AFRL-IF-RS-TR-2006-307
Final Technical Report
October 2006

GENEWAYS FOR BIOCOMPUTING

Columbia University

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO FINAL REPORT

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

AFRL-IF-RS-TR-2006-307 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
THOMAS E. RENZ Work Unit Manager	JAMES A. COLLINS Deputy Chief, Advanced Computing Division Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
OCT 06	Final	May 05 – May 06

4. TITLE AND SUBTITLE

GENEWAYS FOR BIOCOMPUTING

5a. CONTRACT NUMBER
FA8750-04-2-0123

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
61101E

6. AUTHOR(S)
Andrey Rzhetsky

5d. PROJECT NUMBER
S277

5e. TASK NUMBER
00

5f. WORK UNIT NUMBER
TC

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Columbia University
Office of Research Support
1210 Amsterdam Avenue
Manhattan New York 10027

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/IFTC
525 Brooks Road
Rome New York 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2006-307

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA#06-723

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The immense growth in the volume of research literature and experimental data in the field of molecular biology calls for efficient automatic methods to capture and store information. In recent years, several groups have worked on specific problems in this area, such as automated selection of articles pertinent to molecular biology, or automated extraction of information using natural-language processing, information visualization, and generation of specialized knowledge bases for molecular biology. GeneWays is an integrated system that combines several such subtasks. It analyzes interactions between molecular substances, drawing on multiple sources of information to infer a consensus view of molecular networks. GeneWays is designed as an open platform, allowing researchers to query, review, and critique stored information.

15. SUBJECT TERMS
BioInformatics, Genetics, Computational Biology, BioComputing, Literature Search

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Thomas E. Renz
U	U	U	UL	37	19b. TELEPHONE NUMBER (*Include area code*)

Table of Contents

List of Figures

.

List of Tables

1. Summary

Over the past two years we significantly increased recall of GeneWays pipeline processing, developed, tested and applied tools for automated data cleaning (AI curation) of the produced database. We also increased significantly (> 50%) the volume of textual information processed by the GeneWays pipeline, applied AI curator tools to the newly generated database and incorporated automatically curated data into a new version of a GeneWays database.

2. Introduction

Picture a tribe of bright, but ignorant, cave people trying to understand the work of a modern car by analyzing a collection of damaged cars produced by various makers. After many hours of hard manual labor, the cave people disassemble the cars into myriad small parts. Some parts are damaged, whereas some are intact. A few interact with each other, while others do not. Some pieces are different in different cars, yet apparently have the same function. The leap to understanding the whole from knowing the parts requires compilation of many pieces of information into a comprehensive "computable" model. Researchers in the field of molecular biology are in a situation similar to that of the junkyard cave people, save that they are contemplating a collection of diverse pieces of cellular machinery—the number of those cellular components is way greater than the number of parts in a typical car—the number of nodes in human molecular networks is measured in hundreds of thousands when all substances (genes, RNAs, proteins, and other molecules) are considered together. These numerous substances can be in turn present or absent in dozens of cell types in humans—clearly, the complexity is too great to yield to manual analysis.

The information overload in molecular biology is a mere example of the status common to all fields of the current science and culture: An ever-strengthening avalanche of novel data and ideas overwhelms specialists and non-specialists alike, unavoidably fragments knowledge, and makes enormous chunks of knowledge invisible/inaccessible to those who desperately need it.

The help of relieving the information overload may come from the text-miners who can automatically extract and catalogue facts described in books and journals.

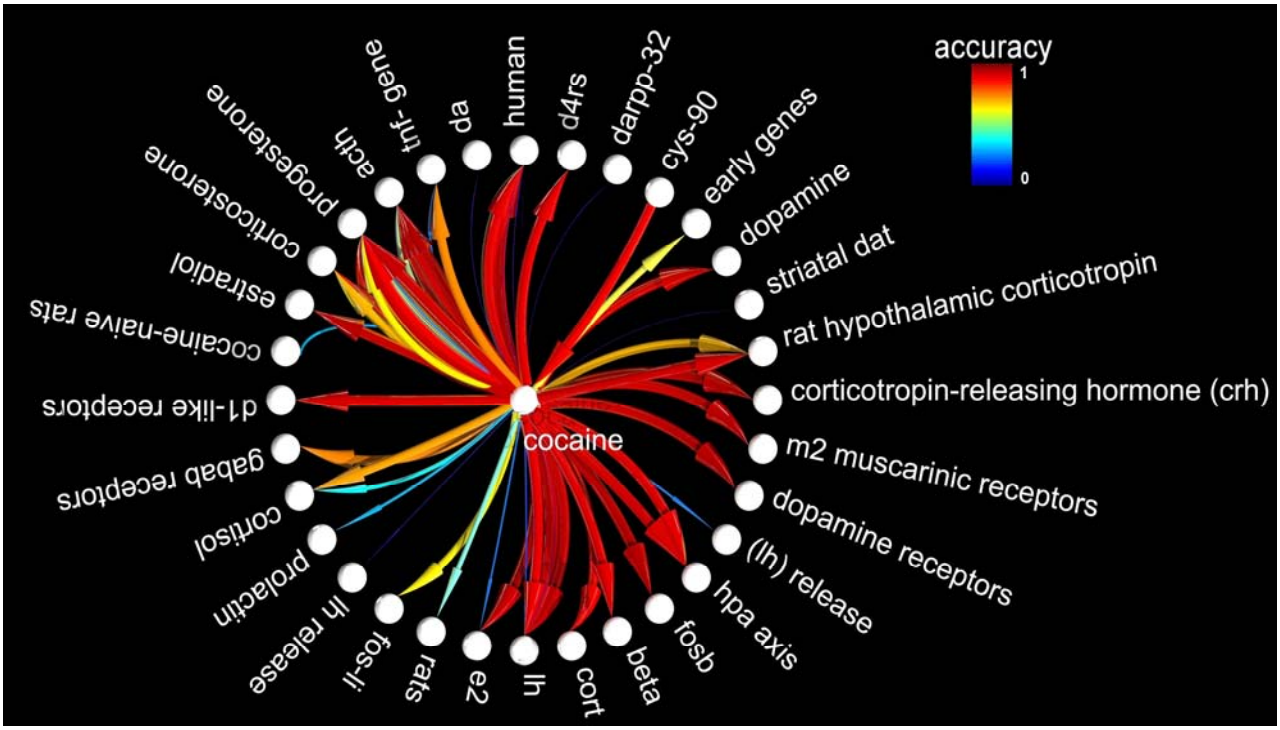

Figure 1. Cocaine : The predicted accuracy of individual text-mined facts involving semantic relation.

In Figure 1, each directed arc from an entity *A* to an entity *B* should be interpreted as a statement "*A stimulates B*", where, for example, A is cocaine and B is progesterone. The predicted accuracy of individual statements is indicated both in color and in width of the corresponding arc. Note that, for example, the relation between cocaine and progesterone was derived from multiple sentences, and different instances of extraction output had markedly different accuracy. Altogether we collected 3, 910 individual facts involving cocaine. Because the same fact can be repeated in different sentences, only 1, 820 facts out of 3, 910 were unique. The facts cover 80 distinct semantic relations, out of which *stimulate* is just one example.

3. Methods, Assumptions, and Procedures

Information extraction uses computer-aided methods to recover and structure meaning that is locked in natural-language texts. The assertions uncovered in this way are amenable to computational processing that approximates human reasoning. In the special case of biomedical applications, the texts are represented by books and research articles, and the extracted meaning comprises diverse classes of facts, such as relations between molecules, cells, anatomical structures, and maladies.

Unfortunately, the current tools of information extraction produce imperfect, noisy results. Although even imperfect results are useful, it is highly desirable for most applications to have the ability to rank the text-derived facts by the confidence in the quality of their extraction (as we did for relations involving *cocaine*, see Figure 1). We focus on automatically extracted statements about molecular interactions, such as *small molecule A binds protein B*, *protein B activates gene C*, or *protein D phosphorylates small molecule E*. (In the following description we refer to phrases that represent biological entities (such as *small molecule A*, *protein B*, and *gene C*) as *terms*, and to biological relations between these entities (such as *activate* or *phosphorylate*) as *relations* or *verbs*.)

Several earlier studies have examined aspects of evaluating the quality of text-mined facts. For example, Sekimizu *et al.* and Ono *et al.* attempted to attribute different confidence values to different verbs that are associated with extracted relations, such as *activate*, *regulate*, and *inhibit* [1,2]. Thomas *et al.* proposed to attach a quality value to each extracted statement about molecular interactions [3], although the researchers did not implement the suggested scoring system in practice. In an independent study [4], Blaschke and Valencia used word-distances between biological terms in a given sentence as an indicator of the precision of extracted facts. In our present analysis we applied several machine-learning techniques to a large training set of $98,679$ manually evaluated examples (pairs of extracted facts and corresponding sentences) to design a tool that mimics the work of a human curator who manually cleans the output of an information-extraction program.

Approach

Our goal was to design a tool that could be used with any information-extraction system developed for molecular biology. In this study, our training data came from the GeneWays project (specifically, GeneWays 6.0 database, [5,6]) and thus our approach is biased toward relationships that are captured by that specific system. We believe that the spectrum of relationships represented in the GeneWays ontology is sufficiently broad that our results will prove useful for other information-extraction projects.

Our approach followed the path of supervised machine-learning. First, we generated a large training set of facts that were originally gathered by our information-extraction system, and then manually labeled as “correct” or “incorrect” by a team of human curators. Second, we used a battery of machine-learning tools to imitate computationally the work of the human evaluators. Third, we split the training set into ten parts, so that we could evaluate the significance of performance differences among the several competing machine-learning approaches.

Methods

Training data

With the help of a text-annotation company, *ForScience Inc.*, we generated a training set of approximately $45,000$ multiple-annotated unique facts, or almost $100,000$ independent evaluations. These facts were originally extracted by the GeneWays pipeline, then were annotated by biology-savvy doctoral-level curators as "correct" or "incorrect," referring to quality of information extraction. Examples of automatically extracted relations, sentences corresponding to each relation, and the labels provided by three evaluators are shown in Table 1.

Table 1. A sample of sentences that were used as an input to automated information extraction

Sentence [Source]	Extracted relation	Evaluation (Confidence)
NIK *binds* to **Nck** in cultured cells.[8]	nik *bind* nck	Correct (High)
One is that **presenilin** is *required for* the proper trafficking of **Notch** and APP to their proteases, which may reside in an intracellular compartment. [9]	presenilin *required for* notch	Correct (High)
Serine 732 *phosphorylation* of **FAK** by **Cdk5** is important for microtubule organization, nuclear movement, and neuronal migration. [10]	cdk5 *phosphorylate* fak	Correct (High)
Histogram quantifying the percent of **Arr2** *bound* to **rhodopsin**-containing membranes after treatment with blue light (B) or blue light followed by orange light (BO). [11]	arr2 *bind* rhodopsin	Correct (Low)
It is now generally accepted that a shift from monomer to dimer and **cadherin** clustering *activates* classic **cadherins** at the surface into an adhesively competent conformation. [12]	cadherin *activate* cadherins	Correct (Low)
Binding of G to **CSP** was four times greater than binding to **syntaxin**. [13]	csp *bind* syntaxin	Incorrect (Low)
Treatment with NEM applied with **cGMP** made *activation* by **cAMP** more favorable by about 2.5 kcal/mol. [14]	camp *activate* cgmp	Incorrect (Low)
This matrix is likely to consist of **actin** filaments, as similar filaments can be *induced* by **actin**-stabilizing toxins (O. S. et al., unpublished data). [15]	actin *induce* actin	Incorrect (High)
A ligand-gated *association* between **cytoplasmic domains** of **UNC5** and DCC family receptors converts netrin-induced growth cone attraction to repulsion. [16]	cytoplasmic domains *associate* unc5	Incorrect (High)

In Table 1, a sample of sentences that were used as an input to automated information extraction (the first column), biological relations extracted from these sentences (either correctly or incorrectly, the second column), and the corresponding evaluations provided by 3 human experts (the third column). A high-confidence label corresponds to a perfect agreement among all experts; a low-confidence label indicates that one of the experts disagreed with the other two. Clearly, automated information extraction can be associated with a loss of detail of meaning, as is in the case of *cadherin activates cadherins* example (sentence 5).

Each extracted fact was evaluated by one, two, or three different curators. The complete evaluation set comprised 98,679 individual evaluations performed by four different people, so most of the statement–sentence pairs were evaluated multiple times, with each person evaluating a given pair at most once. In total, 13,502 statement/sentence pairs were evaluated by just one person, 10,457 by two people, 21,421 by three people, and 57 by all four people. Examples of both high inter-annotator agreement and low-agreement sentences are shown in Table 1.

Table 2. List of annotation choices available to the evaluators.

Term level	
	Upstream term is a junk substance
	Action is incorrect biologically
	Downstream term is a junk substance
Relation level	
	Correctly extracted
	Sentence is hypothesis, not fact
	Unable to decide
	Incorrectly extracted
	Incorrect upstream
	Incorrect downstream
	Incorrect action type
	Missing or extra negation
	Wrong action direction
	Sentence does not support the action
Sentence level	
	Wrong sentence boundary

In Table 2, the term “action” refers to the type of the extracted relation. For example, in statement *A binds B* “binds” is the *action*, “A” is the *upstream term*, and “B” is the *downstream term*. Action direction is defined as *upstream to downstream*, and “junk substance” is an obviously incorrectly identified term/entity.

The statements in the training data set were grouped into chunks; each chunk was associated with a specific biological project, such as analysis of interactions in *Drosophila melanogaster*. Pair-wise agreement between evaluators was high (92%) in most chunks, with the exception of a chunk of

$5,271$ relations where agreement was only 74%. These relatively low-agreement evaluations were not included in the training data for our analysis.

To facilitate evaluation, we developed a Sentence Evaluation Tool implemented in Java programming language by Mitzi Morris and Ivan Iossifov. This tool presented to an evaluator a set of annotation choices regarding each extracted fact; the choices are listed in Table 2. The tool also presented in a single window the fact itself and the sentence it was derived from. In the case where a broader context was required for the judgment, the evaluator had a choice to retrieve the complete journal article containing this sentence by clicking a single button on the program interface.

For convenience in representing the results of manual evaluation, we computed an evaluation score for each statement as follows. Each sentence–statement score was computed as a sum of the scores assigned by individual evaluators; for each evaluator, -1 was added if the expert believed that the presented information was extracted incorrectly, and $+1$ was added if he or she believed that extraction was correct. For a set of three experts, this method permitted four possible scores: $3(1,1,1)$, $1(1,1,-1)$, $-1(1,-1,-1)$, and -3. Similarly, for just two experts, the possible scores are $2(1,1)$, $0(1,-1)$, and $-2(-1,-1)$.

Computational methods

Machine-learning algorithms

General framework

The objects that we want to classify, the fact–sentence pairs, have complex properties. We wanted to place each of the objects into one of two classes, *correct* or *incorrect*. In the training data, each extracted fact was matched to a unique sentence from which it was extracted, even though multiple sentences can express the same fact and a single sentence can contain multiple facts. The i^{th} object (the i^{th} fact–sentence pair) comes with a set of known features or properties that we encoded into a feature vector, $\mathbf{F}_i$:

$$\mathbf{F}_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,n}). \quad (1)$$

In the following description we used C to indicate the random variable that represents class (with possible values $c_{correct}$ and $c_{incorrect}$), and F to represent a $1 \times n$ random vector of feature values (also often called *attributes*), such that F_j is the j^{th} element of F. For example, for fact *p53 activates JAK*, feature F_1 would have value 1 because the upstream term *p53* is found in a dictionary derived from the GenBank database [19]; otherwise, it would have value 0.

Full Bayesian inference

The full Bayesian classifier assigns the i^{th} object to the k^{th} class if the posterior probability $P(C = c_k \mid F = \mathbf{F}_i)$ is greater for the k^{th} class than for any alternative class. This posterior probability is computed in the following way (a re-stated version of Bayes' theorem).

$$P(C = c_k \mid F = \mathbf{F}_i) = P(C = c_k) \times \frac{P(F = \mathbf{F}_i \mid C = c_k)}{P(F = \mathbf{F}_i)}. \quad (2)$$

In real-life applications, we estimate probability $P(F = \mathbf{F}_i \mid C = c_k)$ from the training data as a ratio of the number of objects that belong to the class c_k *and* have the same set of feature values as specified by the vector $\mathbf{F}_i$ to the total number of objects in class c_k in the training data.

In other words, we estimate the conditional probability for every possible value of the feature vector F for every value of class C. Assuming that all features can be discretized, we have to estimate

$$(v_1 \times v_2 \times \ldots v_n - 1) \times m \quad (3)$$

parameters, where v_i is the number of discrete values observed for the i^{th} feature and m is the number of classes.

Clearly, even for a space of only 20 binary features the number of parameters that we would need to estimate is $(2^{20} - 1) \times 2 = 2,097,150$, which exceeds several times the number of data points in our training set.

Naïve Bayes classifier

The most affordable approximation to the full Bayesian analysis is the Naïve Bayes classifier. It is based on the assumption of conditional independence of features:

$$P(F = \mathbf{F}_i \mid C = c_k) = P(F_1 = f_{i,1} \mid C = c_k)$$

$$\times P(F_2 = f_{i,2} \mid C = c_k) \ldots$$

$$\times P(F_n = f_{i,n} \mid C = c_k). \quad (4)$$

Obviously, we can estimate $P(F_j = f_{i,j} \mid C = c_k)$'s reasonably well with a relatively small set of training data, but the assumption of conditional independence (Equation 4) comes at a price: the

Naïve Bayes classifier is usually markedly less successful in its job than are its more sophisticated relatives.

In an application with m classes and n features (given that the i^{th} feature has v_i admissible discrete values), a Naïve Bayes algorithm requires estimation of $m \times \sum_{i=1,n} (v_i - 1)$ parameters (which value, in our case, is equal to $4,208$).

Middle ground between the full and Naïve Bayes: Clustered Bayes

We can find an intermediate ground between the full and Naïve Bayes classifiers by assuming that features in the random vector F are arranged into groups or clusters, such that all features within the same cluster are dependent on one another (conditionally on the class), and all features from different classes are conditionally independent. That is, we can assume that the feature random vector (F) and the observed feature vector for the i^{th} object ($\mathbf{F}_i$) can be partitioned into sub-vectors:

$$F = (\Phi_1, \Phi_2, \ldots, \Phi_M), \text{and} \qquad (5)$$

$$\mathbf{F}_i = (\mathbf{f}_{i,1}, \mathbf{f}_{i,2}, \ldots, \mathbf{f}_{i,M}), \quad (6)$$

respectively, where Φ_j is the j^{th} cluster of features; $\mathbf{f}_{i,j}$ is the set of values for this cluster with respect to the i^{th} object, and M is the total number of clusters of features.

The Clustered Bayes classifier is based on the following assumption about conditional independence of *clusters* of features:

$$P\left(F = \mathbf{F}_i \mid C = c_k\right) = P\left(\Phi_1 = \mathbf{f}_{i,1} \mid C = c_k\right)$$

$$\times P\left(\Phi_2 = \mathbf{f}_{i,2} \mid C = c_k\right) \ldots$$

$$\times P\left(\Phi_M = \mathbf{f}_{i,M} \mid C = c_k\right). \qquad (7)$$

We tested two versions of the Clustered Bayes classifier: one version used all 68 features (Clustered Bayes 68) with a coarser discretization of feature values; another version used a subset of 44 features (Clustered Bayes 44) but allowed for more discrete values for each continuous-valued feature, see legend to Figure 2.

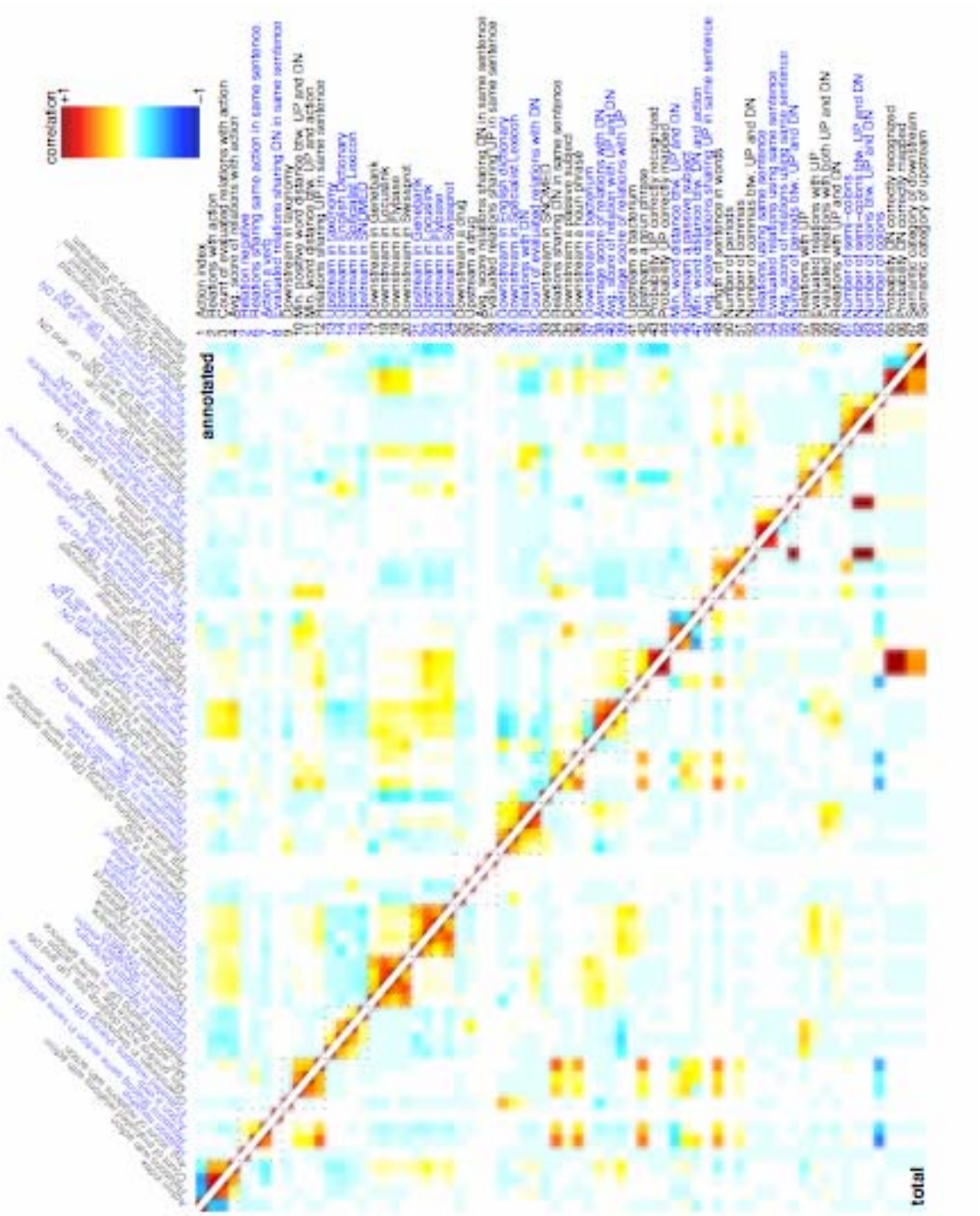

Figure 2. The correlation matrix for the features used by the classification algorithms.

In Figure 2, the half-matrix below the diagonal was derived from analysis of the whole GeneWays 6.0 database; the half-matrix above the diagonal represents a correlation matrix estimated from only the manually annotated data set. The white dotted lines outline clusters of features, suggested by analysis of the annotated data set; we used these clusters in implementation of the Clustered

Bayes classifier. We used two versions of the Clustered Bayes classifier: with all 68 features (Clustered Bayes 68), and with a subset of only 44 features, but higher number of discrete values allowed for non-binary features (Clustered Bayes 44). The Clustered Bayes 44 classifier did not use features 1, 6, 7, 8, 9, 12, 27, 28, 31, 34, 37, 40, 42, 47, 48, 49, 52, 54, 55, 60, 62, 63, and 65.

Linear and quadratic discriminants

Another method that can be viewed as an approximation to full Bayesian analysis is Discriminant Analysis invented by Sir Ronald A. Fisher [20]. This method requires no assumption about conditional independence of features; instead, it assumes that the conditional probability $P\left(F=\mathbf{F}_i \mid C=c_k\right)$ is a multivariate normal distribution.

$$P\left(F=\mathbf{F}_i \mid C=c_k\right)=\frac{e^{-\frac{1}{2}\left(\mathbf{F}_i-\mu_k\right)\mathbf{V}_k^{-1}\left(\mathbf{F}_i-\mu_k\right)}}{\sqrt{\left(2\pi\right)^n \mid \mathbf{V}_k \mid}}, \quad (8)$$

where n is the total number of features/variables in the class-specific multivariate distributions. The method has two variations. The first, *Linear Discriminant Analysis*, assumes that different classes have different mean values for features (vectors μ_k), but the same variance-covariance matrix, $\mathbf{V}=\mathbf{V}_k$ for all k (see Suppl. Note 7). In the second variation, *Quadratic Discriminant Analysis* (QDA), the assumption of the common variance-covariance matrix for all classes, is relaxed, such that every class is assumed to have a distinct variance-covariance matrix, $\mathbf{V}_k$.

In this study we present results for QDA; the difference from the linear discriminant analysis was insignificant for our data (not shown). In terms of the number of parameters to estimate, QDA uses only two symmetrical class-specific covariance matrices and the two class-specific mean vectors. For 68 features the method requires estimation of $2\times(68\times 69)/2+2\times 68=4{,}828$ parameters.

Maximum-entropy method

The current version of the maximum-entropy method was formulated by E.T. Jaynes [21,22]; the method can be traced to earlier work by J. Willard Gibbs. The idea behind the approach is as follows. Imagine that we need to estimate a probability distribution from an incomplete or small data set—this problem is the same as that of estimating the probability of the class given the feature vector, $P\left(C=c_k \mid F=\mathbf{F}_i\right)$, from a relatively small training set. Although we have no hope of estimating the distribution completely, we can estimate with sufficient reliability the first (and, potentially, the second) moments of the distribution. Then, we can try to find a probability distribution that has the same moments as our unknown distribution and the highest possible Shannon's entropy—the intuition behind this approach being that the maximum-entropy distribution will minimize unnecessary assumptions about the unknown distribution. The maximum-entropy distribution with constraints imposed by the first-order feature moments alone (the mean values of features) is known to have the form of an exponential distribution [23]:

$$P\left(C = c_k \mid F = \mathbf{F}_j\right) = \frac{\exp\left(-\sum_{i=1}^{n} \lambda_{i,k} f_{j,i}\right)}{\sum_{l=1}^{2} \exp\left(-\sum_{i=1}^{n} \lambda_{i,l} f_{j,i}\right)}, \quad (9)$$

and the maximum-entropy distribution for the case when both the first- and the second-order moments of the unknown distribution are fixed has the form of a multidimensional normal distribution [23]. The conditional distribution that we are trying to estimate can be written in the following exponential form:

$$P\left(C = c_k \mid F = \mathbf{F}_j\right) = \frac{\exp\left(-\sum_{i=1}^{n} \lambda_{i,k} f_{j,i} - \sum_{x=1}^{n} \sum_{y=x}^{n} \nu_{x,y,k} f_{j,x} f_{j,y}\right)}{\sum_{l=1}^{2} \exp\left(-\sum_{i=1}^{n} \lambda_{i,l} f_{j,i} - \sum_{x=1}^{n} \sum_{y=x}^{n} \nu_{x,y,l} f_{j,x} f_{j,y}\right)}. \quad (10)$$

Parameters $\lambda_{i,k}$'s and $\nu_{x,y,k}$'s are k-class-specific weights of individual features and feature pairs, respectively, and in principle can be expressed in terms of the first and second moments of the distributions. The values of parameters in Equations 9 and 10 are estimated by maximizing the product of probabilities for the individual training examples.

We tested two versions of the maximum-entropy classifier. MaxEnt 1 uses only information about the first moments of features in the training data (Equation 9); MaxEnt 2 uses the set of all individual features and the products of feature pairs (Equation 10). To select the most informative pairs of features we used a mutual information approach, as described in the subsection dealing with classification features.

For two classes (*correct* and *incorrect*) and 68 features MaxEnt 1 requires estimation of 136 parameters. In contrast, MaxEnt 2 requires estimation of $4,828$ parameters: weight parameters for all first moments for two classes, plus weights for the second moments for two classes. MaxEnt 2-v is a version of MaxEnt 2 classifier where the squared values of features are not used, so that the classifier requires estimation of only $4,692$ weight parameters.

Feed-forward neural network

A typical feed-forward artificial neural network is a directed acyclic graph organized into three (or more) layers. In our case, we chose a three-layered network, with a set of nodes of the *input layer*, $\{x_i\}_{i=1,..,N_x}$, nodes of the *hidden layer*, $\{y_j\}_{j=1,..,N_y}$, and a single node representing the *output layer*, z_1, see Figure 2. The number of input nodes, N_x, is determined by the number of features used in the analysis (68 in our case). The number of hidden nodes, N_y, determines both the network's expressive power and its ability to generalize. Too small a number of hidden nodes makes a

simplistic network that cannot learn from complex data. Too large a number makes a network that tends to overtrain—that works perfectly on the training data, but poorly on new data. We experimented with different values of N_y and settled on $N_y = 10$.

The values of the input nodes, $\{x_i\}_{i=1,..,N_x}$, are feature values of the object that we need to classify. The value of each node, y_j, in the hidden layer is determined in the following way:

$$y_j = F(w_{j,1}x_1 + w_{j,2}x_2 + \ldots + w_{j,N_x}x_{N_x}), \quad (11)$$

where $F(x)$ is a hyperbolic tangent function that creates an S-shaped curve:

$$F(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \qquad (12)$$

and $\{w_{j,k}\}$ are weight parameters. Finally, the value of the output node, z_1 is determined as a linear combination of the values of all hidden nodes:

$$z_1 = a_1 y_1 + a_2 y_2 + \ldots + a_{N_y} y_{N_y}, \quad (13)$$

where $\{a_k\}$ are additional weight parameters. We trained our network, using a back-propagation algorithm [24], to distinguish two classes, *correct* and *incorrect*, where positive values of z_1 corresponded to the class *correct*.

The feed-forward neural network that we used in our analysis can be thought of as a model with $N_x \times N_y + N_y$ parameters (690 in our case).

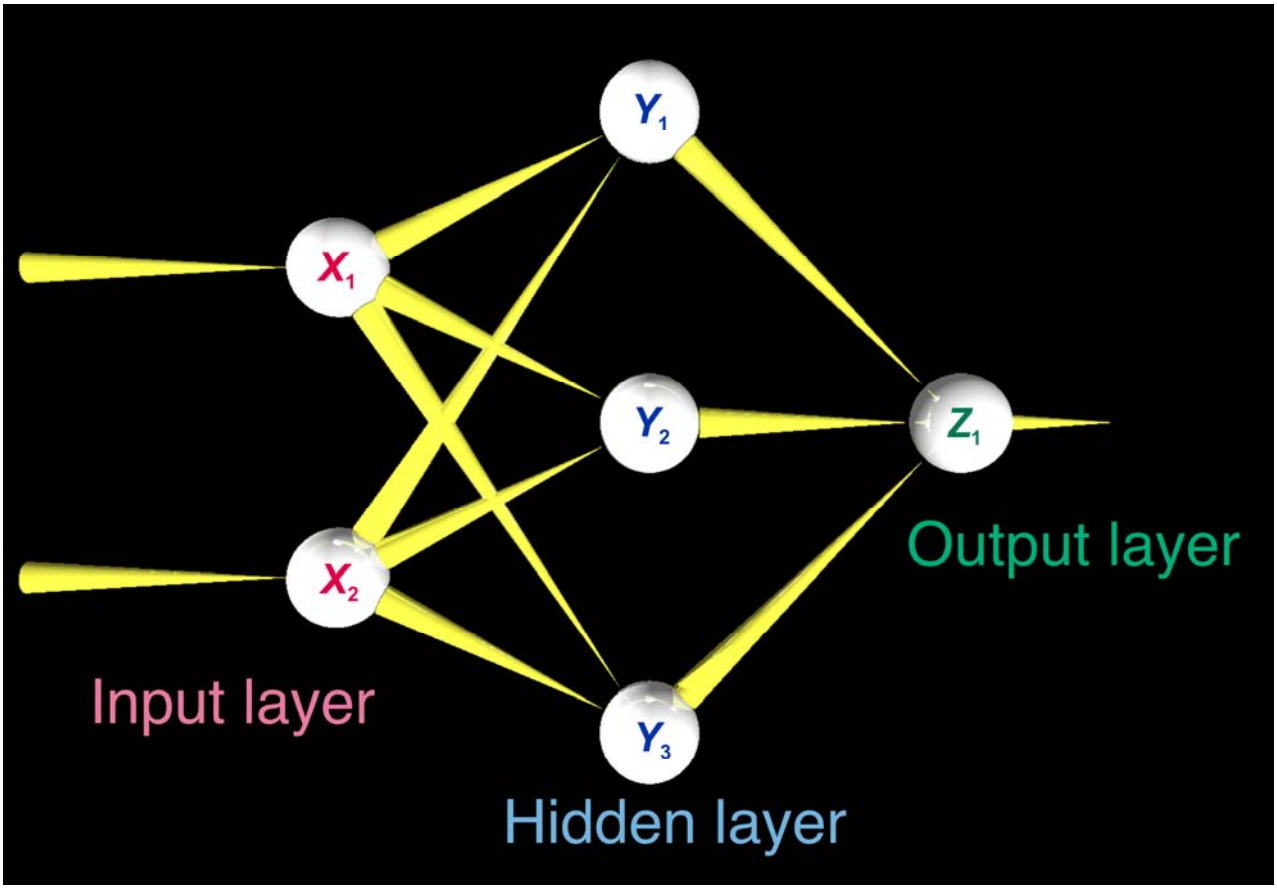

Figure 3. A hypothetical three-layered feed-forward neural network.

In Figure 3, we used a similar network with 68 input units (one unit per classification feature) and 10 hidden-layer units.

Support vector machines

The Support Vector Machines (SVM, [25,26]) algorithm solves a binary classification problem by dividing two sets of data geometrically, by finding a hyperplane that separates the two classes of objects in the training data in an optimum way (maximizing the margin between the two classes).

The SVM is a *kernel*-based algorithm, where the kernel is an inner product of two feature vectors (function/transformation of the original data). In this study, we used three of the most popular kernels: the linear, polynomial and Rbf (radial basis function) kernels. The linear kernel $K^L(\mathbf{x}_1,\mathbf{x}_2) = <\mathbf{x}_1,\mathbf{x}_2>$ is simply the inner product of the two input feature vectors; an SVM with the linear kernel searches for a class-separating hyperplane in the original space of the data. Using a polynomial kernel, $K_d^P(\mathbf{x}_1,\mathbf{x}_2) = (1+<\mathbf{x}_1,\mathbf{x}_2>)^d$, is equivalent to transforming the data into a higher-dimensional space and searching for a separating plane there. Finally, using an Rbf kernel, $K_g^{\text{Rbf}}(\mathbf{x}_1,\mathbf{x}_2) = e^{-g<\mathbf{x}_1-\mathbf{x}_2>^2}$, corresponds to finding a separating hyperplane in an infinite-dimensional space.

In most real-world cases the two classes cannot be separated perfectly by a hyperplane, and some classification errors are unavoidable. SVM algorithms use the C-parameter to control the error rate during the training phase (if the error is not constrained, the margin of every hyperplane can be extended infinitely). In this study, we used the default values for the C-parameter suggested by the SVM Light tool. Table 3 lists the SVM models and C-parameter values that we used in this study.

Table 3. Parameter values used for various SVM classifiers in this study.

Model	Kernel	Kernel parameter	C-parameter
SVM (OSU SVM)	Linear		1
SVM-t0 (SVM Light)	Linear		1
SVM-t1-d2	Polynomial	$d=2$	0.3333
SVM-t1-d3	Polynomial	$d=3$	0.1429
SVM-t2-g0.5	Rbf	$g=0.5$	1.2707
SVM-t2-g1	Rbf	$g=1$	0.7910
SVM-t2-g2	Rbf	$g=2$	0.5783

The output of an SVM analysis is not probabilistic, but there are tools to convert an SVM classification output into "posterior probabilities," see chapter by J. Platt in [27]. (A similar comment is applicable to the artificial neural network.)

The number of support vectors used by the SVM classifier depends on the size and properties of the training data set. The average number of (1×68-dimensional) support vectors used in 10 cross-validation experiments was $12,757.5$, $11,994.4$, $12,092$, $12,289.9$, $12,679.7$, and $14,163.8$, for SVM, SVM-t1-d2, SVM-t1-d3, SVM-t2-g0.5, SVM-t2-g1, and SVM-t2-g2 classifiers, respectively. The total number of data-derived values (which we loosely call "parameters") used by the SVM in our cross-validation experiments was therefore, on average, between $827,614$ and $880,270$ for various SVM versions.

Table 4. Machine learning methods used in this study and their implementations.

Method	Implementation	URL	Number of parameters
Naïve Bayes	this study, WEKA	http://www.cs.waikato.ac.nz/ml/weka/	4,208
Clustered Bayes 68	this study	N/A	276,432
Clustered Bayes 44	this study	N/A	361,270
Discriminant Analysis	this study	N/A	4,828
SVM	OSU SVM Toolbox for Matlab	http://sourceforge.net/projects/svm	827,614
*SVM-t**	SVM light [28]	http://svmlight.joachims.org/	827,614 to 880,270
Neural Network	Neural Network toolbox for Matlab	N/A	690
MaxEnt 1	Maximum Entropy Modeling Toolkit for Python and C++	http://homepages.inf.ed.ac.uk /s0450736/maxent_toolkit.html	136
MaxEnt 2	same as the MaxEnt 1	same as the MaxEnt 1	4,828
MaxEnt 2-v	same as the MaxEnt 1	same as the MaxEnt 1	4,692
Meta-Classifier	OSU SVM Toolbox for Matlab	http://sourceforge.net/projects/svm	> 11,560

Meta-method

We implemented the meta-classifier on the basis of the SVM algorithm (linear kernel with $C = 1$) applied to predictions (converted into probabilities that the object belongs to the class *correct*) provided by the individual "simple" classifiers. The meta-method used $1,445$ support vectors (1×7 -dimensional), in addition to combined parameters of the seven individual classifiers used as input to the meta-classifier.

Implementation

A summary of the sources of software used in our study is shown in Table 4.

Features used in our analysis

We selected 68 individual features covering a range of characteristics that could help in the classification, see Table 5. To capture the flow of information in a molecular interaction graph (the edge direction), in each extracted relation we identified an "upstream term" (corresponding to the graph node with the outgoing directed edge) and a "downstream term" (the node with the incoming directed edge): for example, in the phrase "*JAK* phosphorylates *p53*," *JAK* is the upstream term, and *p53* is the downstream term. Features in the group *keywords* represent a list of tokens that may signal that the sentence is hypothetical, interrogative, negative, or that there is confusion in the relation extraction (e.g. the particle "by" in passive-voice sentences). We eventually abandoned *keywords* as we found them to be uninformative features, but they are still listed for the sake of completeness.

Table 5. List of the features that we used in the present study.

Group of features	Feature(s)	Values	Number of features
Dictionary look-ups	{Upstream, downstream} term can be found in {GeneBank, NCBI taxonomy, LocusLink, SwissProt, FlyBase, drug list, disease list, Specialist Lexicon, Bacteria, English Dictionary}	Binary	20
Word metrics	Length of the sentence (word count)	Positive integer	1
	Distance between the upstream and the downstream term	Integer	1
	Minimum non-negative word distance between the upstream and the downstream term	Non-negative Integer	1
	Distance between the upstream term and the action	Integer	1
	Distance between the downstream term and the action	Integer	1
Previous scores	Average score of relationships with the same {upstream term, downstream term, action}	Real	3
	Count of evaluated relationships with the same {upstream term, downstream term, action}	Positive integer	3
	Total count of relationships with the same {upstream term, downstream term, action}	Positive integer	3
	Average score of relationships that share the same pair of upstream and downstream terms	Real	1
	Total count of evaluated relationships that share the same pair of upstream and downstream terms	Positive integer	1

	Total count of relationships with both the same upstream and downstream terms	Positive integer	1
	Number of relations extracted from the same sentence	Positive integer	1
	Number of evaluated relations extracted from the same sentence	Positive integer	1
	Average score of relations from the same sentence	Real	1
	Number of relations sharing upstream term in same sentence	Positive integer	1
	Number of evaluated relations sharing upstream term in the same sentence	Positive integer	1
	Average score of relations sharing upstream term in same sentence	Real	1
	Relations sharing downstream term in the same sentence	Positive integer	1
	Evaluated relations sharing downstream term in the same sentence	Positive integer	1
	Average score of relations sharing downstream term in the same sentence	Real	1
	Number of relations sharing same action in the same sentence	Positive integer	1
	Number of evaluated relations sharing action in the same sentence	Positive integer	1
	Average score of relations sharing action in the same sentence	Real	1
Punctuation	Number of {periods, commas, semi-colons, colons} in the sentence	Non-negative integer	4
	Number of {periods, commas, semi-colons, colons} between upstream and downstream terms	Non-negative integer	4
Terms	Semantic sub-class category of the {upstream, downstream} term	Integer	2
	Probability that the {upstream, downstream} term has been correctly recognized	Real	2
	Probability that the {upstream, downstream} term has been correctly mapped	Real	2
Part-of-speech tags	{Upstream, downstream} term is a noun phrase	Binary	2
	Action is a verb	Binary	1
Other	Relationship is negative	Binary	1
	Action index	Positive integer	1
	Keyword is present	Binary	(not used)

Dictionary lookups are binary features indicating absence or presence of a term in a specific dictionary. Previous scores are the average scores that a term or an action has in other relations evaluated. Term- recognition probabilities are generated by the GeneWays pipeline and reflect the likelihood that a term had been correctly recognized and mapped. Sharing of the same action (verb) by two different facts within the same sentence occurs in phrases such as *A* and *B* were shown to phosphorylate *C*. In this example, two individual relations, *A phosphorylates C* and *B phosphorylates C*, share the same verb, *phosphorylate*. Semantic categories are entities (semantic classes) in the GeneWays ontology (e.g. *gene*, *protein*, *geneorprotein*). Part-of-speech tags were generated by the Maximum Entropy tagger, MXPOST [29].

To represent the second-order features (pairs of features), we defined a new feature as a product of the normalized values of two features. We obtained the normalized values of features by subtracting the mean value from each feature value, then dividing the result by the standard deviation for this feature.

After a number of feature-selection experiments for the MaxEnt 2 method we settled on using all second-order features.

Separating data into training and testing: Cross-validation

To evaluate the success of our classifiers we used a 10-fold cross-validation approach, where we used $\frac{9}{10}$ of data for training and $\frac{1}{10}$ for testing. More precisely, given a partition of the manually evaluated data into 10 equal portions, we created 10 different pairs of training–test subsets, so that 10 distinct testing sets put together covered the whole collection of the manually evaluated sentences. We then used 10 training–test set pairs to compare all algorithms.

Comparison of methods: Receiver operating characteristic (ROC) scores

To quantify and compare success of the various classification methods we used receiver operating characteristic (ROC) scores, also called areas under ROC curve [32].

An ROC score is computed in the following way. All test-set predictions of a particular classification method are ordered by the decreasing quality score provided by this method; for example, in the case of the Clustered Bayes algorithm, the quality score is the posterior probability that the test object belongs to the class *correct*. The ranked list is then converted into binary predictions by applying a decision threshold, T. All test objects with a quality score above T are classified as *correct* and all test objects with low-than-threshold scores are classified as *incorrect*. The ROC score is then computed by plotting the proportion of true-positive predictions (in the test set we know both the correct label and the quality score of each object) against false-positive predictions for the whole spectrum of possible values of T, then integrating the area under the curve obtained in this way, see Figure 4.

The ROC score is an estimate of the probability that the classifier under scrutiny will label correctly a pair of statements, one of which is from the class *correct* and one from the class *incorrect* [32]. A completely random classifier therefore would have an ROC score of 0.5, whereas a hypothetical perfect classifier would have an ROC score of 1. It is also possible to design a classifier that performs less accurately than would one that is completely random; in this case the ROC score is less than 0.5, which indicates that we can improve the accuracy of the classifier by simply reversing all predictions.

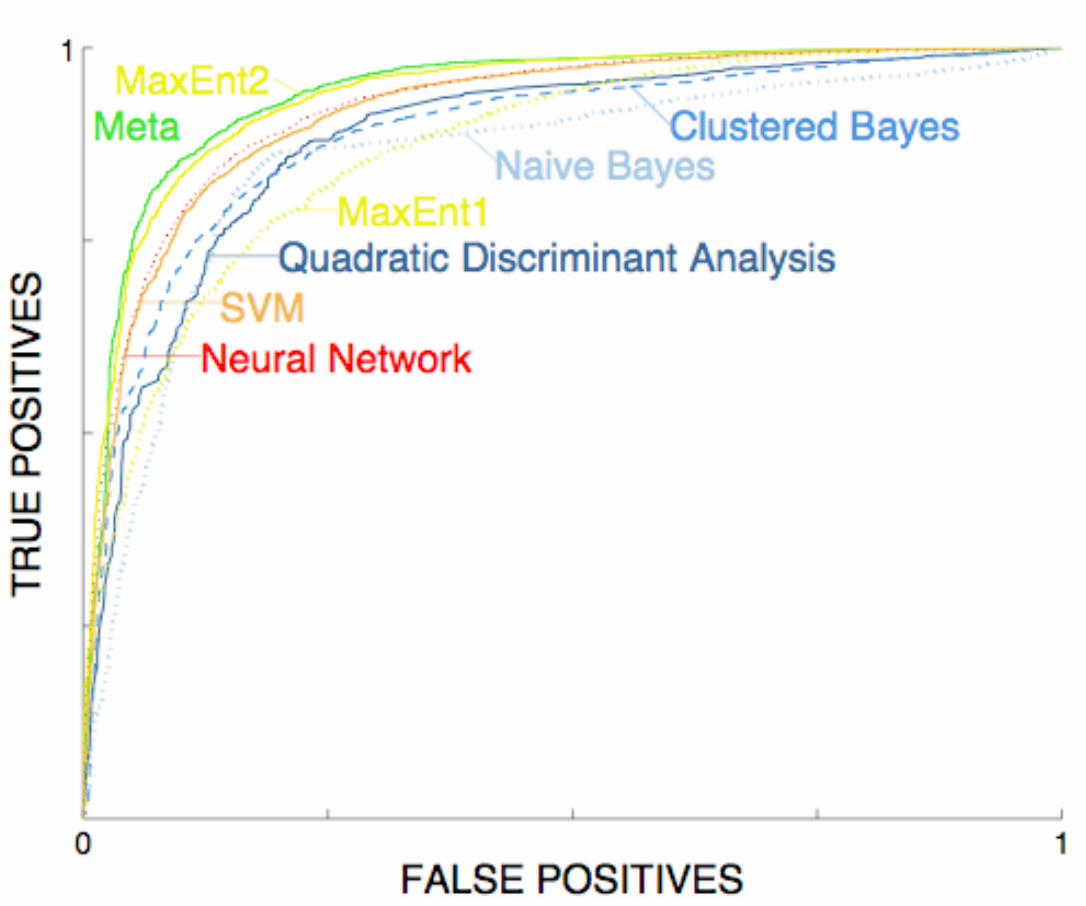

Figure 4. Receiver-operating characteristic (ROC) curves for the classification methods that we used in the present study.

In Figure 4, we show only the linear-kernel SVM and the Clustered Bayes 44 ROC curves to avoid excessive data clutter.

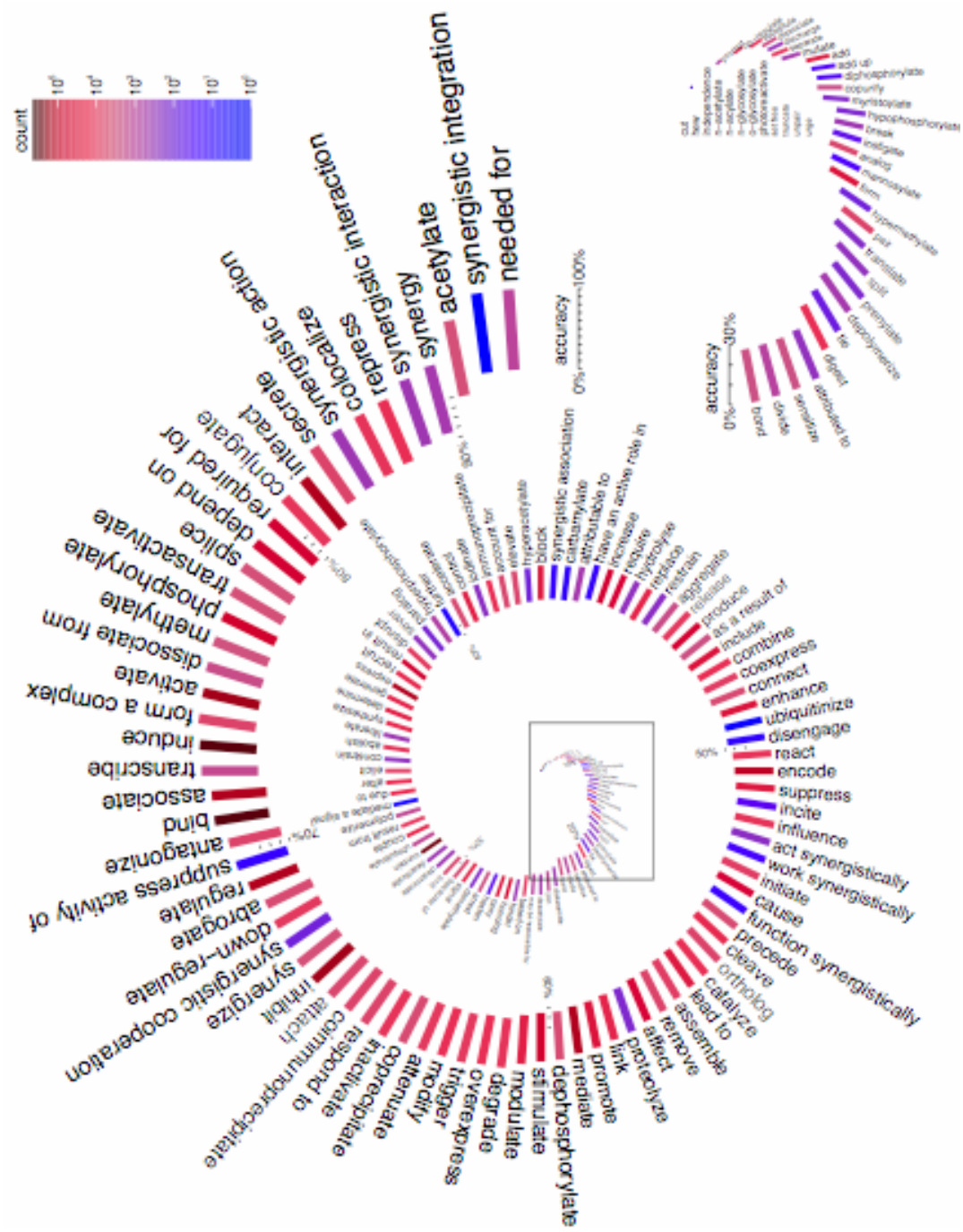

Figure 5. Accuracy of the raw (non-curated) extracted relations in the GeneWays 6.0 database.

The accuracy was computed by averaging over all individual specific information extraction examples manually evaluated by the human curators. The plot compactly represents both the per-relation accuracy of the extraction process (indicated with the length of the corresponding bar) and the abundance of the corresponding relations in the database (represented by the bar color). There are relations extracted with a high precision; there are also many noisy relationships. The database accuracy was markedly increased by the automated curation outlined in this study, see Figure 6.

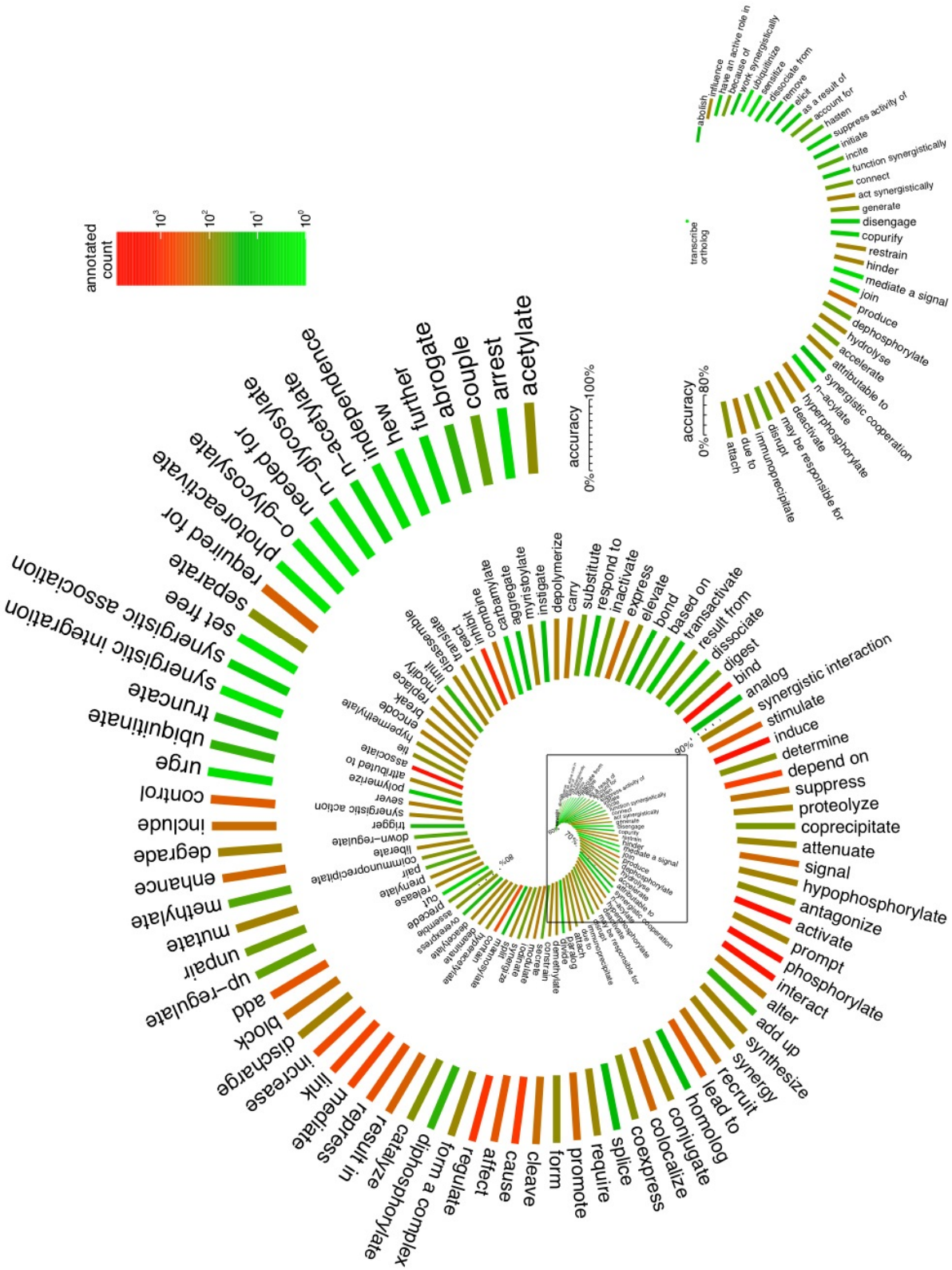

Figure 6. Accuracy and abundance of the extracted and automatically curated relations.

Figure 6 represents both the per-relation accuracy after both information extraction and automated curation were done. Accuracy is indicated with the length of the relation-specific bars, while the abundance of the corresponding relations in the manually curated data set is represented by color. Here, the MaxEnt 2 method was used for the automated curation. The results shown correspond to a score-based decision threshold set to zero; that is, all negative-score predictions were treated as "incorrect." An increase in the score-based decision boundary can raise the precision of the output at the expense of a decrease in the recall, see Figure 9.

4. Results

The raw extracted facts produced by our system are noisy. Although many relation types are extracted with accuracy above 80%, and even above 90% (see Figure 2), there are particularly noisy verbs/relations that bring the average accuracy of the "raw" data to about 65%. Therefore, additional purification of text-mining output, either computational or manual, is indeed important.

The classification problem of separating correctly and incorrectly extracted facts appears to belong to a class of easier problems. Even the simplest Naïve Bayes method had an average ROC score of 0.84, which more sophisticated approaches surpassed to reach almost 0.95. Judging by the average ROC score, the quality of prediction increased in the following order of methods: Clustered Bayes 68, Naïve Bayes, MaxEnt 1, Clustered Bayes 44, Quadratic Discriminant Analysis, artificial neural network, support vector machines, and MaxEnt 2/MaxEnt 2-v (see Table 6). The Meta-method was always slightly more accurate than MaxEnt 2, as explained in legend to Table 6 and shown in Figure 4.

Table 6 provides a somewhat misleading impression that MaxEnt 2 and MaxEnt 2-v are *not* significantly more accurate than their closest competitors (the SVM family), because of the overlapping confidence intervals. However, when we trace the performance of all classifiers in individual cross-validation experiments (see Figure 7) it becomes clear that MaxEnt 2 and MaxEnt 2-v outperformed their rivals in every cross-validation experiment. The SVM and artificial neural network methods performed essentially identically, and were always more accurate than three other methods: QDA, Clustered Bayes 44, and MaxEnt 1. Finally, the performance of the Clustered Bayes 68 and the Naïve Bayes methods was reliably the least accurate of all methods studied.

It is a matter of both academic curiosity and of practical importance to know how the performance of our artificial intelligence curator compares to that of humans. If we define the *correct* answer as a majority-vote of the three human evaluators(see Table 4), the average accuracy of MaxEnt 2 is slightly lower than, but statistically indistinguishable from humans (at the 99% level of significance, see Table 4; capital letters "A," "L," "S," and "M" hide the real names of the human evaluators). If, however, in the spirit of Turing's test of machine intelligence [17], we treat the MaxEnt 2 algorithm on an equal footing with the human evaluators, compute the average over predictions of all four anonymous evaluators, and compare the quality of the performance of each evaluator with regard to the average, MaxEnt 2 always performs slightly more accurately than one of the human evaluators. (In all cases we compared performance of the algorithm on data that was not used for its training; see Tables 4 and 5.)

Table 6. Comparison of the performance of human evaluators and of the MaxEnt 2 algorithm.

Evaluator	Correct	Incorrect	Accuracy [99% CI]
Batch A			
A.	10,981	208 (11,189)	0.981410 [0.978014 0.984628]
L.	10,547	642 (11,189)	0.942622 [0.936902 0.948253]
M.	10,867	322 (11,189)	0.971222 [0.967111 0.975244]
MaxEnt 2	10,537	652 (11,189)	0.941728 [0.935919 0.947359]
Batch B			
A.	9,796	430 (10,226)	0.957950 [0.952767 0.962938]
M.	9,898	328 (10,226)	0.967925 [0.963329 0.972325]
S.	9,501	725 (10,226)	0.929102 [0.922453 0.935556]
MaxEnt 2	9,379	847 (10,226)	0.917172 [0.910033 0.924115]

The first column in Table 6 lists all evaluators (four human evaluators, “A”, “L”, “M”, and “S”, and the MaxEnt 2 classifier). The second column gives the number of correct answers (with respect to the gold standard) produced by each evaluator. The third column shows the number of incorrect answers for each evaluator out of the total number of examples (in parentheses). The last column shows the accuracy and the 99%confidence interval for the accuracy value. The gold standard was defined as the majority among three human evaluators (examples with uncertain votes were not considered, so each evaluator’s vote was either strictly negative or strictly positive). Batches A and B were evaluated by different sets of human evaluators. We computed the binomial confidence intervals at the α-level of significance ($\alpha \times 100\%$ CI) by identifying a pair of parameter values that separate areas of approximately $\frac{(1-\alpha)}{2}$ at each distribution tail.

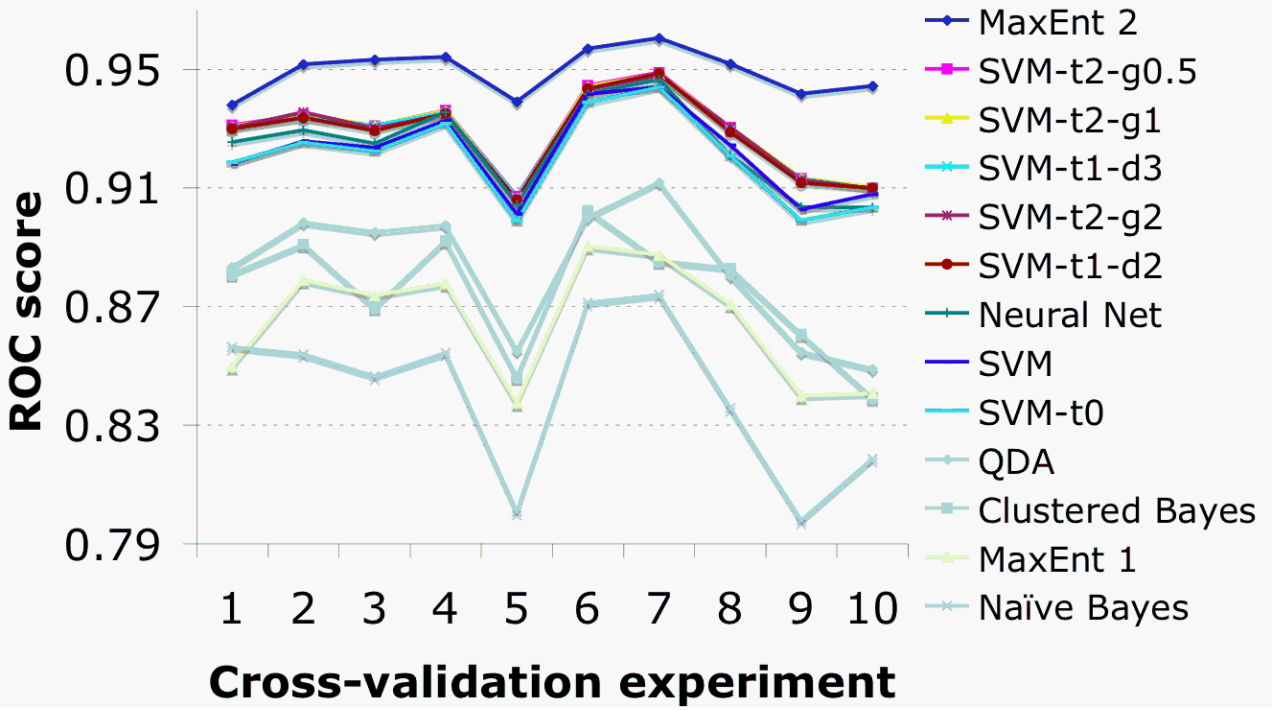

Figure 7. Ranks of all classification methods used in this study in 10 cross-validation experiments.

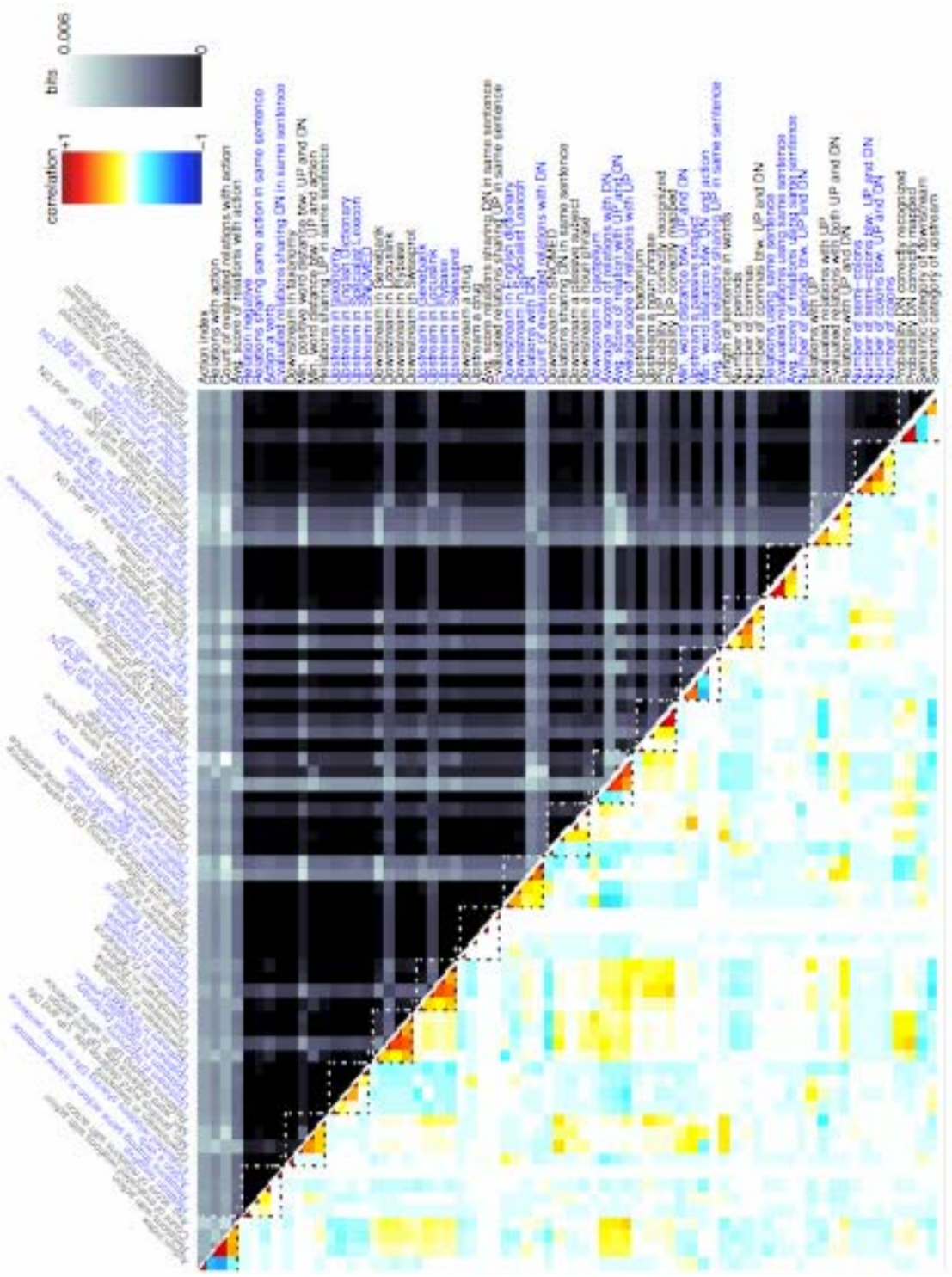

Figure 8. Comparison of a correlation matrix for the features

Figure 8 is a comparison of a correlation matrix for the features (colored half of the matrix) computed using only the annotated set of data and a matrix of mutual information between all feature pairs and the statement class (correct or incorrect). The plot indicates that a significant

amount of information critical for classification is encoded in pairs of weakly correlated features. The white dotted lines outline clusters of features, suggested by analysis of the annotated data set; we used these clusters in implementation of the Clustered Bayes classifier

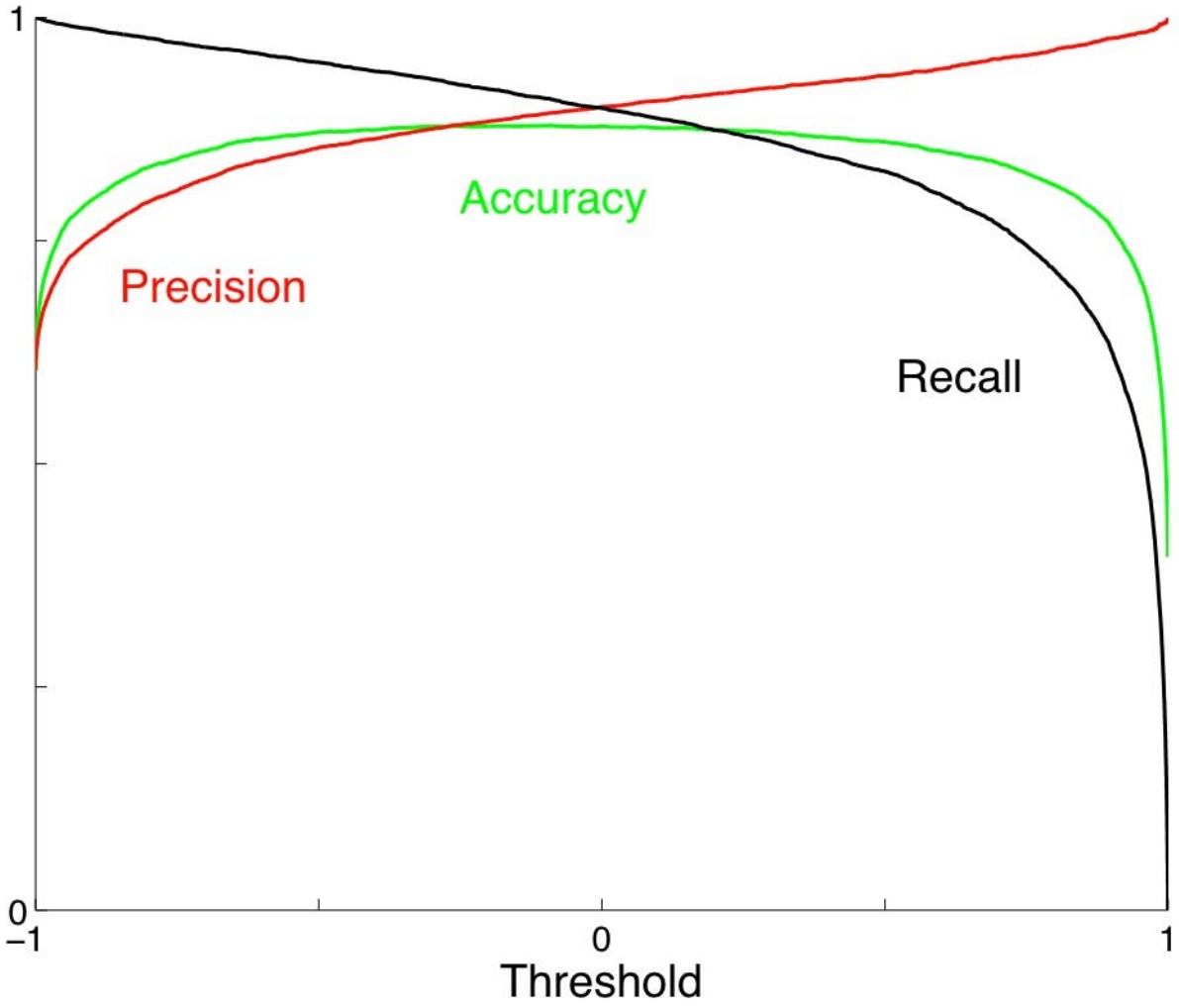

Figure 9. Values of precision, recall and accuracy of the MaxEnt 2 classifier plotted against the corresponding log-scores provided by the classifier.

The optimum accuracy was close to 88%, and attained at score threshold slightly above 0. We can improve precision at the expense of accuracy: For example, by setting the threshold score to 0.6702 we can bring the overall database precision to 95%, which would correspond to a recall of 77.91% and to an overall accuracy of 84.18%.

Table 7. Comparison of human evaluators and a program that mimicked their work.

Evaluator	Correct	Incorrect (Total)	Accuracy [99% CI]
Batch A			
A.	10,700	182 (10,882)	0.983275 [0.980059 0.986400]
L.	10,452	430 (10,882)	0.960485 [0.955615 0.965172]
M.	10,629	253 (10,882)	0.976751 [0.972983 0.980426]
MaxEnt 2	10,537	345 (10,882)	0.968296 [0.963885 0.972523]
Batch B			
A.	9,499	363 (9,862)	0.963192 [0.958223 0.967958]
M.	9,636	226 (9,862)	0.977084 [0.973130 0.980836]
S.	9,332	530 (9,862)	0.946258 [0.940276 0.952038]
MaxEnt 2	9,379	483 (9,862)	0.951024 [0.945346 0.956500]

The first column in Table 7 lists all evaluators (four human evaluators, “A”, “L”, “M”, and “S”, and the MaxEnt 2 classifier). The second column gives the number of correct answers (with respect to the gold standard) produced by each evaluator. The third column shows the number of incorrect answers for each evaluator out of the total number of examples (in parentheses). Examples with tied scores (i.e. two positive and two negative votes) were not considered for the gold standard. The last column shows the accuracy and the 99% confidence interval for the accuracy value. The gold standard was defined as the majority among three human evaluators *and* the MaxEnt 2 algorithm. We did not include evaluation ties (two positive and two negative evaluations for the same statement–sentence pair) into the gold standard, which explains the difference in the number of the statement-sentence pairs used in the 3-evaluator-gold-standard and 4-evaluator-gold-standard experiments. The even (2-by-2) evaluator splits are clearly uninformative in assessing the relative performance of our evaluators because all four evaluators get an equal penalty for each tie case. Batches A and B were evaluated by different sets of human evaluators. We computed the binomial confidence intervals at the α -level of significance ($\alpha \times 100\%$ CI) by identifying a pair of parameter values that separate areas of approximately $\frac{(1-ha)}{2}$ at each distribution tail.

The features that we used in our analysis are obviously not all equally important. To elucidate the relative importance of the individual features and of feature pairs, we computed the mutual information between all pairs of features and the class variable, (see Figure 8). The mutual information of class variable, C, and a pair of feature variables, $\left(F_i, F_j\right)$ is defined in the following way (e.g., see [23,30]).

$$I\left(C;F_i,F_j\right)=$$

$$I\left(F_i,F_j;C\right)=H\left(F_i,F_j\right)+H\left(C\right)-H\left(C,F_i,F_j\right),\ (14)$$

where function $H(P[x])$ is Claude E. Shannon's entropy of distribution $P(x)$ (see p. 14 of [31]), defined in the following way:

$$H\left(P\right)=-\sum_x P(x)\log P\left(x\right),(15)$$

where summation is done over all admissible values of x. Figure 8 shows that the most informative standalone features, as expected, are those that are derived from the human evaluations of the quality of extraction of individual relations and terms (such as the average quality scores), and features reflecting properties of the sentence that was used to extract the corresponding fact. In addition, some dictionary-related features, such as finding a term in the Locus Link, are fairly informative. Some features, however, become informative only in combination with other features. For example, the minimum positive distance between two terms in a sentence is not very informative by itself, but becomes fairly useful in combination with other features, such as the number of commas in the sentence, or the length of the sentence (see Figure 8). Similarly, while finding a term in GenBank does not help the classifier by itself, the feature becomes informative in combination with syntactic properties of the sentence and statistics about the manually evaluated data.

Assignment of facts to classes *correct* and *incorrect* by evaluators is subject to random errors. Facts that were seen by many evaluators would be assigned to the appropriate class with higher probability than facts that were seen by only one evaluator. This introduction of noise affects directly the estimate of the accuracy of an artificial intelligence curator. If the gold standard is noisy, the apparent accuracy of the algorithm compared to the gold standard is lower than the real accuracy. Indeed, the three-evaluator gold standard, see Table 4, indicated that the actual optimum accuracy of the MaxEnt 2 classifier is higher than 88% percent. (The 88% accuracy estimate came from comparison of MaxEnt 2 predictions to the whole set of annotated facts, half of which were seen by only one or two evaluators, see Figure 9) When MaxEnt 2 was compared with the three-human gold standard, the estimated accuracy was about 91%.

5. Discussion

As evidenced by Figures 2 and 3, the results of our study are directly applicable to analysis of large text-mined databases of molecular interactions. We can identify sets of molecular interactions with any pre-defined level of precision (see Figure 9). For example, we can request from a database all interactions with extraction precision 95% or greater, which would result in the case of the GeneWays 6.0 database in recall of 77.9%. However, we are not forced to discard the unrequested lower-than-threshold-precision interactions. Intuitively, even weakly supported facts (i.e. those on which there is not full agreement) can be useful in interpreting experimental results, and may gain additional support when studied in conjunction with other related facts (see Figure 1 for examples of weakly supported yet useful facts, such as *cocaine stimulates prolactin*—with a low extraction confidence, but biologically plausible, the accuracy predictions were computed using the MaxEnt 2 method). We envision that, in the near future, we will have computational approaches, such as probabilistic logic, that will allow us to use weakly supported facts for building a reliable model of molecular interactions from unreliable facts (paraphrasing John von Neumann's "synthesis of reliable organisms from unreliable components" [18]).

Experiments with any stand alone set of data generate results insufficient to allow us to draw conclusions about the general performance of different classifiers. Nevertheless, we can speculate about the reasons for the observed differences in performance of the methods when applied to our data. The modest performance of the Naïve Bayes classifier is unsurprising: We know that many pairs of features used in our analysis are highly or weakly correlated (see Figures 8 and 9). The actual feature dependencies violate the method's major assumption about the conditional independence of features. MaxEnt 1 performed significantly more accurately than the Naïve Bayes in our experiments, but was not as efficient as other methods. It takes into account only the class-specific mean values of features. It does not incorporate parameters to reflect dependencies between individual features. This deficiency of MaxEnt 1 is compensated by MaxEnt 2, which has an additional set of parameters for pairs of features leading to a markedly improved performance.

Our explanation for the superior performance of the MaxEnt 2 algorithm with respect to the remainder of the algorithms in the study batch is that MaxEnt 2 requires the least parameter tweaking in comparison to other methods of similar complexity. Performance of the Clustered Bayes method is highly sensitive to the definition of feature clusters and to the way we discretize the feature values—essentially presenting the problem of selecting an optimal model from an extensive set of rival models, each model defined by a specific set of feature clusters. Our initial intuition was that a reasonable choice of clusters can become clear from analysis of an estimated feature-correlation matrix. We originally expected that more highly correlated parameters would belong to the same cluster. However, the correlation matrices estimated from the complete GeneWays 6.0 database and from a subset of annotated facts turned out to be rather different (see Figure 8) suggesting that we could group features differently. In addition, analysis of mutual information between the class of a statement and pairs of features (see Figure 8) indicated that the most informative pairs of features are often only weakly correlated. It is quite likely that the optimum choice of feature clusters in the Clustered Bayes method would lead to classifier performance accuracy significantly higher than that of MaxEnt 2 in our study, but the road to this improved classifier lies through a search in an astronomically large space of alternative models.

Similar to optimizing the Clustered Bayes algorithm through model selection, we can experiment with various kernel functions in the SVM algorithm, and can try alternative designs of the artificial neural network. These optimization experiments are likely to be computationally expensive, but are almost certain to improve the prediction quality. Furthermore, there are bound to exist additional useful classification features waiting to be discovered in future analyses. Finally, we speculate that we can improve the quality of the classifier by increasing the number of human evaluators who annotate each data point in the training set. This would allow us to improve the gold standard itself, and could lead to development of a computer program that performs the curation job consistently and at least as accurately as an average human evaluator.

Table 8. The receiver operator characteristic (ROC) scores

Method	ROC score $\pm$ 2σ
Clustered Bayes 68	0.8115 ± 0.0679
Naïve Bayes	0.8409 ± 0.0543
MaxEnt 1	0.8647 ± 0.0412
Clustered Bayes 44	0.8751 ± 0.0414
QDA	0.8826 ± 0.0445
SVM-t0	0.9203 ± 0.0317
SVM	0.9222 ± 0.0299
Neural Network	0.9236 ± 0.0314
SVM-t1-d2	0.9277 ± 0.0285
SVM-t2-g2	0.9280 ± 0.0285
SVM-t1-d3	0.9281 ± 0.0280
SVM-t2-g1	0.9286 ± 0.0283
SVM-t2-g0.5	0.9287 ± 0.0285
MaxEnt 2	0.9480 ± 0.0178
MaxEnt 2-v	0.9492 ± 0.0156

Table 8 gives the receiver operator characteristic (ROC) scores (also called *the area under the ROC curve*) for methods used in this study, with error bars calculated in 10-fold cross-validation. The Meta-method is much more expensive computationally than the rest of the methods, so we evaluated it using a smaller data set and the corresponding results are not directly comparable with those for the other methods. The Meta-method outperformed other methods listed in this table when trained on the same data (not shown).

6. Conclusions

Text-mining algorithms make mistakes in extracting facts from the natural-language texts. In biomedical applications, which rely on use of text-mined data, it is critical to assess the quality (the probability that the message is correctly extracted) of individual facts—to resolve data conflicts and inconsistencies. Using a large set of almost manually produced evaluations (most facts were independently reviewed more than once producing independent evaluations), we implemented and tested a collection of algorithms that mimic human evaluation of facts provided by an automated information-extraction system. The performance of our best automated classifiers closely approached that of our human evaluators (ROC score close to 0.95). Our hypothesis is that, were we to use a larger number of human experts to evaluate any given sentence, we could implement an artificial-intelligence curator that would perform the classification job at least as accurately as an average individual human evaluator.

7. References

[1] Sekimizu T, Park HS, Tsujii J (1998) Identifying the interaction between genes and gene products based on frequently seen verbs in MEDLINE abstracts. Genome Inform Ser Workshop Genome Inform 9:62–71.

[2] Ono T, Hishigaki H, Tanigami A, Takagi T (2001) Automated extraction of information on protein-protein interactions from the biological literature. Bioinformatics 17:155–161.

[3] Thomas J, Milward D, Ouzounis C, Pulman S, Carroll M (2000) Automatic extraction of protein interactions from scientific abstracts. Pac Symp Biocomput: 541–552.

[4] Blaschke C, Valencia A (2001) The potential use of SUISEKI as a protein interaction discovery tool. Genome Inform Ser Workshop Genome Inform 12:123–134.

[5] Friedman C, Kra P, Yu H, Krauthammer M, Rzhetsky A (2001) GENIES: a natural language processing system for the extraction of molecular pathways from journal articles. Bioinformatics 17 Suppl. 1:S74–S82.

[6] Rzhetsky A, Iossifov I, Koike T, Krauthammer M, Kra P, et al. (2004) GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data. J Biomed Inform 37:43–53.

[7] Carletta J (1996) Assessing agreement on classification tasks: The kappa statistic. Computational Linguistics 22:249–254.

[8] Ruan W, Pang P, Rao Y (1999) The sh2/sh3 adaptor protein dock interacts with the ste20-like kinase misshapen in controlling growth cone motility. Neuron 24:595–605.

[9] Chan YM, Jan YN (1999) Presenilins, processing of beta-amyloid precursor protein, and notch signaling. Neuron 23:201–204.

[10] Niethammer M, Smith DS, Ayala R, Peng J, Ko J, et al. (2000) Nudel is a novel cdk5 substrate that associates with lis1 and cytoplasmic dynein. Neuron 28:697–711.

[11] Alloway PG, Howard L, Dolph PJ (2000) The formation of stable rhodopsin-arrestin complexes induces apoptosis and photoreceptor cell degeneration. Neuron 28:129–138.

[12] Tanaka H, Shan W, Phillips GR, Arndt K, Bozdagi O, et al. (2000) Molecular modification of n-cadherin in response to synaptic activity. Neuron 25:93–107.

[13] Magga JM, Jarvis SE, Arnot MI, Zamponi GW, Braun JE (2000) Cysteine string protein regulates g protein modulation of n-type calcium channels. Neuron 28:195–204.

[14] Gordon SE, Varnum MD, Zagotta WN (1997) Direct interaction between amino- and carboxyl-terminal domains of cyclic nucleotide-gated channels. Neuron 19:431–441.

[15] Gad H, Ringstad N, Low P, Kjaerulff O, Gustafsson J, et al. (2000) Fission and uncoating of synaptic clathrin-coated vesicles are perturbed by disruption of interactions with the sh3 domain of endophilin. Neuron 27:301–312.

[16] Van Vactor D, Flanagan JG (1999) The middle and the end: slit brings guidance and branching together in axon pathway selection. Neuron 22:649–652.

[17] Turing A (1950) Computing machinery and intelligence. Mind 59:433–560.

[18] von Neumann J (1956) Probabilistic logics and the synthesis of reliable organisms from unreliable components. In: Shannon CE, McCarthy J, editors, Automata Studies, Princeton, NJ: Princeton University Press. pp. 43–98.

[19] Benson DA, Karsch-Mizrachi I, Lipman DJ, Ostell J, Wheeler DL (2005) GenBank. Nucleic Acids Res 33:D34–38.

[20] Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugenic 7:179–188.

[21] Jaynes ET (1957) Information theory and statistical mechanics. Physical Review 106:620–630.

[22] Jaynes ET, Bretthorst GL (2003) Probability theory: the logic of science. Cambridge, UK ; New York, NY: Cambridge University Press.

[23] Cover TM, Thomas JA (2005) Elements of information theory. Hoboken, N.J.: J. Wiley, 2nd edition.

[24] Chauvin Y, Rumelhart DE (1995) Back propagation: theory, architectures, and applications. Developments in connectionist theory. Hillsdale, N.J.: Erlbaum.

[25] Vapnik V (1995) The nature of statistical learning theory. Statistics, Computer Science, Psychology. New York: Springer.

[26] Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines: and other kernel-based learning methods. Cambridge, New York: Cambridge University Press.

[27] Smola AJ (2000) Advances in large margin classifiers. Cambridge, Mass.: MIT Press.

[28] Joachims T (1998) Making large-scale support vector machine learning practical. In: Sch¨olkopf B, Burges C, Smola A, editors, Advances in Kernel Methods: Support Vector Machines, MIT Press, Cambridge, MA.

[29] Ratnaparkhi A (1996) A maximum entropy part-of-speech tagger. In: Empirical Methods in Natural Language Processing. University of Pennsylvania, Philadelphia, pp. 491– 497.

[30] Church KW, Hanks P (1989) Word association norms, mutual information, and lexicography. In: Proceedings of the 27th annual meeting on Association for Computational Linguistics. Morristown, NJ, USA: Association for Computational Linguistics, pp. 76– 83.

[31] Shannon CE, Weaver W (1949) The mathematical theory of communication. Urbana: University of Illinois Press.

[32] Hanley JA, McNeil BJ (1982) The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143:29–36.