Table 3: The 10 most effective features for people.

Feature	Coefficient
log-sum-likes(dbo:occupation)	0.753
avg-likes(dbo:parent)	0.697
log-max-likes(dbo:award)	0.640
sqrt-sum-likes(dbo:occupation)	0.609
sqrt(dbo:networth)	0.590
log-avg-likes(dbo:parent)	0.563
sum-likes(dbo:spouse)	0.532
sum-likes(dbo:parent)	0.528
max-likes(dbo:parent)	0.511
sqrt-sum-likes(dbo:influenced)	0.499

Table 4: The 10 most influential predicates for people.

Predicate	Influence
dbo:occupation	2.88
dbo:parent	2.52
dbo:award	2.48
dbo:hometown	2.25
dbo:spouse	1.75
dbo:influenced	1.51
dbo:influenced/reverse	1.39
dbo:education	1.30
dbo:relative	1.17
dbo:militaryBranch	1.10

high transmissibility. For that reason, the public figures are growing with their mutual influence. This bidirectional effect is apparent in classical musicians.

Next predicate, dbo:education is the kind of university. That is a person who received a good education becomes popular. This case is apparent for politicians and scientists.

Therefore, we find that popularity is inherited by people from people. If one wants to become popular, making a network link to popular person would be good strategy.

5.3 Mutual Influence Network

Next, we portray the obtained influences as a network. MIN, which represents Like influences among various classes such as not only persons but also places, books, and albums in Fig. 7. A node represents a class which has URI as same as the label of node. The green ones denote person class and subclasses. Thickness of an edge represents the influence $\|\mathbf{u}_p\|_{L1}$ of the corresponding predicate p (see Section 4.3 for definition of $\mathbf{u}_p$). We generate the network based on the ontology of DBPedia: we use the information of domain and range of predicate. If there are triples; then we make a labeled edge and weight it based on the influence.

As discussed in Section 5.2, numerous edges exist with respect to the person class. Many are not shown in the figure.

The original obtained graph is too large. Therefore, we choose two meaningful sub-networks. Fig. 7 shows the sub-network related to work. The center green node is the person class. The Like of a band affects its members, and the Like of Band is transmitted by its single music selections. Like of a single is determined by the artist and its Like is determined by their overall work. For example, Likes of the band members of Beatles depend on the following.

Class `:Work` is the class of all works. The Like of each work transmits to its author, creator, writer and even publisher. Likes of each work is affected by other works. Predicates `:previousWork` and `:subsequentWork` means the order relation of works and it makes list structure. Likes of class :Book influences to its cover artist, illustrator and especially writer. For example, the author of Harry Potter is now attracting many Likes. This fact supports the first assumption. Likes of a scientist are affected strongly by the scientist's doctoral student and notable students. For example, a student in a popular laboratory tends to attract fans in the researcher community.

6. RELATED WORK

Some work related to this paper was conducted earlier. We particularly introduce those studies.

6.1 Prediction

Link-based prediction: Several past studies cope with problems of predicting attributes with regard to given entity from the relations around the entity. Chang *et al.* [3] predict users' locations in Twitter based on social network. The experimentally obtained results dealing with social network improve the accuracy. Lu *et al*. [8] employ entity relations to classification problems. They proposed link-based classification and predict the class of given entity as the function of class of neighbors of entity. Similar to that approach, Hu *et al*. [6] bring a Wikipedia entity relation into a user query intent-classification problem in information retrieval. They first define the ground truth of the user intent class onto a few seed entities and propagate the class into related entities. In contrast to these approaches, the novelty of our approach is disguising multiple types of relationships using predicates. The experimentally obtained results shows agreement with predicates, which contributes to the model's good performance.

Feature construction from ontology: In this paper, we explained automatic feature construction from DBPedia. Paulheim *et al*. [11] provide a feature constructor from Linked Open Data: FeGeLOD. It automatically generates SPARQL, and performs it to RDF data and obtains features. However, the approach does not consider multiple objects with regard to a subject and a predicate exist. This case occurs in feature construction from DBPedia frequently. For example, `db:parent` would take two objects. In this paper, we consider multiple objects and proposed aggregation.

6.2 Entity Linking

Entity linking is an important task of our study that creates a bridge between Facebook and DBPedia. Some approaches have been proposed. Shen *et al.* [12] proposed *LINDEN*, which connects named entities with a knowledge base. They proposed a probabilistic model using description text in addition to title to solve the disambiguation problem. Although we can apply this method to this issue, the difference of the approach presented herein from their approach is that it uses category information provided both by Facebook and by DBPedia. As described in this paper, we show that category information improves the accuracy of the linker. Demartini *et al*. [4] proposed *ZenCrowd*, which leverages a crowd-sourcing platform to solve complex linking tasks. They combined the automatic entity linking method with one done manually by setting minimal confidence to an automatic linker and throwing the task to a crowd-sourcing platform when confidence of the linker falls below the minimal confidence level. That framework can be combined with our approach. Milne *et al*. [9] proposed a knowledge base search engine, *Koru*, which integrates search results and DBPedia and enables users to expand queries. Our Like information would improve their search result ordering algorithm.

6.3 Facebook

This paper is the first describing analysis of Like counts on a large scale. We introduce some earlier reports related to Facebook.

Crawling: The crawling approach for Facebook was proposed [10] with simple crawling approaches such as breadth first search and random walk, which tend to misestimate the original degree distribution. For instance, random walk overestimates values