurements.

For the node-based feature vectors, we assigned one of three values to each of the nodes in the pair (u, v) depending upon what type of nodes they are: *entity*, *predicate*, or *attribute*. We also computed and summed each of the node's in-degree and out-degrees. An interesting approach we also made to solving our problem of link prediction on an image-based dataset was incorporating in text-based features. We use GloVe vectors to represent entities that are tagged in the images, and for each pair of entities in our training or test set, we compute a cosine-similarity score to be included in the feature vectors. We'll elaborate more on this technique and how we use it in the following *Results* section.

For the proximity-based feature vectors, we computed various values used by Liben-Nowell et al. [3]:

- **Number of common neighbors** This is the number of neighbors that are shared by the two nodes in the node

pair.

$$|\Gamma(u) \cap \Gamma(v)|$$

- **Adamic/Adar score** This score favors common neighbors of the node pair that have a lower degree. The reasoning is that nodes with lower degrees are more rare, and thus, if both of the nodes we're considering have it as a common neighbor, then it is more meaningful than if they were to share a neighbor connected to a lot of other nodes. It's computed as follows:

$$\sum_{z \in |\Gamma(u) \cap \Gamma(v)|} \frac{1}{log(\Gamma(z))}$$

- **Jaccard's Coefficient** It is more commonly referred to as the intersection over the union, and it can be computed as follows:

$$\frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}$$

- **Preferential Attachment score** The idea of this score is that a node that has more neighbors has a greater probability of being an endpoint in a future edge. This value is:

$$|\Gamma(u)| * |\Gamma(v)|$$

- **PageRank score** This is a measure that evaluates a node's importance based on its links. The idea is that a more important node would receive more links from other important nodes.

Models After computing the feature vectors, we can use them in the following models in order to achieve link prediction.

- **Stochastic Gradient Descent (SGD) Classifier** This model requires that we continuously loop through the training data, and for each training example, the algorithm's parameters are updated based on the gradient of the error of that example. The update would thus be written as follows [4]:

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^(i)))x_j^{(i)}$$

  for every j where $j = 0, ..., n$ and n is the number of features and $i = 0, ..., m$ where m is the number of values in the training set.

- **Logistic Regression** Using a logistic regression makes sense in this problem because we're classifying edges between two nodes as existent (1) or nonexistent (0). Thus, the values for y are between 0 and 1. We can therefore think of the prediction for y as following this formula for the logistic function:

$$\frac{1}{1 + e^{-\theta^T x}}$$

- **Support Vector Machine (SVM)** SVMs are a commonly used supervised learning algorithm, and this classifier can be written as [5]:

$$y = g(w^T x + b)$$

  where $y = 1$ if $w^T x + b \geq 0$. The value of b is θ_0, and w is $[\theta_1 ... \theta_n]^T$.

- **Random Forest** The random forest averages across ten decision trees modeled on sub-samples of the dataset [7].

Results

Community Detection

We investigated three approaches for community detection, as mentioned in the *Approach* section.

- *Link clustering* using the Link Community detection algorithm [1]
- *Node clustering* using the BIGCLAM algorithm [12]
- *Link clustering extended with textual similarity features*

Because of the large size of the original Visual Genome tripartite graph, we first reduced the network to only nodes for which there existed a Wikipedia article. This limited the graph to largely *entity* nodes and to proper, encyclopedic sorts of entities. Using this first formulation, which we call the **vg-wiki-graph**, there were 357 nodes and 1266 edges; there are 16 attribute nodes, 112 predicate nodes, and 229 entity nodes.

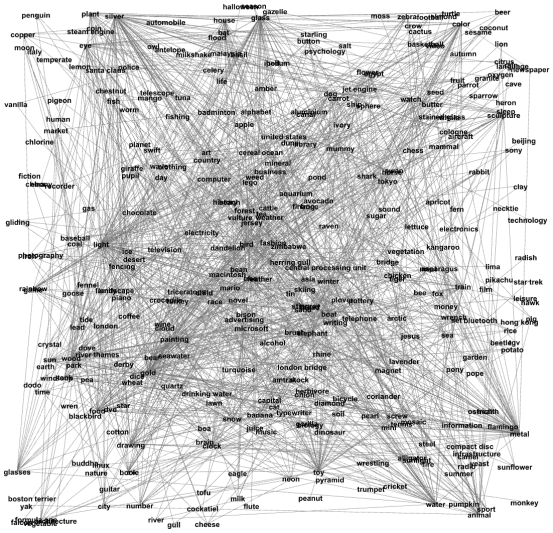

Figure 2: vg-wiki-graph

Then, to investigate the graph structure in a more balanced manner while evaluating a smaller-size network,

we selected a random sample of 10% of the nodes in the Visual Genome tripartite graph and only preserved the edges between these selected nodes. Using this formulation, which we call the **vg-rand-graph**, there were 3971 nodes and 6977 edges. Of these nodes, there were 1323 attribute nodes, 982 predicate nodes, and 1666 entity nodes.

For each of these graph types, we perform community detection using both Link Clustering and BIGCLAM. In general, we find that the BIGCLAM method appears a bit more successful upon *qualitative* evaluation than the Link Clustering method (See Tables 10-13 for summaries of each method for these two graph formulations). Since there is no ground truth community for this dataset, we evaluate the coherence of the identified communities using GloVe word vectors as a proxy for the relatedness of the community members. We retrieve word vectors for nodes in each community, find the centroid for each community, and calculate the average cosine similarity between member nodes and the centroid. Then, we calculate the average similarity across the communities for each community detection method to compare relative success. These results can be found in Table 2.

Next, we perform community detection with the extension of *textual similarity features*. Since our text features are based on Wikipedia data, which are heavily skewed toward nodes of the *entity* type, we created a third graph formulation for this task, which we'll call **vg-entity**. This graph consists of only *entities*, and edges are added between entities that are connected by a *predicate* in the Visual Genome graph. We then apply the *Link clustering + text features* methods described in the *Approach* section to generate communities for this modified graph structure, and we find that the similarity of the words in these clusters is noticeably higher than that of the original methods (See Table 3). See Tables 4 and 5 to view samples of communities detected by this text-enhanced link clustering community detection approach.

Algorithm	**vg-wiki**	**vg-rand**
BIGCLAM	0.587	0.530
Link clustering	0.599	0.555

Table 2: **Tripartite graphs** – Average cosine similarity of words in detected communities

Algorithm	**vg-entity**
Link clustering + TF-IDF	0.694
Link clustering + GloVe	0.723

Table 3: **Entity-only graph** – Average cosine similarity of words in detected communities

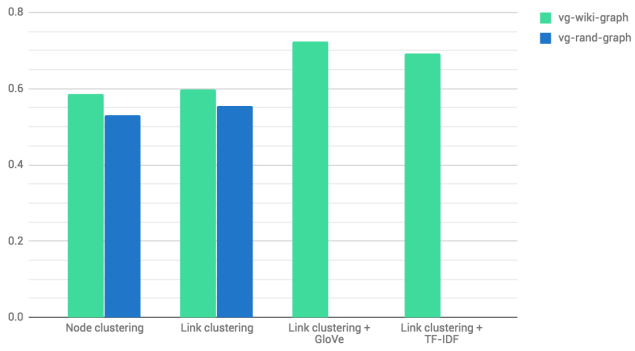

Figure 3: Average cosine similarity of words in detected communities

CID	Nodes
1	fruit, celery, cheese, bean, soup, vegetable, bread, apple, rice, carrot, butter, parsley, food, meat, lettuce, chicken, banana, onion, potato, pea
2	river, lawn, bird, vegetation, goose, forest, pond, water
3	tree, hyena, lion, elephant, vegetation, cheetah, antelope, gazelle, plant, giraffe, animal, water, zebra
4	computer, information, book, television
5	dog, tree, cat, flower, plant, bird, clock, glass, animal, sunlight, wood, light, child, house, television

Table 4: Sample of communities for **vg-entity, Link Clustering + tfidf**

CID	Nodes
1	toy, clock, nintendo, binoculars, game, house, house, television, computer, star wars, family
2	river, wave, ship, ocean, sea, beach, sand, boat, train, bridge, water
3	meat, celery, lemon, soup, vegetable, bread, chicken, rice, carrot, fish, butter, parsley, food, oregano, lettuce, juice, potato, onion, basil, chives, cheese
4	pigeon, sculpture, park, lawn, metal, plant, wood, light, bicycle, flower, bird, glass, animal, horse, tree, city, dog, garden, sand, cat, forest, train, bridge, water
5	metal, sculpture, button, fencing, bicycle

Table 5: Sample of communities for **vg-entity, Link Clustering + GloVe**

Relationship prediction

As mentioned in the *Approach* section of the paper, we consider two types of feature vectors: (1) individual node-based

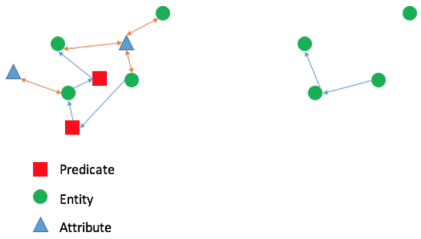

Figure 4: An illustration of the original tripartite graph (left) and entity-only graph (right)

features and (2) proximity-based features. In addition to comparing these two feature vectors though, we also generated an additional entity-only graph. This entity graph consists only of nodes that are entities, and two entities are linked by an edge if they are related to one another by a predicate in our original tripartite graph (See Figure 4).

The reason we created this additional graph is because we wanted to incorporate text-features into our link prediction. In particular, it makes sense for entities (as opposed to predicates such as "underneath" or "on top of") to have meaningful GloVe vector representations. Thus, we can incorporate these representations into a cosine-similarity score between the two nodes in each node pair of our training and test sets. Because these scores both take into account each individual node's characteristics and rely also on the other node, we could have included this feature in either of the two types of feature vectors. Ultimately, we chose to include it into the node-based features, though, because that feature vector is more sparse now that we don't need to include the type of node as features, given that all of the nodes in this graph are *entities*. The values computed and included in the proximity-based feature vector remain the same for this entity-only graph. Therefore, we are ultimately viewing the link prediction problem in four different ways:

1. Tripartite graph with node-based feature vectors
2. Tripartite graph with proximity-based feature vectors
3. Entity-only graph with node-based + GloVe-based feature vectors
4. Entity-only graph with proximity-based feature vectors

For each of these, we use the four different models (SGD, Logistic Regression, SVM, and Random Forest) to predict links, and we display accuracy, precision, recall, and F1 scores in Tables 6 - 9.

Overall, we achieve very high accuracy, ranging from 95.5% to 99.2% for each of the best models across the four different graph + feature vector combinations. Similarly, all of the best precision, recall, and F1 scores for each of the combinations all range from the mid- to high-90 percentages.

Model	Accuracy	Precision	Recall	F1
SGD	0.504	0.502	1.0	0.668
Log. Regression	0.515	0.507	**1.0**	0.673
SVM	0.976	0.958	0.996	0.976
Random Forest	**0.978**	**0.987**	0.969	**0.978**

Table 6: Tripartite Graph + Node-based feature vector

Model	Accuracy	Precision	Recall	F1
SGD	0.585	0.546	1.0	0.707
Log. Regression	0.585	0.546	**1.0**	0.707
SVM	0.986	0.976	0.996	0.986
Random Forest	**0.992**	**0.993**	0.991	**0.992**

Table 7: Tripartite Graph + Proximity-based feature vector

Model	Accuracy	Precision	Recall	F1
SGD	**0.955**	0.960	0.95	**0.955**
Log. Regression	0.937	**0.981**	0.891	0.934
SVM	0.953	0.946	**0.961**	0.953
Random Forest	0.929	0.946	0.909	0.927

Table 8: Entities Graph + GloVe-based feature vector

Model	Accuracy	Precision	Recall	F1
SGD	0.732	0.652	0.994	0.788
Log. Regression	0.5	0.5	**1.0**	0.667
SVM	0.956	**0.962**	0.95	0.956
Random Forest	**0.958**	0.961	0.954	**0.957**

Table 9: Entities Graph + Proximity-based feature vector

Discussion

For both community detection and relationship prediction, the approaches we used achieved both good and meaningful results.

For community detection, we saw that node clustering (with BIGCLAM) and link clustering performed similarly using a word-vector-based quantitative similarity evaluation. However, upon qualitative evaluation, node clustering appeared to have better results. This likely occurred because the links in our tripartite graph connect *across* the three node types and thus may be more noisy and less informative than links for a typical graph (which tend to connect nodes of the same type), thus disadvantaging the link-based clustering method. Both methods saw a slightly lower performance for the *vg-rand-graph* than for the *vg-wiki-graph*, which was likely a result of the greater variety of node types in this graph formulation that similarly complicates the process of community detection.

For community detection extended with text features and an entity-only graph, we found that we were able to achieve significant improvements in our community-similarity metric for both approaches, and the improvements were of a similar magnitude for both the entity name-based GloVe approach and the article content-focused TF-IDF approach, indicating that even though the similarity metric is tied to word vectors, textual information in other forms also is highly beneficial for our community detection problem. In all though, community detection was quite a difficult task with this dataset since a great deal of the entities were quite generic (ex: dog, tree, plant), so it was difficult to generate cohesive, defined communities, and it was difficult to objectively evaluate the detected communities beyond measures such as our text-based similarity metric.

For relationship prediction, the overall accuracy, precision, recall, and F1 scores were all consistently very high, but there are still interesting comparisons to be made. In particular, for the original tripartite graph, the results were quite similar regardless of whether we used a node-based or proximity-based feature vector because both perform so well, but the proximity-based feature vector does indeed surpass the node-based one. Intuitively, it makes sense because the proximity-based feature vector takes into account both nodes for each measurement, as opposed to the node-based vector where each value is only associated to a particular node. Another significant result to note is that the random forest model achieves the best accuracy and precision for the tripartite graph for both types of feature vectors. A reason why decision trees perform well is because the relationship between the cost of predicting the correct classification for y-values and the amount of training data provided is logarithmic. Also, they're versatile because they're able to accept both numerical and categorical data [7].

Now, if we look at the entity-only graph, the best-performing model for the proximity-based feature vector still achieves a bit higher accuracy than the best-performing model for that of the node-based + GloVe-based feature vector. Upon closer inspection, though, we see that the node-based + GloVe-based feature vector yields more consistently accurate results. The accuracy across all of the models for the node-based + GloVe-based feature vector is always in the 90s percentages, and thus is a lot higher than that of the accuracy values for some of the models for proximity-based feature vector. This indicates that using text-based features really enriches the feature vector and helps achieve consistently high accuracy regardless of the machine learning model used.

Thus, the results of incorporating the GloVe vectors into link prediction support our initial hypothesis that entities whose names are more similar to one another as words are also more likely to be related, and thus appear in images together.

Conclusion

Not only did we generate multiple graph networks, including a tripartite one with three different types of nodes (*entities*, *predicates*, and *attributes*) and an entity-only one which is connected based on whether a predicate exists between them, but also applied community detection and link prediction algorithms to the graphs. We extended link clustering-based community detection techniques with Wikipedia text features and observed improved community cluster results. For relationship/link prediction, we looked at different types of feature vectors, including one which incorporated text-features based on GloVe vectors, and we used four different machine learning models to predict whether a link exists between any given two nodes. Overall, we achieved high accuracy, precision, and recall across the board, with the highest accuracy being up to 99.2%.

Future Work

Because we found that textual information was highly beneficial to community detection, we would like to push it even further. The CESNA algorithm is designed to use binary features, and we found that our problem did not lend itself well to binary features because of the great diversity of node types and node attributes across the graph. In the future, we might reformulate CESNA to use non-binary features so that we can incorporate these useful textual features, while also making use of graph structure, for a more performant community detection algorithm.

Another interesting approach that we would like to consider next is how to use additional text features for relationship prediction. In our project, we used GloVe vectors in order to compute cosine-similarity scores between two nodes. Another approach could be to consider what links to other entity names are linked from a current entity's Wikipedia page. That would mean that we could use an entirely different network to train on in order to try to predict the relationships in the Visual Genome network.

References

Yong-Yeol Ahn, James P. Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 2010.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016.

David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, pages 1019–1031, 2007.

Andrew Ng. Cs 229 lecture notes 1. *CS 229 Course Website*.

Andrew Ng. Cs 229 lecture notes 3. *CS 229 Course Website*.

CID	Nodes
1	wood, boat, train, metal, writing, light, house, color, set, toy, ship, dog, tree, bridge, book, iron, bear, glass, piano, lego, bird, plant, bat, star, giraffe, sun, cat
2	animal, elephant, eye, bear, bird, giraffe, watch, drawing, sculpture, zebra, dog, child, toy, color, mammal, chocolate, family, light, cat, drinking water, lead, owl, sand, wrestling
3	water, beach, ocean, light, sand, wave, boat, sea, turquoise, tree, sport, summer, house, crystal, forest, bird, tide, plant, pond, skiing, glasses, lavender, color, autumn, swift, glass
4	food, light, plant, vegetable, chicken, bread, soup, meat, flower, market, tree, banana, chocolate, desert, lettuce, glass, wheat, fruit, potato, garden, fish, tuna, butter, juice, giraffe, carrot, rice, weed, fennel, vegetation, lavender
5	apple, mini, drawing, fruit, toy, banana, tree, carrot, juice, plant, lemon, light, computer, rainbow, color, magnet, art, flower, glass, dog, train, cactus, macintosh, citrus

Table 10: Sample of communities for **vg-wiki-graph, BIGCLAM**

CID	Nodes
1	in front with, flat land, wire is hanging, has open red, table, white stalks, intersections, tan jacket, attached on to, enter, keeping people off, bullet shaped, outcropping, number 25, on galloping
2	pedestrian crosswalk, draperies
3	red and blue flower, sandles, green bottle, star pattern, chalice
4	wooden floor, children's playroom
5	seen from, clouds, mountains in distance

Table 11: Sample of communities for **vg-wiki-graph, Link Clustering**

Symeon Papadopoulos and et al. Image clustering through community detection on hybrid image similarity graphs. *ICIP International Conference on Image Processing*, 2010.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

CID	Nodes
1	donuts, assortment, middle, store bought, regular, with white, flat, delicious, covered, in, full of, around, on, side, sprinkles, multple , jelly filled, wreath, yeast, galed, chocolate frosted, organized, Dunkin Donuts, six in total, pastries, frying, chocolate glazed, frying, chocolate frosted, looking at a display of several, white towels
2	aircraft, metal, turning, small , olive, in, to, yellow, grey, steel, low to, on, airbourne, military style, designated, upside, general aviation, Flying in, tied down to, on front of, belonging to
3	garbage, can, sitting beside, placed, covered, full of, in, before, gray, on, grey, against, dirty, steel, discarded, metal, piled in, turned over, blue , collecting, staning on, in garbage, overflowing with, heaped, redbull, comet cleanser, opener
4	chandelier, simple, hanging down from, around, lighted, hangs over, in, hanging over, on, attached by, antique, reflected in, brass, metal, screwed into, lit, ceilinglight, silver ring
5	green leaves, tree, tree bark, in water, on, Is on, Dark green, full of, lush, in, tree pleasant, needles, hanging over, decal, tree line, motor boat, green trees

Table 12: Sample of communities for **vg-rand-graph, BIGCLAM**

Abbeel Taskar and et al. Discriminative probabilistic models for relational data. *UAI Association for Uncertainty in Artificial Intelligence*, 2002.

Wong Taskar and et al. Link prediction in relational data. *NIPS Neural Information Processing Systems*, 2003.

Robert West, Joelle Pineau, and Doina Precup. Wikispeedia: An online game for inferring semantic distances between concepts. *21st International Joint Conference on Artificial Intelligence (IJCAI)*, 2009.

Jaewon Yang and Jure Leskovec. Overlapping community detection at scale: A nonnegative matrix factorization approach. *WSDM*, 2013.

Jaewon Yang, Julian McAuley, and Jure Leskovec. Community detection in networks with node attributes. *IEEE International Conference on Data Mining*, 2013.

CID	Nodes
1	base, encased in plastic, old, detached, flat, front left of, wired usb, lays on, standard pc colour, dark grey, near the, piano, mac, ergonomically correct, built in, beige, round circle compressor above, tinted, full of buttons, sits, folding, wrapped, existing, on in, usb, before, dirty, edge, full size, close, laptop, on a, missing keys, rounded, laptop, keyboard, beneath a, music, encapsulates, Dell, located on, sideview
2	dinner plate, pineapple, delicious, hamburger, toppings
3	curtain-holder, ornamental, black iron fence, footboard, iron rods, iron hook, iron fencing, faucets, birdcage, grill railing, iron
4	diced onion, delicous looking, spinach, on the pizza, various pizza, toppings
5	baking sheet, stainless, logo on cupcake, cooking on a, used to shape bread, frying, for frying, foil, nonstick, pans, pan, dedicated to, frying, Wrought iron, reaching for, laying atop, shiny aluminium, whole, space on pizza

Table 13: Sample of communities for **vg-rand-graph, Link Clustering**