Bioinformatics Advance Access published August 12, 2007

Learning string similarity measures for gene/protein name dictionary look-up using logistic regression

Yoshimasa Tsuruoka [1]*, John McNaught [1,2], Jun'ichi Tsujii [1,2,3], and Sophia Ananiadou [1,2].

[1]School of Computer Science, The University of Manchester, UK
[2]National Centre for Text Mining (NaCTeM), Manchester, UK
[3]Department of Computer Science, The University of Tokyo, Japan

Associate Editor: Dr. Jonathan Wren

ABSTRACT

Motivation: One of the bottlenecks of biomedical data integration is variation of terms. Exact string matching often fails to associate a name with its biological concept, i.e. ID or accession number in the database, due to seemingly small differences of names. Soft string matching potentially enables us to find the relevant ID by considering the similarity between the names. However, the accuracy of soft matching highly depends on the similarity measure employed.

Results: We used logistic regression for learning a string similarity measure from a dictionary. Experiments using several large-scale gene/protein name dictionaries showed that the logistic regression-based similarity measure outperforms existing similarity measures in dictionary look-up tasks.

Availability: A dictionary look-up system using the similarity measures described in this paper is available at

http://text0.mib.man.ac.uk/software/mldic/

Contact: yoshimasa.tsuruoka@manchester.ac.uk

1 INTRODUCTION

Looking up a gene/protein dictionary is a common task for both computer systems and researchers in biomedical research. Many of the information extraction systems developed for biomedical documents provide a mapping between gene/protein names found in text and their corresponding identifiers (IDs) in biological databases (Morgan *et al.*, 2004; Hoffmann and Valencia, 2005; Miyao *et al.*, 2006). Databases of genes and proteins usually provide an interface that allows the user to search for the entry of interest using a name.

One of the major obstacles that hinder the effective use of a gene/protein dictionary is the problem of term variation. This is also one of the reasons why text mining systems often fail to find genes or proteins mentioned in the text (Yeganova *et al.*, 2004; Hanisch *et al.*, 2005; Crim *et al.*, 2005; Morgan and Hirschman, 2007).

Types of term variation include orthographic variation (e.g. 'IL2' and 'IL-2'), morphological variation (e.g. 'GHF-1 transcriptional factor and 'GHF-1 transcription factor'), Roman-Arabic (e.g. 'Synapsin 3' and 'Synapsin III'), acronym-definition (e.g. 'IL-2' and 'interleukin-2'), extra words (e.g. 'Zfp580' and 'Zfp580 protein'), different word ordering (e.g. 'Serotonin receptor 1D' and 'Serotonin 1D receptor'), and parenthetical material (e.g. 'Ah receptor' and 'Ah (dioxin) receptor'). Attested term variants often result from a combination of these and can be very complex.

One way to alleviate the problem is to normalize the terms (Fang *et al.*, 2006). For example, converting capital letters to lower case and deleting hyphens and spaces can resolve some of the mismatches caused by orthographic variation.

Another approach, which may be employed in conjunction with the normalization approach, is to use soft string matching methods. Soft matching gives similarity scores between strings, which allows us to associate termforms even when they are not identical. Moreover, soft matching can provide the user with multiple candidates that are ranked according to their similarity scores.

The effectiveness of soft matching almost exclusively depends on the design of the similarity measure that quantifies the degree of similarity between two given strings. One could use a similarity measure designed manually (Krauthammer *et al.*, 2000; Tsuruoka and Tsujii, 2004), but recent studies have shown that automatically tuned measures often give better results. For the task of associating gene/protein names, Yeganova *et al.* (2004) used a hidden Markov model which optimizes its parameters by using synonymous pairs of strings in the dictionary. Cohen and Minkov (2006) used a soft matching technique called SoftTFIDF, which enables us to focus on salient words when comparing the strings.

This paper explores the use of logistic regression to learn a good string similarity. Unlike the aforementioned method based on hidden Markov models, we use not only synonymous pairs of strings but also non-synonymous pairs when optimizing the similarity measure. Moreover, the model allows us to make use of diverse types of information as features that characterize each string pair.

We compare the performance of several similarity measures against the one we derive through logistic regression in two sets of dictionary lookup experiments. In the first set of experiments, we use five species-specific dictionaries, and evaluate the performance of similarity measures using the entries which are held out from each dictionary. In the second set of experiments, we use gene/protein names that actually appear in MEDLINE abstracts for evaluation.

This paper is organized as follows. Section 2 summarizes previous work on string similarity measures. In section 3, we describe the dictionaries and data sets used in the experiments. Section 4 presents a similarity measure based on logistic regression and features used for representing a sample. Section 5 describes experimental results using dictionaries created from BioThesaurus (Liu *et al.*, 2006) and the data provided by the BioCreAtIvE II Gene Normalization task (Morgan and Hirschman, 2007).

*to whom correspondence should be addressed

2 RELATED WORK

Simple similarity measures for soft string matching include character n-gram similarity, the Levenshtein distance (Levenshtein, 1965) and the Jaro-Winkler measure (Winkler, 1999), in which the same penalty value is used regardless of the characters to be matched (or ignored). In general, these simple measures do not work very well for gene/protein names on their own, but they can be useful in situations where the computational cost needs to be minimized.

One can employ a more sophisticated measure by defining different scores for different characters and matching operations. It is straightforward to use different scores in edit distance-like similarity measures. Krauthammer *et al*. (2000) used the well-known BLAST algorithm to identify gene and protein names in text. They converted each character in the strings into an amino-acid sequence so that the BLAST algorithm can find matched strings in a sentence using the similarity measure originally developed for gene sequence alignment. Tsuruoka and Tsujii (2004) used an edit distance measure for protein named entity recognition. They manually tuned the cost function to make it less sensitive to capitalization and hyphenation, and reported an improvement of recall.

There are algorithms that enable us to "learn" string similarity from actual examples of string pairs. Ristad and Yianilos (1998) proposed a generative model for edit distance, and presented an algorithm for tuning the cost of edit operations using synonymous pairs of strings. Smith *et al.* (2003) proposed to use an HMM-based probabilistic model for sequence alignment, and described a training algorithm based on the forward-backward algorithm with which one can estimate the parameters of the HMM using pairs of relevant sequences as the training data. Yeganova *et al.* (2004) applied this model to the identification of related gene/protein names using manually curated training data, and reported their advantages over the aforementioned BLAST-based method. Wellner *et al.* (2005) proposed to train conditional random fields on the optimal operation sequences given by the Levenshtein distance.

The learnable string similarity approaches described above, however, use only synonymous pairs of strings for tuning the parameters. A relatively new line of research is to use non-synonymous pairs as well as synonymous pairs by employing discriminative learning models. Cohen and Richman (2002) proposed to use a maximum entropy-based binary classifier to combine multiple similarity metrics, and applied their method to the task of integrating database entities. Bilenko and Mooney (2003) used a support vector machine for a similar task. Bilenko *et al*. (2005) proposed online learning of similarity functions using a voted-perceptron algorithm. McCallum *et al.* (2005) presented a string edit distance function based on conditional random fields, which allows us to use a variety of features of strings and edit operations.

These discriminative learning-based approaches are attractive because they (a) enable us to explicitly consider the "dissimilarity" of strings, (b) allow us to incorporate a variety of features for characterizing a string pair. For instance, the protein names 'GATA binding protein 2' and 'GATA binding protein 5' are very similar on the character level, but one might want to focus on the difference in the numbers ('2' and '5') when quantifying the similarity between them. This type of information can be easily incorporated as a feature in these discriminative learning models.

ID	Gene/Protein Name
O00203	AP3B1
O00203	AP-3 complex subunit beta-1
O00203	AP3-complex beta-3A-adaptin chain
O00203	Adapter-related protein complex 3 beta-1 subunit
O00203	Adaptor protein complex AP-3 beta-1 subunit
O00203	beta-3A-adaptin subunit of the AP-3 complex
O00203	HPS
O00203	HPS2
O00203	HPS2 GENE
O00203	HERMANSKY-PUDLAK SYNDROME
O00203	HERMANSKY-PUDLAK SYNDROME 2
P53677	AP3M2
P53677	AP-3 complex subunit mu-2
:	:

Table 1. Part of a gene/protein name dictionary.

Dictionary	# of IDs	# of names	# of names per ID
Human	14,893	205,909	13.8
Mouse	11,753	111,702	9.5
E. coli	4,875	37,095	7.6
Yeast	5,914	59,020	10.0
Drosophila	2,376	30,891	13.0
BioCreAtIvE	32,975	182,996	5.5
Overall	72,786	627,613	8.6

Table 2. Statistics of dictionaries used in the experiments.

3 DATA FOR TRAINING AND EVALUATION

This work is largely motivated by the recent development of large-scale gene/protein name dictionaries, including GENA (Koike *et al.*, 2003), ProMiner (Hanisch *et al.*, 2005), and BioThesaurus. These dictionaries are typically constructed by extracting names and descriptions from general biological databases (e.g. HUGO, OMIM, Swiss-Prot, Locuslink) and species-specific databases (e.g. MGI, FlyBase, MGD, SGD).

What makes these dictionaries particularly appealing is that they contain variants of names as well as canonical names. Table 1 shows an example of a gene/protein dictionary. Actual dictionaries typically contain other types of information such as DNA sequences and literature references, but here we focus only on the names and their IDs.

We can learn a variety of information from such dictionary entries. For example, Table 1 tells us that we should match 'AP3B1' and 'AP-3 complex subunit beta-1' because these names share the same ID. At the same time, the dictionary tells us that we should not match 'AP-3 complex subunit beta-1' with 'AP-3 complex subunit mu-2' because they belong to different IDs. In other words, these entries suggest that we treat 'AP3B1' and 'AP-3 complex subunit beta-1' as being *similar*, and 'AP-3 complex subunit beta-1' and 'AP-3 complex subunit mu-2' as being *dissimilar*.

BioThesaurus (Liu *et al.*, 2006) is a collection of more than two million gene/protein names from many different species, and as it includes variants of different types it is a useful resource to enable us to learn string similarity.

For the experiments in this paper, we create species-specific dictionaries from BioThesaurus data[1] for five species (Human, Mouse, E. coli, Yeast, and Drosophila). Each entry in BioThesaurus is associated with a UniProt ID. We consulted the UniProtKB/Swiss-Prot database[2] for selecting a species-specific subset from BioThesaurus. We then removed nonsensical terms, e.g. accession numbers for other databases, using simple regular expressions. Each resulting dictionary is split into two sets. One is used for training, and the other is used for evaluation.

For the second set of experiments, we use a human gene/protein name dictionary, and names that actually appear in MEDLINE abstracts. We extracted these data from the data set originally developed for the gene normalization task at BioCreAtIvE II (2006). The original data set contains the text of MEDLINE abstracts as well, but we do not use them since the recognition of gene/protein names is not our focus in this paper.

Table 2 shows statistics of the dictionaries used in the experiments The dictionaries contain many variants. Each ID has 5 to 14 synonymous names on average.

4 LEARNING STRING SIMILARITY MEASURES USING LOGISTIC REGRESSION

As discussed in the previous sections, a dictionary provides information about what kind of string pairs should be regarded as synonymous (or non-synonymous). In order to build a similarity measure based on the dictionary, we need to generalize this information so that we can make this judgement (synonymous or not) for an arbitrary pair of strings.

One way of achieving this generalization is to use a machine learning approach. In particular, we can use supervised classification models. The task is formalized as a binary classification problem, where the input is a pair of strings and the output is a prediction of whether the strings are synonymous or not. In general, a machine learning-based classifier can output a confidence value for each prediction, so we can use this value as the similarity value for the string pair.

In this work, we use logistic regression for the classification model, which is defined as:

$$p_\lambda(y|x) = \frac{1}{Z(x)} \exp(\sum_i \lambda_i f_i(x,y)),$$

$$Z(x) = \sum_y \exp(\sum_i \lambda_i f_i(x,y)),$$

where x is a pair of strings, y is a binary prediction (synonymous or not), $f_i(x,y)$ is either a binary or a real-valued feature function that characterizes the string pair, and λ_i is the weight for the feature. The weight parameters are determined in such a way that the parameters maximize the conditional log-likelihood of the training data $\sum_{j=1}^{n} \log p(y^{(j)}|x^{(j)})$.

Each item of the training data consists of a string pair and a binary label indicating whether the members of the pair are synonymous or not. Given a dictionary, we could, in principle, create training data in the following way.

1. Generate all possible pairs of names.
2. Label each pair as
   SYNONYMOUS, if the pair share the same ID,
   NON-SYNONYMOUS, otherwise.

However, this method generates $O(n^2)$ samples from a dictionary of n entries, and this amount of training data leads to a prohibitive computational cost for training unless the dictionary is small.

Moreover, there are some cases where we should not treat names as being similar even if they share the same ID. For example, Table 1 contains 'HPS' as a synonym for the ID 'O00203', but, on the surface, the name 'HPS' does not have any similarity with a synonymous name 'AP3B1'. This is because the name 'HPS' has a different naming history to that of 'AP3B1' — 'HPS' stems from a disease name. In such cases, we should not expect a machine-learning algorithm, which solely relies on surface string similarity, to associate the names.

To let the logistic regression model learn from only meaningful samples, we introduce a filtering process. We create a training sample from a string pair, whether it is synonymous or not, only when at least one of the following conditions is satisfied:

- The two strings have a high value (> 0.5) of character bigram similarity, which is computed as follows:

$$(similarity) = \frac{2|g_1 \cap g_2|}{|g_1| + |g_2|},$$

  where g_1 and g_2 are the bigrams in the strings.
- All the characters in the shorter string are included in the longer string in the same order.

In the evaluation stage, the measure simply gives a similarity value of 0 to the pairs which do not pass this filtering.

This filtering process also has the merit of reducing the amount of training samples, but the training cost was still very high. The number of training samples for non-synonymous pairs is much higher than that for synonymous pairs, and we found, in preliminary experiments, that reducing the samples for non-synonymous pairs does not have a large impact on the overall performance. We therefore discarded three quarters of the non-synonymous samples by random sampling[3]. Furthermore, we limited the maximum number of names used in the training to 32,000.

4.1 Features

Logistic regression modelling allows us to incorporate a variety of features. Since the performance of machine learning heavily depends on how to represent the samples, it is important to use features that can well characterise a string pair. In other words, the features should be able to capture the similarity between a variety of variants (e.g. orthographical, morphological, syntactical, modifiers) while highlighting the difference between those terms which are not synonymous.

In our method, we use the following types of features.

- Character Bigrams

[1] "bioThesaurus.dist_2.0.gz" available at ftp.pir.georgetown.edu.

[2] Available at http://www.ebi.uniprot.org/database/download.shtml

[3] Since the dictionaries for E. coli, Yeast, and Drosophila were small, we were able to carry out experiments without this reduction process, but the performance differences were very small. The recall scores at rank 1 were 51.5%, 60.1%, and 63.6% respectively (See Table 4 for comparison).

We use the common character bigrams between the two input strings as features. For example, 'IL2' consists of two bigrams ('IL' and 'L2'), and 'IL2R' consists of three bigrams ('IL', 'L2' and '2R'). We use the shared bigrams ('IL' and 'L2') as binary features. We also use the value of character bigram similarity as a real-valued feature. The similarity between orthographical, morphological and syntactic variants is expected to be captured by these character n-gram based features.

- Prefix/Suffix
  The prefix features focus on the prefixes of the strings. Up to three characters are extracted from the beginning of each string, and the combination of them are used as features. Similarly, the suffix features focus on the suffixes of the strings.
- Sharing the Same Number
  The numbers in the names often convey important information. For example, the string pair 'GATA binding protein 2' and 'GATA binding protein 5' shares many characters, but the difference of '2' and '5' indicates that they belong to different IDs. We use a binary feature which indicates whether the strings contain the same number or not.
- Acronym
  We define a feature which can capture the possibility that one string is an acronym of the other. More specifically, this binary feature indicates whether all the characters in the shorter string are included in the longer string in the same order.
- Common Tokens
  In addition to the character-level features described above, we use token-level features. We first tokenize each string using white spaces and some predefined delimiters ('-', '/', '(', ')', '[', ']', and ','). We then generate the intersection of the two token sets as features. For example, we get 'GATA', 'binding', and '5' as the common tokens from the string pair 'GATA binding protein 5' and 'GATA binding factor 5'.
- Different Tokens
  We also use the symmetrical difference between the two token sets. For the above example, we generate 'protein' and 'factor' as the different tokens. This feature type is expected to capture tokens which are not important in conveying the concept of a term.
- SoftTFIDF
  One of the merits of using machine learning is that we can incorporate information from a different similarity measure. We use the value of SoftTFIDF similarity as a real-valued feature.

5 EXPERIMENTS

We present experiments comparing our similarity measure to other approaches that use soft string matching.

5.1 Existing soft-matching methods

For the purpose of performance comparison between our proposed method and existing soft matching methods, we used the following four existing methods.

5.1.1 Levenshtein distance This is also known as the uniform cost edit distance. The *Levenshtein distance* between two strings s_1 and s_2 is defined as the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion or substitution of a single character. Note that this measure is defined as *distance*, but a distance value is easily convertible to a similarity value in an obvious way.

5.1.2 Jaro-Winkler measure The *Jaro* measure between s_1 and s_2 is defined as:

$$Jaro(s_1, s_2) = \frac{1}{3}\left(\frac{|s_1'|}{|s_1|} + \frac{|s_2'|}{|s_2|} + \frac{|s_1'| - T_{s_1,s_2}}{2|s_1'|}\right),$$

where $|s_1|$ and $|s_2|$ are the lengths of s_1 and s_2 respectively. $|s_1'|$ is the number of "matching" characters in s_1, where a character in s_1 is considered matching if there is the same character in s_2 and they are not farther than $\min(|s_1|, |s_2|)/2$. $|s_2'|$ is defined analogously. T_{s_1,s_2} is the number of character positions at which the character from s_1 and the one from s_2 are different.

Let p_0 be the number of common prefix characters between s_1 and s_2. The Jaro-Winkler measure is

$$Jaro{-}Winkler(s_1, s_2) = Jaro(s_1, s_2) + \frac{p}{10}(1 - Jaro(s_1, s_2)),$$

where $p = \max(p_0, 4)$.

5.1.3 Hidden Markov Model based approach (Smith et al.*, 2003)* The similarity value is defined as the probability of generating matching operations with a hidden Markov model. The cost values of matching operations are optimized using synonymous pairs of strings and a forward-backward algorithm.

We used their source code available at their ftp site[4] for the implementation. The model has several meta-parameters. We used the best setting for the meta-parameters reported in Yeganova *et al.* (2004), that is, the number of states was set to three, and the model was forced to be symmetrical.

To compute the similarity score from the output of the HMM, we used the following equation as in Smith *et al.* (2003).

$$score(s_2; s_1) = \log_{10} Pr(s_1, s_2) - \log_{10} Pr(null, s_2)$$

5.1.4 SoftTFIDF We follow the definition of the SoftTFIDF similarity given in Cohen and Minkov (2006). Each token t_i in string s is given a weight value $w(t_i, s)$, which is computed as $\log(1 + TF) * \log(IDF)$, where TF is the frequency of the word in the dictionary and IDF is the inverse of the fraction of names in the dictionary that contain that word.

The SoftTFIDF similarity is given by

$$SoftTFIDF(s_1, s_2) = \frac{\sum_i \sum_j w(t_i, s_1) w(t_j, s_2) sim(t_i, t_j)}{\sqrt{\sum_i w(t_i, s_1)^2}\sqrt{\sum_j w(t_j, s_2)^2}},$$

where $sim(t_i, t_j)$ is the Jaro-Winkler measure between token t_i and t_j if the measure is equal or greater than 0.9, or 0 otherwise.

5.2 Dictionary lookup evaluation with held-out entries

What we evaluate in our experiments is how accurately the various similarity measures enable us to map gene/protein names with the correct IDs. The first set of experiments uses held-out entries from

[4] ftp://ftp.ncbi.nlm.nih.gov/pub/lsmith

the dictionary as the evaluation data. For each species-specific dictionary that we derived from BioThesaurus, we randomly select 1,000 entries and remove them from the dictionary. These removed entries are kept as the evaluation set. We tune (or learn) string similarity using the remaining entries in the dictionary. Finally, we evaluate the performance using the evaluation set.

The gene/protein name of each entry in the evaluation set is matched against the entries in the corresponding dictionary using the similarity measure learned on the dictionary[5]. We define the score for each ID as the maximum similarity value for the gene/protein names that belong to the ID. The system then ranks the IDs according to their scores.

Table 3 shows an example of a ranked list of IDs for the input term 'Acetylating enzyme for N-terminal of ribosomal protein S5', which is matched against the E. coli dictionary. The correct ID for this protein name is 'P0A948', which has been ranked second by the system for some (here unspecified) similarity measure.

Once we obtain the ranked ID lists according to each similarity measure for all the samples in the evaluation set, we can evaluate the recall score for retrieving correct IDs at each rank, which is given by $\frac{M_i}{N}$, where N is the number of samples in the evaluation set, and M_i is the number of correct IDs included in the top i IDs output by the system.

Figures 1 to 5 compare the performance of the five different methods for each species-specific set. The x-axis gives the ranks, and the y-axis is the recall value achieved at each rank.

As expected, the simple similarity measures based on the Levenshtein distance and the Jaro-Winkler metric were confirmed as not good as the learnable similarity measures. The relative performance varies depending on the species-specific dictionary, but in most cases SoftTFIDF performed slightly better than hidden Markov models. The logistic regression method gave the best results with large margins of more than 10% except for the E. coli data.

We should note that the absolute performance reported in Figures 1 to 5 does not necessarily reflect the difficulty of mapping gene/protein names in text to their IDs. For example, the high number of polysemous terms in a dictionary is a major causes of performance deterioration in these experiments, but it does not necessarily mean that the actual mapping task is difficult because such polysemous terms may not often appear in text.

One may be interested in how general each similarity measure is. Our preliminary experiments indicated that the logistic regression-based similarity measure is highly tuned to the training dictionary. In other words, the similarity measure trained for one species is not very useful for a different species. One possible way of creating a more "general" similarity measure is to use a dictionary containing entries from multiple species in training, which, however, entails an increased cost for training.

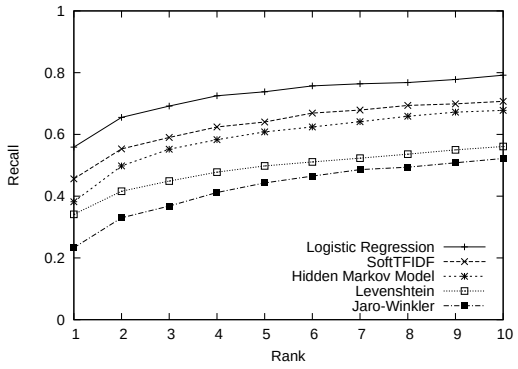

Fig. 1. Dictionary look-up performance (Human).

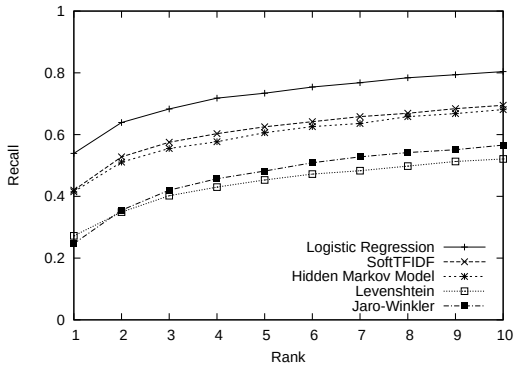

Fig. 2. Dictionary look-up performance (Mouse).

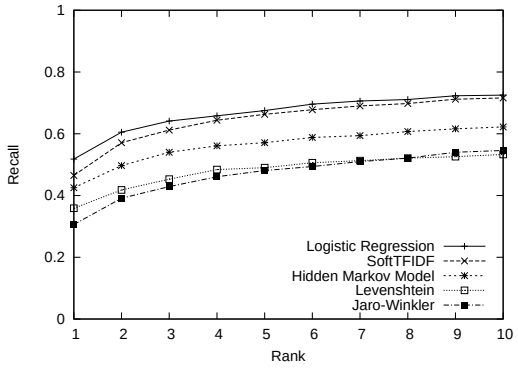

Fig. 3. Dictionary look-up performance (E. coli).

5.3 Dictionary lookup evaluation with gene/protein names appearing in text

The training data set provided by the BioCreAtIvE II (2006) gene normalization task contains human gene/protein name snippets from MEDLINE abstracts and their EntrezGene identifiers. For instance, the data provide a snippet 'Ah (dioxin) receptor' and its EntrezGene ID '196' for the sentence "The Ah (dioxin) receptor binds a number of widely disseminated...". The data also provide a dictionary, which we used for training the similarity measures.

By using these data, we can conduct dictionary lookup experiments similar to the ones presented in the previous section. The difference is that this evaluation uses actual gene/protein names attested in text rather than the gene/protein names held out from the dictionary. This setting could be seen as the situation where we have

[5] For all experiments, we applied very basic normalization (conversion of capital letters to lower case and hyphens to spaces) to the strings prior the use of similarity measures. This normalization has been shown to have little side effect (Cohen *et al.*, 2002).

Rank	ID	Score	Gene/Protein Name	Similarity
1	P0A7W1	0.969	ribosomal protein S5	0.969
			30S ribosomal protein S5	0.896
2	P0A948	0.824	ribosomal-protein-S5-alanine N-acetyltransferase	0.824
			acetylase	0.065
			Ribosomal-protein-alanine acetyltransferase	0.064
3	P0A944	0.125	ribosomal-protein-alanine N-acetyltransferase rimI	0.125
			Ribosomal-protein-alanine acetyltransferase	0.064
:	:	:	:	:

Table 3. Ranked list of IDs for the input 'Acetylating enzyme for N-terminal of ribosomal protein S5' being matched against the E. coli dictionary. The correct ID for this protein name is 'P0A948'.

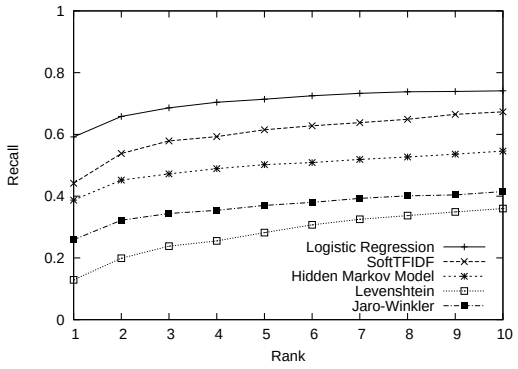

Fig. 4. Dictionary look-up performance (Yeast).

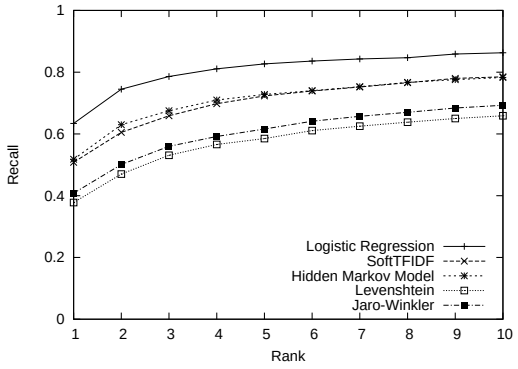

Fig. 5. Dictionary look-up performance (Drosophila).

a tagger that can perfectly identify gene/protein names in text. Thus, the difficulties of name tagging are not considered here.

Note that we do not consider the problem of ambiguity either. In the actual BioCreAtIvE II (2006) gene normalization task, one would need to perform disambiguation for the polysemous names using the context in which the name appears. In this work, we are focusing on the relative performance of soft-matching techniques, so we do not use any information about the context. The performance reported in this section, therefore, should not be compared against the performance of systems in BioCreAtIvE II.

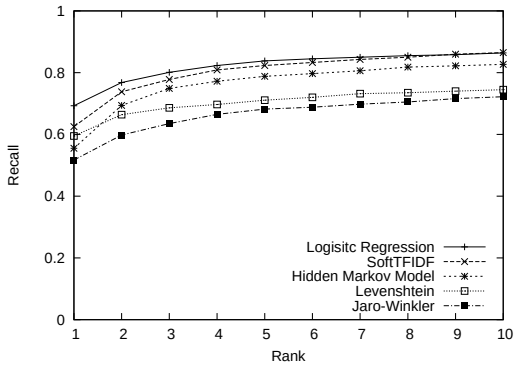

Fig. 6. Dictionary look-up performance (BioCreAtIvE).

Figure 6 shows the results. We can see the same trend between the performance of the methods. The Jaro-Winkler method and the Levenshtein distance are not as effective as the others. SoftTFIDF and the logistic regression method performed relatively well. The logistic regression method outperformed SoftTFIDF. The difference was evident especially where top ranked terms were concerned (69.3% vs 62.5% at rank 1 and 76.8% vs 73.8% at rank 2). The performance differences among the similarity measures were smaller than in the experiments using held-out entries, because many of the gene/protein names in the evaluation set can be associated with the correct IDs by exact string matching.

The logistic regression method failed to associate about 14% of the name snippets with the correct IDs even when top 10 names were considered. One of the major causes of the failures was the ambiguity of snippets like 'alpha 1 subunit' and 'beta c'. Some kind of syntactic grammar should help resolve some of the difficulties caused by complex construction such as coordination (e.g. 'PKC alpha, epsilon, and zeta'). About two thirds of the mismatched names had been ruled out by the filtering process described in section 4. Further refinement of the filtering method should be necessary not to rule out associations such as acronyms with different character ordering (e.g. 'type 2 CRF receptors' and 'CRFR2').

Additional experiments have been carried out using the BioThesaurus dictionary (Human) instead of the dictionary provided, achieving a recall of 91.5% at rank 10. This is an additional indication that the completeness of the dictionary significantly influences the performance.

	Dictionary					
Removed feature type	Human	Mouse	E. coli	Yeast	Drosophila	BioCreAtIvE
None	55.9	53.8	51.7	59.2	63.2	69.2
Character Bigrams	51.8 (-4.1)	50.1 (-3.7)	49.5 (-2.2)	57.3 (-1.9)	61.5 (-1.7)	61.9 (-7.1)
Prefix/Suffix	52.3 (-3.6)	50.8 (-3.0)	51.3 (-0.4)	58.2 (-1.0)	59.4 (-3.8)	66.0 (-3.2)
Sharing the Same Number	54.0 (-1.9)	52.2 (-1.6)	51.8 (+0.1)	58.7 (-0.5)	62.8 (-0.4)	68.6 (-0.6)
Acronym	54.0 (-1.9)	53.2 (-0.6)	51.1 (-0.6)	58.9 (-0.3)	63.0 (-0.3)	68.2 (-1.0)
Common Tokens	54.3 (-1.6)	52.8 (-1.0)	51.3 (-0.4)	58.9 (-0.3)	63.0 (-0.2)	69.6 (+0.4)
Different Tokens	55.2 (-0.7)	55.0 (+1.2)	50.8 (-0.9)	59.2 (-0.0)	62.7 (-0.5)	69.5 (+0.3)
SoftTFIDF	55.2 (-0.7)	53.6 (-0.2)	51.5 (-0.2)	58.7 (-0.5)	63.2 (-0.0)	69.2 (-0.0)

Table 4. Contribution of each feature type. This table shows the performance (recall percentage at rank 1) achieved when one of the feature types is unused.

5.4 Contribution of each feature type

Table 4 shows how each feature type contributed to the dictionary-lookup performance. The first row shows the recall values at rank 1 achieved when all feature types were used. The other rows show the performance achieved when one of the feature types was unused. The amount of contribution from each feature type varies depending on the dictionary, but bigrams, prefixes and suffixes were, overall, most influential.

6 CONCLUSION

We have described a method for learning a string similarity measure using a logistic regression model and a gene/protein name dictionary. We evaluated our method with several dictionary look-up tasks. Experimental results show that our logistic regression-based method outperforms existing soft matching methods.

The simplest application of our similarity measure would be in a user interface for a database where one can search for the ID of a gene/protein of interest by using its name. Our technique is also applicable in information extraction systems to aid mapping from text strings to canonical entries. Further application can be found in areas such as ontology construction and merging; aids to authors (mapping of author usage to preferred usage); and term classification (Spasic *et al.*, 2005) and term clustering which can be used for advanced information retrieval/extraction applications.

ACKNOWLEDGEMENTS

We thank P. Cotter, Y. Sasaki for many valuable comments and discussions, and also the reviewers. Our thanks to the Rebholz Text Mining Group at EMBL-EBI, Hixton, for domain expertise related to bio-resources. This research was supported by EC project BOOTStrep FP6-028099 (www.bootstrep.org). The UK National Centre for Text Mining is sponsored by the JISC/BBSRC/EPSRC.

REFERENCES

Bilenko, M. and Mooney, R. J. (2003). Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)*, pages 39–48.

Bilenko, M., Basu, S., and Sahami, M. (2005). Adaptive product normalization: Using online learning for record linkage in comparison shopping. In *Proceedings of the 5th Internatinoal Conference on Data Mining (ICDM-2005)*, pages 58–65.

Cohen, K. B., Dolbey, A. E., Acquaah-Mensah, G. K., and Hunter, L. (2002). Contrast and variability in gene names. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*, pages 14–20.

Cohen, W. W. and Minkov, E. (2006). A graph-search framework for associating gene identifies with documents. *BMC Bioinformatics*, **7**(440).

Cohen, W. W. and Richman, J. (2002). Learning to match and cluster large high-dimensinoal data sets for data integration. In *Proceedings of KDD*, pages 475–480.

Crim, J., McDonald, R., and Pereira, F. (2005). Automatically annotating documents with normalized gene lists. *BMC Bioinformatics*, **6 (Suppl 1)**(S13).

Fang, H., Murphy, K., Jin, Y., Kim, J. S., and White, P. S. (2006). Human gene name normalization using text matching with automatically extracted synonym dictionaries. In *Proceedings of BioNLP'06*.

Hanisch, D., Fundel, K., Mevissen, H.-T., Zimmer, R., and Fluck, J. (2005). ProMiner: rule-based protein and gene entity recognition. *BMC Bioinformatics*, **6 (Suppl 1)**(S14).

Hoffmann, R. and Valencia, A. (2005). Implementing the iHOP concept for navigation of biomedical literature. *Bioinformatics*, **21**(Suppl. 2), 252–258.

Koike, A., Kobayashi, Y., and Takagi, T. (2003). Kinase pathway database: An integrated protein-kinase and NLP-based protein-interaction resource. *Genome Research*, **13**, 1231–1243.

Krauthammer, M., Rzhetsky, A., Morozov, P., and Friedman, C. (2000). Using BLAST for identifying gene and protein names in journal articles. *Gene*, **259**, 245–252.

Levenshtein, V. I. (1965). Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, **1**(1), 8–17.

Liu, H., Hu, Z.-Z., Zhang, J., and Wu, C. (2006). BioThesaurus: a web-based thesaurus of protein and gene names. *Bioinformatics*, **22**(1), 103–105.

McCallum, A., Bellare, K., and Pereira, F. (2005). A conditional random field for discriminatively-trained finite-state string edit distance. In *Proceedings of Conference on Uncertainty in AI (UAI)*.

Miyao, Y., Ohta, T., Masuda, K., Tsuruoka, Y., Yoshida, K., Ninomiya, T., and Tsujii, J. (2006). Semantic retrieval for the accurate identification of relational concepts in massive textbases. In *Proceedings of Coling/ACL 2006*, pages 1017–1024.

Morgan, A. A. and Hirschman, L. (2007). Overview of BioCreative II gene normalization. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pages 17–22.

Morgan, A. A., Hirschman, L., and Colosimo, M. (2004). Gene name identification and normalization using a model organism database. *Journal of Biomedical Informatics*, **37**, 396–410.

Ristad, E. S. and Yianilos, P. N. (1998). Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(5), 522–532.

Smith, L., Yeganova, L., and Wilbur, W. J. (2003). Hidden Markov models and optimized sequence alighments. *Computational Biology and Chemistry*, **27**, 77–84.

Spasic, I., Ananiadou, S., and Tsujii, J. (2005). MaSTerClass: a case-based reasoning system for the classification of biomedical terms. *Bioinformatics*, **21**, 2748–2758.

Tsuruoka, Y. and Tsujii, J. (2004). Improving the performance of dictionary-based approaches in protein name recognition. *Journal of Biomedical Informatics*, **37**, 461–470.

Wellner, B., Castaño, J., and Pustejovsky, J. (2005). Adaptive string similarity metrics for biomedical reference resolution. In *Proceedings of BioLink 2005*, pages 9–16.

Winkler, W. E. (1999). The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Bureau of the Census.

Yeganova, L., Smith, L., and Wilbur, W. J. (2004). Identification of related gene/protein names based on an hmm of name variations. *Computational Biology and Chemistry*, **28**, 97–107.