Conditional Weighted Transaction Aggregation for Credit Card Fraud Detection

Wee-Yong Lim, Amit Sachan, Vrizlynn Thing

Chapter 1

CONDITIONAL WEIGHTED TRANSACTION AGGREGATION FOR CREDIT CARD FRAUD DETECTION

Wee-Yong Lim, Amit Sachan and Vrizlynn Thing

Abstract Credit card fraud causes substantial losses to credit card companies and consumers. Consequently, it is important to develop sophisticated and robust fraud detection techniques that can recognize the subtle differences between fraudulent and legitimate transactions. Current fraud detection techniques mainly operate at the transaction level or account level. However, neither strategy is foolproof against fraud, leaving room for alternative techniques and improvements to existing techniques. Transaction-level approaches typically involve the analysis and aggregation of previous transaction data to detect credit card fraud. However, these approaches usually consider all the transaction attributes to be equally important. The conditional weighted transaction aggregation technique described in this paper addresses this issue by leveraging supervised machine learning techniques to identify fraudulent transactions. Empirical comparisons with existing transaction level methods and other transaction aggregation based methods demonstrate the effectiveness of the proposed technique.

Keywords: Credit card fraud, transaction analysis, aggregation, machine learning

1. Introduction

Credit card fraud causes significant losses to credit card companies and consumers. Security upgrades are constantly thwarted by fraudsters intent on stealing credit card data dumps. Given the prevalence of credit card theft, it is important to develop fraud detection techniques that can recognize the subtle differences between fraudulent and legitimate transactions.

Most credit card fraud detection techniques work at the transaction level and/or account level. Transaction-level fraud detection systems

classify individual transactions based on characteristics such as transaction amount and payment mode. A transaction-level model is limited because it does not consider previous transactions that could help identify fraudulent activities.

Account-level fraud detection systems model the normal behavior of consumers; a substantial deviation from normal behavior is an indicator of possible fraudulent activity. A typical example is an unexpected large online transaction from an account for which the vast majority of previous transactions were retail purchases at supermarkets. The account-level model can take into account the previous transactions of consumers, but it is limited by the impracticality of creating a global model based on all consumers.

Previous transactions are not considered in a transaction-level model because of the possibly large number of features that come into play. To address this drawback, Whitrow, *et al.* [9] proposed a transaction aggregation strategy that only aggregates transactions of the previous few days to boost fraud detection performance. The method works well compared with single transaction based methods. However, the primary limitation is that it treats all previous transactions as equal, ignoring the sequential nature of credit card transactions.

This paper describes the conditional aggregated transaction aggregation method, which leverages supervised machine learning techniques to determine fraudulent credit card transactions. It addresses the limitation of the transaction aggregation strategy of Whitrow, *et al.* [9] by performing aggregation in a weighted manner. In particular, the method modifies the weight of a previous transaction based on its distance from the current transaction.

2. Related Work

Fraud detection methods involving transaction data are generally divided into two categories. The first category includes methods for detecting outliers in transaction data. These methods generally use clustering algorithms to group transactions and identify outlier transactions from the known clusters. Predefined rules are typically used to classify transactions as fraudulent or legitimate.

The second category of methods analyze individual transactions using models trained by classifiers such as artificial neural networks [1, 5] and support vector machines [3]. This section reviews some of these methods before focusing on the transaction aggregation strategy. Interested readers are referred to Bhattacharyya, *et al.* [2] for a comprehensive survey of the various methods.

Panigrahi, *et al.* [6] have proposed a hybrid approach for credit card fraud detection. Their approach involves three steps. The first step is rule-based filtering in which rules are applied to filter fraudulent transactions based the deviations of transactions from normal user profiles. The rules, which focus on address mismatch and outlier detection, are based on characteristics such as transaction amount, inter-transaction time gap, billing address and shipping address. DBSCAN (density-based spatial clustering of applications with noise) [4] is used to detect outliers. In the second step, a data fusion technique (i.e., a Dempster-Shafer adder) is applied to combine evidence from the address mismatch and outlier detection rules. In the third step, Bayesian learning is employed to assess the suspicion score of a transaction. Another approach [8], which focuses on fraudulent insurance claim detection, uses superior boosted naive Bayes classification.

Yu and Wang [10] have proposed an outlier detection method that computes the distance of each transaction from every other transaction. A transaction whose sum of distances is greater than a threshold distance is considered to be an outlier. The method exhibits good accuracy, but incurs high overhead to compute and compare the distances for all the transactions.

A two-stage approach for credit card fraud detection, involving anomaly checking and comparisons against known fraud history, has been proposed by Sherly and Nedunchezhian [7]. K-means clustering of training data and special rules are used to discretize transaction attributes into categories to accommodate transaction variability. However, aside from the discretization, no multi-transaction behavioral information is extracted from the training transaction records to construct the training model.

Most of the work described in the literature uses credit card transaction data for fraud detection purposes. However, Chen, *et al.* [3] also incorporate questionnaire data collected from consumers to create user profiles. A major drawback of this method is that new questionnaires have to be created and analyzed when user behavior changes.

Whitrow, *et al.* [9] have proposed a novel approach using transaction aggregation as an alternative strategy to employing single transaction or behavioral models. They argue that transaction aggregation is a better fraud predictor than individual transaction and behavioral models (behavioral models do not consider global patterns). Experimental results using real-world data demonstrate the effectiveness of the method over different aggregation periods for several classification techniques (random forests, logistic regression, support vector machines, naive Bayes,

quadratic discriminant analysis, classification and regression trees, and k-nearest neighbors; the best results were obtained with random forests).

3. Fraud Detection Methodology

This section describes the proposed conditional weighted transaction aggregation method.

3.1 Data Attributes

The datasets used in this work comprised several transaction entries. Each entry had the attributes: consumer ID, credit card limit, transaction time, transaction amount, transaction mode ("online" or "pos" for point of sale) and address match result ("match," "mismatch" or "NA" for not applicable). Each transaction was labeled as "fraudulent" or "legitimate." The objective of credit card fraud detection is to predict the label of a new transaction given its attribute values and the relevant accumulated attribute values from previous transactions.

3.2 Transaction Aggregation Method

This section outlines the transaction aggregation method of Whitrow, *et al.* [9] and proceeds to describe the enhanced conditional weighted transaction aggregation method.

Transaction Aggregation Method. The method of Whitrow, *et al.* [9] computes the aggregate of the transaction amounts associated with different attribute values over the previous d days. The aggregate and the individual transaction attributes are used as features in fraud detection.

Let N be the total number of transactions during the previous d days and let $Amount(i)$ be the monetary amount of the i^{th} transaction. Assume that an attribute A considered for aggregation has m possible values (e.g., attribute "transaction mode" has two possible values "online" and "pos") with A_j denoting the j^{th} ($1 \leq j \leq m$) value. The aggregation involves a summation over all transactions over the previous d days that have similar values for attribute A; this helps capture the relevant behavioral characteristics over the aggregation period.

The aggregation of an attribute value A_j is given by:

$$Aggregation(A_j) = \sum_{i=1}^{N} Amount_i : A(i) = A_j \tag{1}$$

where $A(i)$ is the value of attribute A in the i^{th} transaction. The aggregation values obtained from Equation (1) are used as features in addition to the attributes listed in Section 3.1.

Conditional Weighted Transaction Aggregation Method. The method of Whitrow, *et al.* [9] considers all transactions during the previous d days to be equal, ignoring the sequence of transactions and examining only the state of the credit card account at a given point in time. Whitrow and colleagues acknowledge that excessive aggregation over a long time span could negatively affect the prediction model.

To address these concerns, we designed the conditional weighted transaction aggregation method to capture the sequential nature of credit card transactions. Weights are assigned to all previous transactions. The weights are inversely proportional to the distance between current and previous transactions. That is, for a set of N previous transactions, the i^{th} $(1 \leq i \leq N)$ transaction in the history (from the first transaction to the current transaction) is assigned a weight w_i that increases with increasing i. This gives more weight to recent transactions compared with older transactions that have lower correlations with the current transaction. Details about the weighting function are provided below.

As in the method of Whitrow, *et al.* [9], aggregation is only performed for attributes that are relevant to the current transaction. The idea is that these attributes, when aggregated, define a behavior characteristic of the consumer. For example, a card holder's transaction mode is regarded as a behavioral attribute while the address match indicator is not. This is because the former provides insight into the shopping habits of the consumer; the latter is dependent only on the merchant or transaction mode. In this work, previous transactions with the "pos" payment mode are not considered in the aggregation if the current transaction is performed "online." This helps reduce false predictions by filtering transactions with irrelevant attributes. The conditional selection of previous transactions is reflected in Equation (2) below using a binary multiplication factor C_j that is dependent on the relevance of the j^{th} attribute.

Finally, a factor is associated with each aggregated attribute based on the probability of the attribute value. Given a consumer c and probability $p_c(A_j)$ of an attribute value, the factor for its aggregated feature is $1 - p_c(A_j)$ (see Equation (2)). In other words, the higher the probability of an attribute value for the consumer, the lower the weight for each of its occurrences. Thus, if Consumer A engages in "online" transactions less frequently than Consumer B, the factor for the aggregation value for "online" transactions would be higher for Consumer A than for Con-

sumer B. In the experiments discussed in this paper, the probability of the payment mode was calculated based on all the transactions during the first four months in the training and test datasets.

Based on the preceding discussion, the aggregation for attribute value A_j is computed as:

$$Aggregation(A_j) = C_j * (1 - p_c(A_j)) * \sum_{i=1}^{N} w_i * Amount_i : A(i) = A_j \quad (2)$$

Weighting Function. Two weighting functions, the transaction gap function and the time gap function, are used to give more weight to recent transactions during aggregation. In the case of the transaction gap function, the weight w_i simply corresponds to the number of transactions between the i^{th} transaction and the current transaction. Specifically, the weight w_i is given by:

$$w_i \ = \ N - i \quad (3)$$

where N is the total number of transactions in the period under consideration.

The time gap function assigns weights based on the time difference between the current and latest transaction. For an aggregation period of d days with N previous transactions, the i^{th} transaction is given a weight of:

$$w_i \ = \ d \ - \ (time_N - time_i) \quad (4)$$

where $time_N - time_i$ is the time gap between the N^{th} and i^{th} transactions.

Experiments were performed using the two weighting functions, with the results favoring the time gap function under most conditions. This implies that closeness in time is a better indicator of fraudulent behavior than closeness based on transaction gap.

4. Experiments

This section describes the experimental setup and the experimental results.

4.1 Experimental Setup

Experiments were performed using datasets with simulated credit card transactions. Specifically, three datasets corresponding to low-dominant, middle-dominant and egalitarian distributions of spending profiles in the

Table 1. Distribution of spending profiles in simulated datasets.

Dataset	Spending Profiles		
	Low	**Medium**	**High**
1: Low-Dominant	0.80	0.15	0.05
2: Middle-Dominant	0.20	0.60	0.20
3: Egalitarian	0.33	0.33	0.33

population were generated. Table 1 shows that the distributions differ in the ratios of low, medium and high spending profiles in the population. Each dataset contained 200 credit card accounts, 100 of which were used for training and 100 for testing.

Table 2. Parameters for legitimate spending profiles.

Profile	**Transaction (tx)**	**Prob.**	**Mean**	**Std. Dev.**	**Min.**
Low	Low	0.90	30	10	10
	Medium	0.08	150	50	50
	High	0.02	500	200	200
Medium	Low	0.80	30	10	10
	Medium	0.15	150	50	50
	High	0.05	500	200	200
High	Low	0.60	30	10	10
	Medium	0.25	150	50	50
	High	0.12	700	300	200
	V. High	0.03	2,000	500	1,000

Each spending profile models the monetary amounts involved in low, medium or high transaction amounts using a Gaussian distribution. Table 2 presents the parameters for Gaussian simulated transaction (tx) amounts for legitimate spending profiles.

In addition to the legitimate transaction profiles, two fraudster profiles of equal probability were created. The first profile assumes that fraud is committed via infrequent, but large, transaction amounts. The second profile assumes smaller transaction amounts (possibly larger than legitimate transactions) with a wider spread over time. Table 3 presents the parameters for Gaussian simulated transaction (tx) amounts for fraudulent spending profiles.

The arrival rates of transactions were simulated as Poisson processes. Different arrival rates were used for legitimate and fraudulent transactions based on the following equation:

$$T_n = T_{n-1} - \log(U)/\lambda \tag{5}$$

Table 3. Parameters for fraudulent spending profiles.

Profile	Transaction (tx)	Prob.	Mean	Std. Dev.	Min.
Active	Low	0.1	700	300	200
	Medium	0.1	1,300	400	400
	High	0.8	2,200	600	500
Passive	Low	0.1	150	50	50
	Medium	0.8	500	100	200
	High	0.1	1,500	300	500

where T_n is the occurrence time of the n^{th} transaction, T_{n-1} is the occurrence time of the n–1^{th} transaction, U is a random value from [0,1] and λ is the arrival rate.

The training and testing data simulations were executed for periods of ten months, with fraudulent transactions occurring with a 15% probability during each of the previous five months. This reflects a likely real-world scenario where the number of legitimate transactions is generally much larger than the number of fraudulent transactions. Since most machine learning algorithms work best on balanced datasets, subsets of the legitimate samples were used during the training process. These subsets were chosen randomly and the performance values of the resulting classifiers were aggregated and averaged to assess the overall performance. Note that this sub-sampling process overcomes dataset imbalance and it is, arguably, necessary when dealing with large real-world datasets.

4.2 Performance Measurement

The misclassification rate, which measures the number of misclassified instances from among all the instances, is commonly used to assess classification performance. However, using the ratio of legitimate to fraudulent transactions is not recommended because it is highly imbalanced; also, different costs are associated with the misclassification of legitimate and fraudulent transactions. Whitrow, *et al.* [9] have suggested that, except for the case of a correctly classified legitimate transaction, costs are involved in all the other classification scenarios. Legitimate and fraudulent transactions that are classified as fraudulent incur costs of $C_{l/f}$ and $C_{f/f}$, respectively; the costs represent the work required to conduct further investigations. Likewise, a fraudulent transaction that is wrongly classified as legitimate incurs a cost of $C_{f/l}$, which corresponds to the amount of money lost in the transaction.

Due to the different costs associated with different classifications, we used a loss function to measure classification performance instead of the typical misclassification rate. In particular, we used the same costs and cost function as Whitrow, *et al.* [9]. Based on an analysis of real credit card data, the cost values $C_{l/f} = 1$, $C_{f/f} = 1$, $C_{f/l} = 100$ and $C_{l/l} = 0$ were employed.

Given n_f fraudulent transactions and n_l legitimate transactions, let $n_{l/f}$ be the number of legitimate transactions classified as fraudulent, $n_{f/f}$ be the number of fraudulent transactions classified as fraudulent and $n_{f/l}$ be the number of fraudulent transactions classified as legitimate. Then, the total cost function for the classification is given by:

$$C = \frac{C_{l/f} * n_{l/f} + C_{f/f} * n_{f/f} + C_{f/l} * n_{f/l}}{C_{f/l} * n_f + C(l/f) * n_l} \tag{6}$$

where the numerator is the cost of the current classification and the denominator is the maximum cost that can be incurred. Thus, the cost associated with a transaction is bounded between 0 and 1, and a method with a lower cost is considered to be better than a method with a higher cost.

4.3 Comparison of Methods

Four methods were compared based on their costs. The methods were: (i) transaction based (Tx) (i.e., without any aggregation information); (ii) simple aggregation (SA); (iii) weighted aggregation with transaction gap as the weighting function (TxG); and (iv) weighted aggregation with time gap as the weighting function (TG). The simple aggregation method is a minor adaptation of the method of Whitrow and colleagues [9] for the (simpler) simulated dataset used in this work. Note that all the methods, except for the transaction based method, require the aggregation of features over some time period.

The performance of a method clearly depends on the aggregation period. Experimentation with aggregations ranging from one day to seven days indicated that the best performance is achieved with aggregations of three to five days for all three datasets. Thus, the results reported for all the methods involving aggregated features are based on the average performance over aggregation periods of three to five days.

Several common classification techniques were used in the experiments: random forests, naive Bayes, AdaBoost, logistic regression and k-nearest neighbors (kNN). To obtain a balanced dataset, a random subset of the legitimate transaction was selected before training and testing each classifier. The selection, training and testing process was iterated

Table 4. Costs ($\times 10^3$) of methods with different classifiers (Dataset 1).

Classifier	***TG***	***TxG***	***SA***	***Tx***
Random Forests	14.897	**13.765**	14.309	17.404
Naive Bayes	**32.396**	41.776	37.878	35.252
AdaBoost	**24.741**	25.785	25.283	32.034
Logistic Regression	**18.304**	20.468	18.909	24.527
kNN	**33.307**	37.052	36.049	40.641
Average	**24.729**	27.769	26.486	29.972

Table 5. Costs ($\times 10^3$) of methods with different classifiers (Dataset 2).

Classifier	***TG***	***TxG***	***SA***	***Tx***
Random Forests	23.946	23.897	**23.599**	28.375
Naive Bayes	61.065	78.934	79.131	**53.992**
AdaBoost	34.858	35.692	**34.108**	51.293
Logistic Regression	**25.999**	27.582	26.018	34.993
kNN	**43.174**	48.318	46.054	48.241
Average	**37.808**	42.885	41.782	43.379

Table 6. Costs ($\times 10^3$) of methods with different classifiers (Dataset 3).

Classifier	***TG***	***TxG***	***SA***	***Tx***
Random Forests	**21.128**	21.547	21.383	27.275
Naive Bayes	81.688	106.843	102.699	**70.959**
AdaBoost	**33.199**	33.578	33.459	43.613
Logistic Regression	**33.210**	35.449	36.256	40.239
kNN	47.808	51.832	50.810	**46.000**
Average	**43.407**	49.850	48.921	45.617

ten times to obtain the average performance for each classification technique.

Tables 4, 5 and 6 show the performance of different methods and classification techniques for Datasets 1, 2 and 3, respectively. The cost function defined by Equation (6) is used as the performance measure. For purposes of clarity, the actual costs are multiplied by a factor of 10^3 to produce the table values. Note that the costs for the datasets can be very different due to the different transaction amounts in each dataset. Therefore, the costs of different classifiers should be compared only for the same dataset.

The performance of weighted aggregation with transaction gap weighting is poor compared with the other aggregation methods. In fact, it is the worst aggregation method based on its average cost over all the classification techniques. Moreover, its performance is mixed when compared with that of the basic transaction based method.

On the other hand, the time gap weighted aggregation method performed significantly better than the other methods. Although it was not the best performing method for some classification techniques in the case of Datasets 2 and 3, the costs incurred by the time gap weighted aggregation method are consistently the second lowest among the tested methods, and its average cost is the lowest for all methods. This provides empirical evidence of performance stability of the time gap weighted aggregation method across the various classification techniques and datasets.

4.4 Transaction vs. Time Gap Weighting

As discussed above, the weights used during the aggregation process are meant to emphasize the relative importance of newer transactions over older transactions. The results obtained for the datasets clearly show that weighted aggregation helps improve the classification performance when an appropriate weighting function is used. However, it is also apparent that a poor weighting function such as the transaction gap weighting function results in performance that is poorer than the simple aggregation method. This could be because the transaction gap weighting function is unable to capture the different arrival rates between legitimate and fraudulent (passive and active) transactions. Although transaction gap weighting assigns an adjusted weight to each previous transaction during aggregation, the weights are independent of the transaction time and merely capture the transaction sequence.

To conduct a "fairer" analysis of the two weighted aggregation methods, the same experiments were conducted on (additional) Datasets 4, 5 and 6 with the same parameters as Datasets 1, 2 and 3, respectively, except that adjustments were made for equal arrival rates for all legitimate, passive and active fraudulent profiles. This restricted the differences between legitimate and fraudulent transactions to their different spending profiles. Table 7 through 9 show the results of the second set of experiments.

In the original experiments (Tables 4, 5 and 6), the average cost incurred by the transaction gap weighted aggregation method is 13.52% higher than the cost incurred by the time gap weighted aggregation method. However, when the arrival rates are similar, the results in Tables 7, 8 and 9 show that the difference in the average cost between the

Table 7. Costs ($\times 10^3$) of methods with different classifiers (Dataset 4).

Classifier	***TG***	***TxG***	***SA***	***Tx***
Random Forests	19.265	**18.835**	19.377	24.711
Naive Bayes	**42.817**	49.484	49.851	60.221
AdaBoost	32.747	28.289	29.656	**27.747**
Logistic Regression	24.912	**23.887**	24.881	33.418
kNN	45.010	47.160	46.353	**43.030**
Average	**32.950**	33.531	34.024	37.825

Table 8. Costs ($\times 10^3$) of methods with different classifiers (Dataset 5).

Classifier	***TG***	***TxG***	***SA***	***Tx***
Random Forests	**19.873**	21.47	19.926	24.711
Naive Bayes	**58.403**	68.359	77.104	60.221
AdaBoost	39.052	37.552	**36.675**	46.642
Logistic Regression	**26.915**	28.563	27.053	38.776
kNN	**40.704**	42.788	43.025	40.776
Average	**36.990**	39.747	40.757	42.225

Table 9. Costs ($\times 10^3$) of methods with different classifiers (Dataset 6).

Classifier	***TG***	***TxG***	***SA***	***Tx***
Random forests	24.375	**21.864**	23.248	27.520
Naive Bayes	99.868	103.931	113.325	**87.364**
AdaBoost	**42.552**	46.730	44.579	53.542
Logistic regression	38.179	39.184	**36.683**	45.936
kNN	46.925	49.983	47.888	**40.776**
Average	**50.380**	52.338	53.145	51.028

two weighted aggregation methods drops to just 4.37%. This confirms that capturing the arrival rates of transactions can significantly improve classification performance. Nevertheless, the superior performance of the time gap weighted aggregation method over the transaction gap weighted aggregation method suggests that weights assigned based on the temporal differences between previous transactions are better than the assignments based on the sequence of previous transactions.

5. Conclusions

Transaction aggregation is a promising strategy for detecting fraudulent credit card transactions. The experimental results in this paper (as well as previous work in the area) demonstrate that aggregation based methods work better than transaction based methods. This is mainly due to the ability of aggregation based methods to capture additional evidence from previous transactions.

The main contribution of this paper is an enhanced aggregation based method that incorporates weights for all the previous transactions in an aggregated period. Transaction gap weighting considers the number of transactions between current and previous transactions while time gap weighting considers the temporal difference between current and previous transactions. The experimental results demonstrate that time gap weighting intensifies the importance of recent transactions over older transactions, thereby enhancing credit fraud detection. However, care should be taken when choosing weighting functions because weights that are directly dependent on transaction time result in better classification performance compared with weights that are based on the sequence of previous transactions.

References

[1] E. Aleskerov, B. Freisleben and B. Rao, Cardwatch: A neural network based database mining system for credit card fraud detection, *Proceedings of the IEEE/IAFE Conference on Computational Intelligence for Financial Engineering*, pp. 220–226, 1997.

[2] S. Bhattacharyya, J. Sanjeev, K. Tharakunnel and J. Westland, Data mining for credit card fraud: A comparative study, *Decision Support Systems*, vol. 50(3), pp. 602–613, 2011.

[3] R. Chen, S. Luo, X. Liang and V. Lee, Personalized approach based on SVM and ANN for detecting credit card fraud, *Proceedings of the International Conference on Neural Networks and the Brain*, vol. 2, pp. 810–815, 2005.

[4] M. Ester, H. Kriegel, J. Sander and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

[5] S. Ghosh and D. Reilly, Credit card fraud detection with a neural network, *Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences*, vol. 3, pp. 621–630, 1994.

[6] S. Panigrahi, A. Kundu, S. Sural and A. Majumdar, Credit card fraud detection: A fusion approach using Dempster-Shafer theory and Bayesian learning, *Information Fusion*, vol. 10(4), pp. 354–363, 2009.

[7] K. Sherly and R. Nedunchezhian, BOAT adaptive credit card fraud detection system, *Proceedings of the IEEE International Conference on Computational Intelligence and Computing Research*, 2010.

[8] S. Viaene, R. Derrig and G. Dedene, A case study of applying boosting naive Bayes to claim fraud diagnosis, *IEEE Transactions on Knowledge and Data Engineering*, vol. 16(5), pp. 612–620, 2004.

[9] C. Whitrow, D. Hand, P. Juszczak, D. Weston and N. Adams, Transaction aggregation as a strategy for credit card fraud detection, *Data Mining and Knowledge Discovery*, vol. 18(1), pp. 30–55, 2009.

[10] W. Yu and N. Wang, Research on credit card fraud detection model based on distance sum, *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 353–356, 2009.