# Aggregating Predictions vs. Aggregating Features for Relational Classification

Oliver Schulte
School of Computing Science
Simon Fraser University
Burnaby, B.C., Canada
Email: oschulte@cs.sfu.ca

Kurt Routley
School of Computing Science
Simon Fraser University
Burnaby, B.C., Canada
Email: kdr4@sfu.ca

***Abstract*—Relational data classification is the problem of predicting a *class label* of a target entity given information about features (attributes) of the entity, of the related entities, or neighbors, and of the links. This paper compares two fundamental approaches to relational classification: aggregating the features of entities related to a target instance, or aggregating the probabilistic predictions based on the feature of each entity related to the target instance. Our experiments compare different relational classifiers on sports, financial, and movie data. We examine the strengths and weaknesses of both score and feature aggregation, both conceptually and empirically. The performance of a single aggregate operator (e.g., average) can vary widely across datasets, for both feature and score aggregation. Aggregate features can be adapted to a dataset by learning with a *set* of aggregate features. Used in this way, aggregate features outperformed learning with a single fixed score aggregation operator. Since score aggregation is usually applied with a single fixed operators, this finding raises the challenge of adapting score aggregation to specific datasets.**

## I. Introduction

Most real-world structured data are stored in the relational format, with different types of entities and information about their attributes and links between the entities. Relational data classification is the problem of predicting a *class label* of a target entity given information about features (attributes) of the entity, of the related entities, or neighbors, and of the links. A key challenge for relational classification is that the number of links of the target entity is not uniformly bounded. Since the features of each neighbor potentially carry information about the target class label, the number of predictive features for classification is thus a function of the size of the target entities neighborhood, rather than a fixed dimensionality $d$. Relational classifiers therefore aggregate the information from the target entity's neighborhood. There are two fundamental options for aggregation: 1) First aggregate the neighbors' features into a single aggregate feature vector, then classify based on the aggregate vector. 2) First derive a classification score based on a single neighbor, then aggregate the scores. The most common approach is to use a probabilistic classifier that assigns probabilities to class labels, then use a *combining rule* to compute a probability from a multiset of probabilities [1], [2]. Figure 1 illustrates these options schematically.

In this paper we compare the two aggregation approaches empirically on data sets with continuous features. We use three

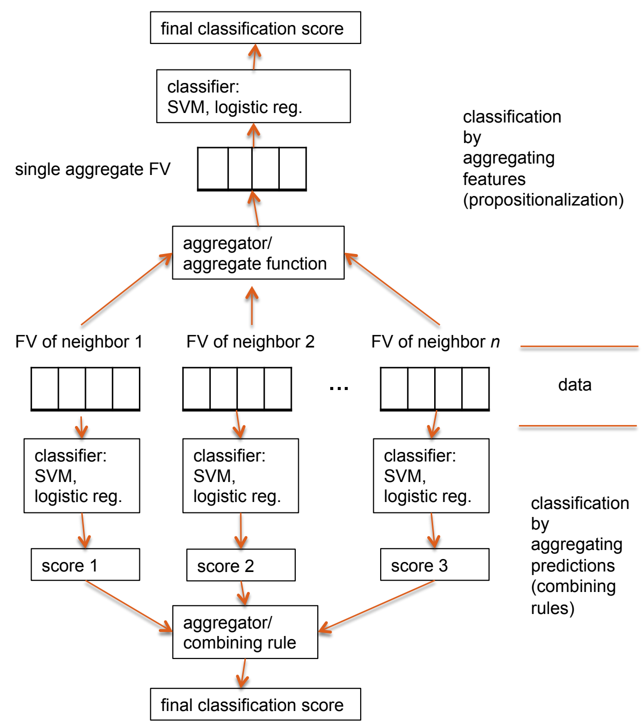


Fig. 1. Two different approaches to relational classification. FV = feature vector. Top: aggregating features combines the $n$ feature vectors into a single one, then applies a standard nonrelational classifier to predict a class label. Bottom: Score aggreation applies a standard nonrelational classifier to each feature vector to obtain $n$ positive class probabilities. A combining rule aggregates the $n$ scores to predict a class label.

real-world continuous datasets that summarize players' actions in ice hockey, soccer, and basketball. This paper is the first to apply relational classification to these sports datasets. Two standard datasets for financial data and IMDb reviews are also analyzed.

### A. Evaluation

Our experiments utilize logistic regression as the base classifier for both feature and score aggregation. Computationally, classifier training with score aggregation can be done very simply by forming a data table such that one row contains the features of one neighbor, and applying a regular non-relational learning algorithm to this table. Our experiments compare a number of standard combining rules. To our knowledge, this is the first extensive comparison of different combining rules on a range of datasets.

We use standard aggregation functions to aggregate continuous features (average, sum, min, max, midrange, geometric mean). Once features have been aggregated, any standard single-table machine learning classifier for continuous features can be applied for classification.

Problems with feature aggregation have been well studied [3], [4]. Aggregating a set of values into a single value loses information about the value distribution. Also, aggregation produces a single aggregate value for a target instance no matter how many links the target instance has. This causes problems in the presence of degree disparity, where some instances are related to many entities and others to only a few. We discuss these issues further in Section IV below. Despite these known problems, feature aggregation outperforms score aggregation in our experiments. We provide evidence that this is mainly due to two issues. First, score aggregation also suffers from degree disparity, because during learning, instances with many links carry more weight than instances with fewer. Second, aggregate features relevant to classification can be separated from irrelevant ones by (more or less) standard feature selection techniques. In contrast, score aggregation is used with a fixed combining rule chosen by the user for a dataset. Our results show that the performance of different combining rules can vary widely for different datasets. It is not clear how a user or a learning algorithm can determine a good combining rule for a given problem.

### B. Contributions

Our main contributions may be summarized as follows.

1) A comparison of an extensive set of combining rules for aggregating probabilistic predictions in relational classification.
2) A comparison of combining rules for score aggregation to learning with a set of aggregate features.

### C. Paper Organization

We first review related work. Then we introduce notation for describing score and feature aggregation. We provide a conceptual discussion of the pros and cons of both approaches, which is the basis for our experiments. Next we describe the datasets used in our experiment, then report results: a comparison of combining rules among themselves, an evaluation of feature aggregation methods, finally a comparison of the score aggregation with feature aggregation.

## II. Related Work

Because of the importance of relational data, there has been much work on relational classification. For overviews please see [5], [6]. We provide a high-level description of the work most relevant to the question of feature vs. score aggregation.

### A. Propositionalization

The majority of work on relational classification has adopted the feature aggregation strategy. This approach of "flattening" the relational structure is generally known as *propositionalization* [7]. For continuous features, propositionalization methods use the same standard aggregate functions that we use in this paper [8], [9].

### B. Learning With Aggregate Features

The most expressive propositionalization models apply feature functions to combinations of the discrete features given in the data (e.g., [10]). For instance, to predict the ranking of a student, we may distinguish the number of A grades achieved in higher-level course from those achieved in lower-level courses. Complex discrete features may be combined with aggregation functions, as aggregation conditions, for continuous variables [9], [11]. For example, to predict the age of a user in a social network, we may consider the average age of her friends who have the same gender and live in the same city.

Several researchers discuss advantages and disadvantages of propositionalization for link-based classification [12], [5]. The main advantage is expressiveness: feature generation methods search a large space of potentially useful features. If an informative new complex feature or aggregate feature can be found, it improves classification performance and informs the user. The disadvantages are problems with both statistical and computational efficiency. Aggregation loses information in the data, which increases the variance of classifier estimates and causes problems with both type 1 and type 2 errors in feature selection [13]. Searching a large space of potential features presents considerable computational challenges. For an example, generating 100,000 features on the standard CiteSeer dataset, can take several CPU days [11, Ch.16.1.2]).

The feature aggregation method we use in this paper is intermediate between choosing a single fixed aggregate operator and searching through a space of complex expressions. For each original unaggregated feature, we apply a fixed set of aggregate operators (such as average, maximum, etc.). These are provided as input features to a standard learning method (e.g., logistic regression). So there is no search through a complex feature space, but learning is used to select and weight relevant aggregate features.

### C. Sports Statistics

The problem of predicting the results of sports matches has received considerable attention for different sports. For an overview please see [14]. We do not claim that the methods in this paper are competitive for predicting the match results. We use sports data for real-world datasets in an interesting domain with interpretable features for comparing aggregating features vs. aggregating predictions.

The closest predecessor to our work is that of Neville *et al.* [3]. Key differences include the following. 1) They used only the average operator for feature aggregation, rather than a set of aggregate operators. 2) For score aggregation, they used the arithmetic and geometric mean only. 3) They did not consider adjusting instance weights to improve score aggregation methods. 4) Their experiments used the Naive Bayes classifier applied with mainly discrete features. We use logistic regression with mainly continuous features, which are more natural for feature aggregation.

## III. Notation and Data Format

We introduce notation to discuss relational features and data and to support theoretical analysis. We follow the functor-based notation for combining statistical and relational concepts due to Poole [15].

### A. Functor Features

A **population** is a set of individuals, corresponding to a domain or type in logic. A **feature** is of the form $f(t_1, \ldots, t_k)$ where $f$ is a functor and each $t_i$ is a first-order variable or a constant. Each feature has a set of values (constants) called the **domain** of the feature. A **grounding** replaces each 1st-order variable in the feature by a constant; the result is a ground feature. A grounding may be applied simultaneously to a set of features. One of the features is the class or **target** feature. A grounding of the target feature is a **target instance**.

### B. Examples

In our datasets the basic populations are teams, players, matches, with corresponding first-order variables $T, P, M$. Examples of features include the following.

- $\mathit{result}(T, M)$ denotes the result of a team in a match (win or lose). This is the target feature.
- The ground feature $\mathit{result}(\mathit{Canucks}, \mathit{1})$ denotes the result of the Canucks in match 1. This is a target instance.
- $\mathit{goals}(P, T, M)$ is the number of goals scored by a player in a match.
- +/-$(P, T, M)$ is the +/- score of a player in a match. This is a common measure of the player's performance; for precise definition see [14].

### C. Aggregation

Given a feature $f$, an aggregate function $\mathit{agg}$ applies to one of the argument variables of $f$. We use the subscript notation $\mathit{agg}_X$ to indicate that variable $X$ is the object of aggregation [11]. The result is a feature with one less argument. Examples include the following.

- $\mathit{goals}(T, M) \equiv \sum_P \mathit{goals}(P, T, M)$ is the number of goals scored by a team in a match.
- $\mathit{past\_goals}(P) \equiv \sum_{M \text{ in past season}} \mathit{goals}(P, T, M))$ denotes the sum of a player's goals in the past season.

### D. Relational Data Tables

Relational data can be visualized in terms of the **groundings data table**. The data table has one column for each feature. It has one row for each simultaneous grounding of all functor features where the instances of the nonclass features are in the neighborhood of the instance of the target feature. Thus if the target functor feature is instantiated with ground instance $t$, the data table contains a row listing the attributes of each neighbor $n$ of $t$. Tables I and II (propositionalized) show an example of groundings data tables. As the examples illustrate, aggregation increases the number of features (columns) and decreases the number of data points (rows). Table I is constructed as follows. A row in this table corresponds to a match, one of the teams involved in the match, and one player who played for that team in the match. Each NHL team dresses exactly 18 skaters per match, so for a given match, the data table contains $2 \times 18 = 36$ rows. The 19 columns represent the result of the match, and the 18 last-season statistics of the player.

TABLE I
GROUNDINGS DATA TABLE FOR NHL.

| Instance Weight | result(T,M) | MatchId M | TeamId T | PlayerId P | past_goals(P) | goals(T,P,M) |
|---|---|---|---|---|---|---|
| 1/18 | Loss | 2010020023 | Canucks | D. Hamhuis | 5 | 0 |
| 1/18 | Loss | 2010020023 | Canucks | D. Sedin | 34 | 0 |
| 1/18 | Loss | 2010020023 | Canucks | H.Sedin | 32 | 0 |
| ... | ... | ... | ... | #4–#18 | ... | ... |
| 1/18 | Win | 2010020033 | Canucks | D. Hamhuis | 5 | 0 |
| 1/18 | Win | 2010020033 | Canucks | C. Ehrhoff | 17 | 0 |
| 1/18 | Win | 2010020033 | Canucks | H. Sedin | 32 | 0 |

TABLE II
AGGREGATE FEATURE DATA TABLE FOR NHL.

| result(T,M) | MatchId M | TeamId T | Sum_past_goals(T) | Sum_goals(T,M) |
|---|---|---|---|---|
| Loss | 2010020023 | Canucks | 252 | 1 |
| Win | 2010020033 | Canucks | 259 | 2 |

## IV. Score Aggregation vs. Feature Aggregation: Strengths and Weaknesses

We describe carrying out relational classification with aggregate features and scores. We discuss the basic strengths and weaknesses of each approach, which motivate the design of the methods in our experiments.

Classification with aggregated features is conceptually straightforward: aggregation produces a data table with one row per target instance that can be treated like a standard attribute vector table. See Table II for illustration.

Classification with aggregated scores can be visualized in terms of the **groundings data table**, or data table for short; see Table I. For simplicity, we discuss score aggregation for a single relationship, which defines a neighborhood for each grounding of the target feature. Our discussion applies equally to classification scores obtained with different types of neighborhoods. Suppose that we have trained a classifier model $\mathcal{M}$ that returns a classification score for a given target label $y$ and feature vector $\boldsymbol{x}$. We write $\mathit{score}_{\mathcal{M}}(y; \boldsymbol{x})$. We can apply this classifier to each row in the groundings data table to derive a classification score from the features of each neighbor of a given target instance $t$. Given a list of classification scores, one for each row in which the target instance appears in the data table, we can apply a standard aggregation function to obtain an overall classification score. We also use the noisy-or rule for combining probabilities [2]. For a classifier whose score indicates the probability of a positive classification, such as logistic regression, we treat the aggregate probability as the overall probability of a positive classification for the target instance, as in [3]. Table III summarizes the aggregation functions shared by feature and score aggregation, as well as the aggregate functions specific to each method.

TABLE III
AGGREGATE FUNCTIONS USED

| | Feature Aggregation | Score Aggregation |
|---|---|---|
| **Shared Functions** | Average $\mu$, Maximum, Minimum, Midrange, Geometric Mean | |
| **Specific Functions** | Sum, Standard Deviation, Degree | Noisy–Or |

### A. Feature Aggregation: Strengths and Weaknesses

Feature aggregation is a very common approach to relational classification and has been much discussed [3], [4], [5]. We review the main points relevant for our study. Feature aggregation is conceptually attractive in that it reduces relational classification to non-relational classification with a single feature vector per target instance. Reducing the size of the data table also speeds up learning, as our experiments show.

The obvious drawback of feature aggregation is that it loses information about the distribution of features. Consider the problem of predicting the box office receipts of a movie from user ratings. As an extreme thought experiment, suppose that all movies in our dataset receive the same average user rating, but that the variance of their ratings differs. Then by using the average rating as the aggregate feature, all predictive information is lost. In our experiments, we address the potential loss of information in two ways. (1) We add a set of aggregate features to the data, rather than fixing a single aggregate operator in advance. In this way, learning can decide which aggregation operation is the most informative. (2) In addition to the mean of a feature, we add its standard deviation as an aggregate feature. Thus learning is provided with information about the first two moments of the feature distribution rather than only the first. To our knowledge, adding standard deviation as an aggregate feature is novel.

Another known problem with aggregate features is *degree disparity.* Degree disparity refers to the fact that the degree, i.e., the size of relational neighborhoods, may vary widely for different target instances. For example, the number of ratings received by a movie may vary from zero to thousands. One problem with using aggregate features with degree disparity is that the data lose the information about the size of the relational neighborhood. Also, the values of many aggregate functions correlate with degree [4], i.e., they tend to increase with the degree. So the aggregate feature conflates information about the degree with information about the original feature. To address this conflation, we add the relational degree of each target instance as an aggregate feature in our experiments. Adding a degree feature is recommended by [4].

### B. Score Aggregation: Strengths and Weaknesses

The main strength of score aggregation is that it retains the full distributional information in the data. A computational drawback is reduced speed because of the larger data table size. Another issue is that it applies a single fixed aggregate function to scores, rather than exploring an aggregate function space. A problem that figured prominently in our experiments, but seems not to have been previously discussed, is that score aggregation is also affected by degree disparity. As an extreme thought experiment, suppose that our dataset contains ratings for 100 movies, 99 of which have received only 1 rating, and 1 of which has received 99 ratings. So the groundings data table contains 99 rows for the one movie, and 99 rows for the other 99 movies. Hence applying a standard machine learning algorithm to the groundings data table "as is" overweights the movie with the large degree. To address degree disparity for score aggregation, we reweight the rows in the data table by dividing by the degree of each row's target instance. In our thought experiment, the rows for the single large-degree movie would be reweighted by 1/99, and the rows for the others would retain unit weight. Table IV summarizes the main points of our discussion. Our empirical evaluation examines these basic aspects of feature and score aggregation and the effectiveness of solutions to address them.

TABLE IV
CONCEPTUAL COMPARISON OF FEATURE VS. SCORE AGGREGATION

| Aggregation | Pros | Cons | Remedy |
|---|---|---|---|
| **Features** | Utilizes multiple aggregate functions | Increases dimensionality | |
| | Fast learning | Loses distribution information | Add standard deviation feature |
| | Less memory required | Ignores Degree Disparity | Add degree feature |
| **Scores** | Full Distribution Information | Uses a single fixed aggregator | |
| | | Degree disparity: overweights instances with many links | Reweight Instances |

## V. DATASETS

We carry out experiments on six data tables derived from five real-world databases. All our datasets are available online http://www.sfu.ca/ kdr4/SSCI2014.zip. The datasets vary in size and degree disparity. For each data table, we obtain two versions: the groundings data table (cf. Table I) and the feature aggregation table (cf. Table II). So each classifier is applied to twelve datasets. Two standard databases have been previously used in studies of relational learning, IMDb and Financial. We introduce four new datasets from sports databases: the National Hockey League (NHL), UK Premier League (PLG), and National Basketball Association (NBA). Sports datasets are challenging for learning because of their complexity. At the same time, they are engaging to many users. They are suitable for studying the effects of aggregation because aggregate functions such as average, sum, etc. most naturally apply to continuous features, and sports datasets contain mainly continuous features, namely counts of players' actions. We describe the details of the datasets. Then we summarize the properties of the datasets that are relevant to feature and score aggregation, as discussed in Section IV, such as degree disparity and the variance of feature distributions.

### A. Dataset Details

For each sports dataset, the target feature is $result(Team, Match)$. A positive classification means that the team is predicted to win the match. The target features for IMDb and Financial are given below.

*1) IMDb:* The hierarchical relational structure of the IMDb dataset[1] is as follows: each *director* has their own attributes and has directed 1 or more *movies*. Each *movie* has been reviewed and rated by 1 or more *users*, who also have their own attributes. During feature aggregation, the *user* attributes and ratings are aggregated. The target feature for the IMDb dataset is $highBoxOffice(Movie, Director)$, where the positive class denotes the movie had a box office receipt of $\$10,000,000$ USD or greater. The IMDb dataset contained five discrete features, which we converted to continuous 0-1 "dummy variables" [16], where the presence of each discrete value is represented by a Boolean indicator variable.

*2) Financial:* The financial dataset has a hierarchical relation structure with *district* at the top level, followed by *accounts* within the *district*, and finally all the *transactions* associated with a particular *account*. During feature aggregation, the attributes of the transaction are aggregated. The financial dataset also suffers from degree disparity, as shown in Table VII. The target feature is $hasLoan(Account, District)$, where a positive classification means there is a loan associated with the account. There were five discrete features present in the financial dataset, which were converted to continuous "dummy variables". This dataset is a modified version of the financial dataset from the discovery challenge at PKDD'99 following the modification from [17].

*3) PLG data tables:* We used Opta data [18], released by Manchester City. It lists ball actions of each player in each game, for the 2011-2012 season. Number of goals, passes and tackles by a player in a match are examples of the information associated with each player. For each player in a match, our data set contains 199 player actions as features.

*4) NBA data tables:* NBA data was obtained manually from http://www.nba.com/. Box scores containing match summary statistics for each player were used to create the data table. For each *player* on a *team* in a *match*, there are 19 continuous player statistics recorded, such as number of free throw attempts and total number of player rebounds. These player statistics are aggregated over each $(team, match)$ instance during feature aggregation. No team attributes or previous player statistics are used in the data table.

*5) NHL data tables:* We used the Selenium webcrawler [19] to download player game statistics (Box Scores) from http://www.nhl.com/gamecenter/ for the seasons 2009–2013. The box scores summarize player statistics for each match, a total of 13 continuous-valued features. We refer to these as **match statistics**. We only consider skaters in our model and remove goalies, as the number of goalies in the NHL is significantly less than the number of skaters, and different statistics are recorded for goalies than skaters. The match features include goals, assists, plus-minus, penalty minutes, and total time on ice. For each player, we sum his match statistics over all NHL games in the previous season to obtain a total of 13 statistic totals for the previous season. In addition, we add 5 other season statistics: number of games played, game winning goals, powerplay goals, shorthanded goals, and shot percentage. We refer to the resulting 18 features as **last-season features**. From this database we prepared the following two groundings data tables, depending on whether we used last-season features only or all features.

**Season** Contains last season features only.

**S+Match** Contains last season features and match statistics.

### B. Feature Distributions in Datasets

We examine summary statistics for our datasets pertinent to the discussion of aggregation in Section IV. Table V shows the strong effect that aggregation has on the data table dimensions. It reduces the number of rows (data points), in the case of IMDb by a factor of around 300. Aggregation increases the number of columns (features), in the case of the PLG soccer data, by a factor of almost 7.

TABLE V
DATA TABLE DIMENSIONS

| Dataset | Rows | Aggregated Rows | Columns | Aggregated Columns |
|---|---|---|---|---|
| IMDb | 909,377 | 2,910 | 64 | 118 |
| Financial | 348,095 | 1,364 | 130 | 280 |
| NHL - S + Match | 138,852 | 7,714 | 35 | 221 |
| NHL - Season | 138,852 | 7,714 | 22 | 130 |
| PLG | 7,933 | 580 | 203 | 1,397 |
| NBA | 767 | 60 | 23 | 137 |

Table VI illustrates how aggregation decreases the variance of features. We selected one attribute for each dataset, and compared its variance on the original groundings data table to its variance after applying the average $\mu$ aggregator. A reduction in variance can be seen as a reduction in information content (as Principal Component Analysis seeks to maximize variance of projections).

TABLE VI
FEATURE VARIANCE VS. AVERAGE FEATURE VARIANCE

| Dataset | Attribute A | Variance A | Variance $\mu$A | Reduction Ratio |
|---|---|---|---|---|
| IMDb | Age(User) | 135.97 | 19.37 | 7.02 |
| Financial | Amount(Transaction) | 112,257,686.00 | 16,838,158.97 | 6.67 |
| NHL - S + Match | GamePlusMinus(Player) | 1.16 | 0.33 | 3.50 |
| NHL - Season | LastSeasonPlusMinus(Player) | 112.80 | 24.69 | 4.57 |
| PLG | Goals(Player) | 0.13 | 0.01 | 13.67 |
| NBA | PlusMinus(Player) | 118.02 | 38.03 | 3.10 |

Table VII shows the sports datasets have small to no degree disparity. This is because the number of players in a team in a match varies very little. In ice hockey, each NHL team dresses exactly 18 skaters per match. The PLG dataset exhibits some small degree disparity, as a maximum of three substitutions per team are allowed during PLG matches. The IMDb and Financial datasets exhibit considerable degree disparity, as shown in Table VII. The number of ratings for movies varies greatly. For financial transactions, different accounts may be involved in transactions to highly varying degrees of frequency.

Table VIII examines the effect of aggregation on the apparent correlation between features and the class label. We used the information gain metric to measure the relevance of a feature to the class label. This measures the reduction in uncertainty from observing the value of the feature, 0 is the

[1] www.imdb.com, July 2013

TABLE VII
DEGREE DISPARITY

| Dataset | Relationship | Average | Standard Deviation | Max | Min |
|---|---|---|---|---|---|
| IMDb | Ratings/Movie | 313.91 | 411.92 | 3,427.00 | 1.00 |
| Financial | Transaction/Account | 255.68 | 134.09 | 675.00 | 9.00 |
| NHL | Players/Team,Match | 18.00 | 0.00 | 18.00 | 18.00 |
| PLG | Players/Team,Match | 13.64 | 0.63 | 14.00 | 11.00 |
| NBA | Players/Team,Match | 12.71 | 0.45 | 13.00 | 12.00 |

minimum and 1 the maximum value. The information gain was computed using Weka's built-in feature selection method [20]. The second column shows the attribute with the highest information gain *before* aggregating in the groundings data table. The last column shows the information gain of the average $\mu$ of the best attribute in the aggregate feature table. In all cases, the information gain increases of the attribute increases, typically by a factor of five or more. There are two ways of looking at this result. Neville *et al.* [4], [21], [3] argue the correlation after aggregating is overestimated, because using a single aggregate value in place of a multiset of values is like replacing each value in the multiset by the aggregate value. This underestimates the variance of the feature and overestimates its correlations to the class feature. They argue aggregation methods are liable to spurious correlations, erroneously accepting features as relevant that in fact are not. Another point of view is that for features that *are* relevant to classification, aggregation helps to reveal the relevance. For example, in the soccer data PLG, it is intuitively clear that the number of goals scored by the players on a team is relevant to predicting the outcome of the match, and so is the average number of goals. So increasing the observed information gain from 0.03166 to 0.57900 is helpful for learning. Another example is the average revenue of a movie's director, over all of his or her movies. This is a feature of a movie, not of its links, and remains the same before and after aggregating (e.g. aggregating the movie ratings). But its information gain increases after collapsing the movie's relational neighborhood into a single vector. In a scenario where aggregation highlights true and spurious correlations to the class label, an effective learner can sort out which aggregate features are truly informative and thus gain from the stronger effects. Our empirical results examine the effect of aggregation on classifier performance.

## VI. EVALUATION

### A. Methods and Comparison Metrics

As a base classifier, we use logistic ridge regression. Parameters of the classifier were set by a grid search that evaluated a parameter setting by examining testing errors. We report the results for the best parameter setting found by the grid search.

For feature aggregation, we report results for pairs (Classifier x Dataset). All aggregations of all features are used to train the classifier. For score aggregation, we report results for pairs (Aggregate operator x Dataset). Table III summarizes the aggregate operators used.

Our basic metrics are **classification accuracy** (percentage of correctly classified target instances) and **F1-measure**, the harmonic mean of precision and recall [22]. We train the classifiers on a training set of target instances and test on the remaining target instances. All datasets use an $80 : 20$ split for the training and test sets.

### B. Results

We first compare different methods for score aggregation, then for feature aggregation, finally both together. For the purpose of discussion, we refer to the average, geometric mean, and midrange operators as *averaging* since they can be viewed as a form of averaging class probabilities[2]. We refer to the remaining three maximum, minimum, and noisy-or as *extremal operators* since they agree with extremely high values (maximum, noisy-or) or low values (minimum).

*1) Score Aggregation:* Table IX shows the accuracies for each combination of (trained base classifier, score aggregator) for each dataset. The F1-Measures showed the same trends. On the datasets with substantial degree disparity (IMDb and Financial, see Table VII), scaling the importance of instances by the number of their links improved accuracy for almost all score aggregators, and led to the best overall performance. The averaging aggregators achieved good predictions on all datasets: (arithmetic) average, midrange, and geometric mean. Methods that favor extremal values over averaging, such as Maximum, Minimum, and Noisy-Or, can perform very well (e.g., Minimum on IMDb), but also very poorly (e.g. Minimum on PLG and NBA). As a default score aggregator, the average aggregator provides consistently accurate predictions. For the comparison with feature aggregation, an important observation is that we do not seem a dominant score aggregator across datasets. On IMDb-W, Minimum is best, on Financial-W and NBA the three averaging operators, on NHL-S+M the arithmetic average, on NHL-Season the two means, on PLG the midrange operator. These differences are statistically significant (t-test, $p < 0.05$). A striking failure of the extremal operators is that for match outcome prediction, they fail to take advantage of obviously relevant match features such as the number of goals performed: their classification accuracy is close to using only statistics from the previous season.

This variation in predictive accuracy suggests finding a good score aggregation method requires experimentation and/or a learning method. Methods for learning a good score aggregation method for a given dataset are an interesting topic for future work. In contrast, standard feature selection techniques can be applied to select a good feature aggregator for a given dataset. In the next experiments we examine the performance of our straightforward approach where the original feature space is expanded with a fixed set of aggregated features.

*2) Feature Aggregation:* Table X presents the results for feature aggregation methods, on all six datasets. The F1-Measures showed the same trends. Adding the standard deviation of continuous features tended to improve predictions, but only the difference on Financial was statistically significant. There is clearly improvement potential for a more sophisticated way of using variance information. Examination of

[2]midrange = ((max - min)/2)

TABLE VIII
ATTRIBUTE INFORMATION GAIN

| **Dataset** | **Best Attribute** | **Information Gain** | $\mu$ **Best Attribute** | **Information Gain** |
|---|---|---|---|---|
| IMDb | Director_AvgRevenue | 0.10154 | Director_AvgRevenue | 0.51870 |
| Financial | Remittance(Transaction) | 0.27386 | AVG(Remittance(Transaction)) | 0.35431 |
| NHL - S + Match | PlusMinus(Player,Match) | 0.11712 | $\text{AVG}_P$(PlusMinus(Player,Match)) | 0.55938 |
| NHL - Season | LastSeasonPlusMinus(Player) | 0.00138 | AVG(LastSeasonPlusMinus(Player)) | 0.00715 |
| PLG | Goals(Player,Match) | 0.03166 | $\text{AVG}_P$(Goals(Player,Match)) | 0.57900 |
| NBA | PlusMinus(Player,Match) | 0.17400 | $\text{AVG}_P$(PlusMinus(Player,Match)) | 0.87400 |

TABLE IX
SCORE AGGREGATION ACCURACIES

| **Aggregator** | IMDb | IMDb-W | Financial | Financial-W | NHL - S + Match | NHL - Season | PLG | NBA |
|---|---|---|---|---|---|---|---|---|
| Average | 78.52% | 81.44% | 69.12% | 73.16% | **87.29%** | **55.25%** | 90.52% | **100.00%** |
| Geometric Mean | 78.69% | 81.44% | 69.49% | 72.43% | 85.08% | 55.12% | 81.03% | **100.00%** |
| Midrange | 78.52% | 80.93% | **72.79%** | **73.53%** | 85.34% | 52.79% | **93.10%** | **100.00%** |
| Maximum | 71.65% | **82.13%** | 63.97% | 68.75% | 52.01% | 50.65% | 53.45% | 50.00% |
| Noisy-Or | 71.65% | 82.13% | 63.97% | 64.34% | 52.01% | 50.65% | 53.45% | 50.00% |
| Minimum | **81.79%** | 79.73% | 53.68% | 51.10% | 50.45% | 50.00% | 51.72% | 58.33% |

regression weights showed average and sum of plus-minus and goals to be strong predictors for sports datasets.

Since feature aggregation produces a standard data table, we can apply any classifier, including nonprobabilistic ones, to benchmark logistic regression. Table XI shows the result of SVMs with various kernels. Comparing with Table X, we see that logistic regression performs well on the aggregated datasets compared to SVMs. Also, among the SVM kernels, the linear kernel provides accurate predictions. Together with the success of logistic regression, this indicates that aggregation makes our datasets close to linearly separable. This is another way in which feature aggregation can improve classification, despite the loss of information it entails.

*3) Score Aggregation vs. Feature Aggregation:* Table XII compares the best feature aggregation method with the best score aggregation method. Feature aggregation has statistically significant greater classification accuracy and F1-measure than score aggregation on all datasets, with the exception of the NHL-Season and NBA datasets, where there is no statistically significant difference. On the Financial dataset, feature aggregation outperforms score aggregation by a wide margin of 13.97%. Feature aggregation outperforms score aggregation the most on the two datasets with the greatest degree disparity. Together with the results of Table IX, this is evidence that degree disparity causes problems for score aggregation as well as feature aggregation.

Table XIII shows that aggregation can speed up learning considerably by reducing the number of data points (speed up a factor of 15 on IMDb) for example. The trade-off is that the number of extra features added slows aggregation down, which we observe on the PLG soccer data set.

In sum, comparing score aggregators, the average suggested by the random selection semantics appears to be the most robust aggregator, providing competitive performance in a variety of settings.

A hybrid approach that combines score aggregation with feature aggregation could address the weaknesses of both approaches. For example good features could be found learning a model based on feature aggregation. Adding good aggregation features to non-aggregated features (e.g., player statistics) in score aggregation could then improve classification accuracy. Conversely, a score-aggregation classifier can be used as a strong baseline for pruning noninformative aggregate features, to reduce the expensive search through the aggregation space.

## VII. CONCLUSION

We considered link-based classification with continuous features of linked entities. Two basic approaches are aggregating features vs. aggregating classifier scores. For aggregating classifier scores using a combining rule, averaging-type rules provide consistent good baseline performance. On some datasets, they can be outperformed by extremal rules, such as maximum or noisy-or.

For feature aggregation, we investigated an approach to finding relevant aggregate features by applying a fixed set of aggregate operators to each original feature, then applying a standard classifier to the aggregate features. This use of feature aggregation outperforms score aggregation, even when matched against the best score aggregation rule selected a posteriori. While feature aggregation has well-known statistical problems, part of the reason for its superior performance is that score aggregation suffers from similar challenges. For instance, degree disparity is a challenge for both approaches, because in score aggregation, target instances with more links carry more weight than those with fewer.

*Future Work.* An open challenge for score aggregation is whether a good combining rule can be learned for a specific dataset. This question seems to be completely open.

A hybrid approach that combines score aggregation with feature aggregation could address the weaknesses of both approaches. For example good features could be found learning a model based on feature aggregation. Adding good aggregation

TABLE X
FEATURE AGGREGATION ACCURACIES

| **Method** | IMDb | Financial | NHL - S + Match | NHL - Season | PLG | NBA |
|---|---|---|---|---|---|---|
| Logistic Regression | **86.05%** | 84.93% | 88.59% | **54.67%** | 95.69% | **100.00%** |
| Logistic Regression + $\sigma$ | 85.57% | **87.50%** | **88.91%** | 54.47% | **96.55%** | **100.00%** |

TABLE XI
FEATURE AGGREGATION - SVM ACCURACIES

| **Method** | IMDb | Financial | NHL - S + Match | NHL - Season | PLG | NBA |
|---|---|---|---|---|---|---|
| SVM - Linear | **84.54%** | **76.84%** | 68.03% | **55.12%** | **95.69%** | **100.00%** |
| SVM - Quadratic | 71.99% | 72.06% | **68.94%** | 52.66% | 90.52% | 91.67% |
| SVM - Gaussian | 82.30% | 65.81% | 60.38% | 52.75% | 87.93% | **100.00%** |

TABLE XII
FEATURE AGGREGATION VS. SCORE AGGREGATION

| Dataset → | IMDb | | Financial | | NHL - S + Match | | NHL - Season | | PLG | | NBA | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method ↓ | Accuracy | F1-Measure | Accuracy | F1-Measure | Accuracy | F1-Measure | Accuracy | F1-Measure | Accuracy | F1-Measure | Accuracy | F1-Measure |
| Feature Aggregation | **86.05%** | **0.86** | **87.50%** | **0.87** | **88.91%** | **0.89** | 55.12% | **0.59** | **96.55%** | **0.96** | **100.00%** | **1.00** |
| Score Aggregation | 82.13% | 0.82 | 73.53% | 0.76 | 87.35% | 0.87 | **55.25%** | 0.46 | 93.10% | 0.93 | **100.00%** | **1.00** |

TABLE XIII
LEARNING TIME IN SECONDS

| | IMDb | Financial | NHL - S + Match | NHL - Season | PLG | NBA |
|---|---|---|---|---|---|---|
| Feature Aggregation | 0.083 | 0.894 | 0.523 | 0.126 | 3.288 | 0.034 |
| Score Aggregation | 14.074 | 6.314 | 0.567 | 0.229 | 0.430 | 0.004 |

features to non-aggregated features (e.g., player statistics) in score aggregation could then improve classification accuracy.

In sum, good accuracy in relational classification can be achieved with both feature aggregation and score aggregation (when used with averaging-type aggregators). The flexibility of feature aggregation to select good aggregates led to consistently superior classification performance in our experiments.

## REFERENCES


[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[2] K. Kersting and L. de Raedt, "Bayesian logic programming: Theory and tool," in *Introduction to Statistical Relational Learning*. MIT Press, 2007, ch. 10, pp. 291–318.

[3] J. Neville, D. Jensen, B. Gallagher, and R. Fairgrieve, "Simple estimators for relational bayesian classifiers," in *ICDM*. IEEE Computer Society, 2003, pp. 609–612.

[4] D. D. Jensen, J. Neville, and M. Hay, "Avoiding bias when aggregating relational data with degree disparity," in *Proceedings of the Twentieth International Conference on Machine Learning*, 2003, pp. 274–281.

[5] H. Chen, H. Liu, J. Han, and X. Yin, "Exploring optimization of semantic relationship graph for multi-relational Bayesian classification," *Decision Support Systems*, vol. 48, no. 1, pp. 112–121, 2009.

[6] B. Bina, O. Schulte, B. Crawford, Z. Qian, and Y. Xiong, "Simple decision forests for multi-relational classification," *Decision Support Systems*, vol. 54, pp. 1269–1279, 2013.

[7] S. Kramer, N. Lavrac, and P. Flach, "Propositionalization approaches to relational data mining," in *Relational Data Mining*. Springer, 2000, pp. 262–286.

[8] M.-A. Krogel and S. Wrobel, "Feature selection for propositionalization," in *DS '02: Proceedings of the 5th International Conference on Discovery Science*. London, UK: Springer-Verlag, 2002, pp. 430–434.

[9] A. Van Assche, C. Vens, H. Blockeel, and S. Dvzeroski, "First order random forests: Learning relational classifiers with complex aggregates," *Machine Learning*, vol. 64, no. 1, pp. 149–182, 2006.

[10] O. Kuzelka and F. Zelezný, "Block-wise construction of tree-like relational features with monotone reducibility and redundancy," *Machine Learning*, vol. 83, no. 2, pp. 163–192, 2011.

[11] A. Popescul and L. Ungar, "Feature generation and selection in multi-relational learning," in *Introduction to Statistical Relational Learning*. MIT Press, 2007, ch. 16, pp. 453–476.

[12] J. N. David Jensen, "Linkage and autocorrelation cause feature selection bias in relational learning (2002)," in *ICML*, 2002.

[13] D. Jensen and J. Neville, "Data mining in social networks," in *In National Academy of Sciences workshop on Dynamic Social Network Modeling and Analysis*, 2002.

[14] R. P. Schumaker, O. K. Solieman, and H. Chen, *Research in Sports Statistics*. Springer US, 2010.

[15] D. Poole, "First-order probabilistic inference," in *IJCAI*, 2003, pp. 985–991.

[16] A. Gelman and J. Hill, *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press, 2007.

[17] X. Yin, J. Han, J. Yang, and P. S. Yu, "Crossmine: Efficient classification across multiple database relations," in *ICDE*. IEEE Computer Society, 2004, pp. 399–410.

[18] "Mcfc analytics," uRL = http://www.mcfc.co.uk/Home/The%20Club/MCFC%20Analytics.

[19] "Selenium," uRL = http://www.seleniumhq.org/.

[20] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explorations*, vol. 11, no. 1, pp. 10–18, 2009.

[21] D. Jensen and J. Neville, "Linkage and autocorrelation cause feature selection bias in relational learning," in *ICML*, 2002.

[22] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann, 2005.

[23] L. Getoor and B. Taskar, *Introduction to Statistical Relational Learning*. MIT Press, 2007.