Imbalanced Classification Using Dictionary-based Prototypes and Hierarchical Decision Rules for Entity Sense Disambiguation

Tingting Mu
National Centre for Text Mining
University of Manchester
tingting.mu@man.ac.uk

Xinglong Wang
National Centre for Text Mining
University of Manchester
xinglong.wang@man.ac.uk

Jun'ichi Tsujii
Department of Computer Science
University of Tokyo
tsujii@is.s.u-tokyo.ac.jp

Sophia Ananiadou
National Centre for Text Mining
University of Manchester
Sophia.Ananiadou@man.ac.uk

Abstract

Entity sense disambiguation becomes difficult with few or even zero training instances available, which is known as imbalanced learning problem in machine learning. To overcome the problem, we create a new set of reliable training instances from dictionary, called dictionary-based prototypes. A hierarchical classification system with a tree-like structure is designed to learn from both the prototypes and training instances, and three different types of classifiers are employed. In addition, supervised dimensionality reduction is conducted in a similarity-based space. Experimental results show our system outperforms three baseline systems by at least 8.3% as measured by macro F_1 score.

1 Introduction

Ambiguities in terms and named entities are a challenge for automatic information extraction (IE) systems. The problem is particularly acute for IE systems targeting the biomedical domain, where unambigiously identifying terms is of fundamental importance. In biomedical text, a term (or its abbreviation (Okazaki et al., 2010)) may belong to a wide variety of semantic categories (e.g., gene, disease, etc.). For example, *ER* may denote protein *estrogen receptor* in one context, but cell subunit *endoplasmic reticulum* in another, not to mention it can also mean emergency room. In addition, same terms (e.g., protein) may belong to many model organisms, due to the nomenclature of gene and gene products, where genes in model organisms other than human are given, whenever possible, the same names as their human orthologs (Wain et al., 2002). On the other hand, public biological databases keep species-specific records for the same protein or gene, making species disambiguation an inevitable step for assigning unique database identifiers to entity names in text (Hakenberg et al., 2008; Krallinger et al., 2008).

One way to entity disambiguation is classifying an entity into pre-defined semantic categories, based on its context (e.g., (Bunescu and Paşca, 2006)). Existing classifiers, such as maximum entropy model, achieved satisfactory results on the "majority" classes with abundant training instances, but failed on the "minority" ones with few or even zero training instances, i.e., the knowledge acquisition bottleneck (Agirre and Martinez, 2004). Furthermore, it is often infeasible to create enough training data for all existing semantic classes. In addition, too many training instances for certain majority classes lead to increased computational complexity for training, and a biased system ignoring the minority ones. These correspond to two previously addressed difficulties in imbalanced learning: *"... either (i) you have far more data than your algorithms can deal with,*

Coling 2008: Poster Volume, pages 851–859,
Beijing, August 2010

and you have to select a sample, or (ii) you have no data at all and you have to go through an involved process to create them" (Provost, 2000). Given an entity disambiguation task with imbalanced data, this paper explores how to create more informative training instances for minority classes and how to improve the large-scale training for majority classes.

Previous research has shown that words denoting class information in the surrounding context of an entity can be an informative indicator for disambiguation (Krallinger et al., 2008; Wang et al., 2010). Such words are refered to as "cue words" throughout this paper. For example, to disambiguate the type of an entity, that is, whether it is a protein, gene, or RNA, looking at words such as *protein*, *gene* and *RNA* are very helpful (Hatzivassiloglou et al., 2001). Similarly, for the task of species disambiguation (Wang et al., 2010), the occurrence of *mouse p53* strongly suggests that *p53* is a *mouse* protein. In many cases, cue words are readily available in dictionaries. Thus, for the minority classes, instead of creating artificial training instances by commonly used sampling methods (Haibo and Garcia, 2009), we propose to create a new set of real training instances by modelling cue words from a dictionary, called dictionary-based prototypes. To learn from both the original training instances and the dictionary-based prototypes, a hierarchical classification system with a tree-like structure is designed. Furthermore, to cope with the large number of features representing each instance, supervised orthogonal locality preserving projection (SOLPP) is conducted for dimensionality reduction, by simultaneously preserving the intrinsic structures constructed from both the features and labels. A new set of lower-dimensional embeddings with better discriminating power is obtained and used as input to the classifier. To cope with the large number of training instances in some majority classes, we propose a committee machine scheme to accelerate training speed without sacrificing classification accuracy. The proposed method is evaluated on a species disambiguation task, and the empirical results are encouraging, showing at least 8.3% improvement over three different baseline systems.

2 Related Work

Construction of a classification model using supervised learning algorithms is popular for entity disambiguation. A number of researchers have tackled entity disambiguation in general text using wikipedia as a resource to learn classification models (Bunescu and Paşca, 2006). Hatzivassiloglou et al. (2001) studied disambiguating proteins, genes, and RNA in text by training various classifiers using entities with class information provided by adjacent cue words. Wang et al. (2010) proposed a "hybird" system for species disambiguation, which heuristically combines results obtained from classifying the context, and those from modeling relations between cue words and entities. Although satisfactory performance was reported, their system incurs higher computational cost due to syntactic parsing and the binary relation classifier.

Many imbalanced learning techniques, as reviewed by Haibo and Garcia (2009), can also be used to achieve the same purpose. However, to our knowledge, there is little research in applying these machine learning (ML) techniques to entity disambiguation. It is worth mentioning that although these ML techniques can improve the learning performance to some extent, they only consider the information contained in the original training instances. The created instances do not add new information, but instead utilize the original training information in a more sophisticated way. This motivates us to pursue a different method of creating new training instances by using information from a related and easily obtained source (e.g., a dictionary), similar to transfer learning (Pan and Yang, 2009).

3 Task and Corpus

In this work, we develop an entity disambiguation technique with the use of cue words, as well as a general ML algorithm for imbalanced classification using a set of newly created dictionary-based prototypes. These prototypes are represented with different features from those used by the original training instances. The proposed method is evaluated on a species disambiguation task: given a text, in which mentions of biomedical named en-

tities are annotated, we assign a *s*pecies identifier to every entity mention. The types of entities studied in this work are genes and gene products (e.g., proteins), and we use the NCBI Taxonomy[1] (taxon) IDs as species tags and to build the prototypes. Note that this paper focuses on the task of species disambiguation and makes the assumption that the named entities are already recognised.

Consider the following sentence as an example: if one searches the proteins (i.e., the underlined term) in a protein database, he/she will find they belong to many model organisms. However, in this particular context, CD200R-CD4d3+4 is *human* and *mouse* protein, while *r*CD4d3+4 is a *rat* one.[2] We call such a task of assigning species identifiers to entities, according to context, as species disambiguation.

> The amounts of *human* and *mouse* CD200R-CD4d3+4 and *r*CD4d3+4 protein on the microarray spots were similar as visualized by the red fluorescence of OX68 mAb recognising the CD4 tag present in each of the recombinant proteins.

The informative cue words (e.g., *mouse*) used to help species disambiguation are called species words. In this work, species words are defined as any word that indicates a model organism and also appears in the organism dictionaries we use. They may have various parts-of-speech, and may also contain multiple tokens (despite the name species *word*). For example, "human", "mice", "bovine" and "E. Coli" are all species words. We detect these words by automatic dictionary lookup: a word is annotated as a species word if it matches an entry in a list of organism names. Each entry in the list contains a species word and its corresponding taxon ID, and the list is merged from two dictionaries: the NCBI Taxonomy and the UniProt controlled vocabulary of species.[3] The NCBI portion is a flattened NCBI Taxonomy (i.e., without hierarchy) including only the identifiers of *genus* and *species* ranks. In total, the merged list contains 356,387 unique species words and 272,991 unique species IDs. The ambiguity in species words is low: 3.86% of species words map to multiple IDs, and on average each word maps to 1.043 IDs.

The proposed method was evaluated on the corpus developed in (Wang et al., 2010), containing $6,223$ genes and gene products, each of which was manually assigned with either a taxon ID or an "Other" tag, with *human* being the most frequent at 50.30%. With the extracted features and the species ID tagged by domain experts, each occurrence of named entities can be represented as a d-dimensional vector with a label. Species disambiguation can be modelled as a multi-classification task: Given n training instances $\{\boldsymbol{x}_i\}_{i=1}^n$, their $n \times d$ feature matrix $\mathbf{X} = [x_{ij}]$ and n-dimensional label vector $\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^T$ are used to train a classifier $\mathfrak{C}(\cdot)$, where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^T$, $y_i \in \{1, 2, \ldots, c\}$, and c denotes the number of existing species in total. Given m different query instances $\{\hat{\boldsymbol{x}}_i\}_{i=1}^m$, their $m \times d$ feature matrix $\hat{\mathbf{X}} = [\hat{x}_{ij}]$ are used as the input to the trained classifier, so that their labels can be predicted by $\{\mathfrak{C}(\hat{\boldsymbol{x}}_i)\}_{i=1}^m$.

We used relatively simple contextual features because this work was focused on developing a ML framework. In more detail, we used the following features: 1) 200 words surrounding the entity in question; 2) two nouns and two adjectives at the entity's left and right; 3) 5 species words at the entity's left and right. In addition, function words and words that consist of only digits and punctuations are filtered out. The final numerical dataset consists of 6,227 instances, each represented by 16,851 binary features and belonging to one of the 13 classes. The dataset is highly imbalanced: among the 13 classes, the numbers of instances in the four majority classes vary from 449 to 3,220, while no more than 20 instances are contained in the eight minority classes (see Table 1).

[1] http://www.ncbi.nlm.nih.gov/sites/entrez?db= taxonomy

[2] Prefix 'r' in "rCD4d3+4" indicates that it is a *rat* protein.

[3] http://www.expasy.ch/cgi-bin/speclist

4 Proposed Method

4.1 Dictionary-based Prototypes

For each existing species, we create a b-dimensional binary vector, given as $\boldsymbol{p}_i = [p_{i1}, p_{i2}, \ldots, p_{ib}]^T$, using b different species words listed in the dictionary as features, which is called dictionary-based prototype. The binary value p_{ij} denotes whether the jth species word belongs to the ith species in the dictionary. This leads to a $c \times b$ feature matrix $\mathbf{P} = [p_{ij}]$ for c species.

Considering that the species words preceding and appearing in the same sentence as an entity can be informative indicators for the possible species of this entity, we create two more $m \times b$ binary feature matrices for the query instances with the same b species words as features: $\hat{\mathbf{X}}_1 = [\hat{x}_{ij}^{(1)}]$ and $\hat{\mathbf{X}}_2 = [\hat{x}_{ij}^{(2)}]$, where $\hat{x}_{ij}^{(1)}$ denotes whether the jth species word is the preceding word of the ith entity, and $\hat{x}_{ij}^{(2)}$ denotes whether the jth species word appears in the same sentence as the ith entity but is not preceding word. Thus, the similarity between each query entity and existing species can be simply evaluated by calculating the inner-product between the entity instance and the corresponding prototype. This leads to the following $m \times c$ similarity matrix $\hat{\mathbf{S}} = [\hat{s}_{ij}]$:

$$\hat{\mathbf{S}} = \theta \hat{\mathbf{X}}_1 \mathbf{P}^T + (1-\theta)\hat{\mathbf{X}}_2 \mathbf{P}^T, \qquad (1)$$

where $0 \leq \theta \leq 1$ is a user-defined parameter controlling the degree of indicating reliability of the preceding word and the same-sentence word. The $n \times c$ similarity matrix $\mathbf{S} = [s_{ij}]$ between the training instances and the species can be constructed in exactly the same way. Based on empirical experience, the preceding word indicates the entity's species more accurately than the same-sentence word. Thus, θ is preferred to be set as greater than 0.5. The obtained similarity matrix will be used in the nearest neighbour classifier (see Section 4.2.1).

Both the original training instances $\mathbf{X}$ and the newly created prototypes $\mathbf{P}$ are used to train the proposed hierarchical classification system. Subject to the nature of the classifier employed, it is convenient to construct one single feature matrix instead of using $\mathbf{X}$ and $\mathbf{P}$ individually. Aiming at keeping the same similarity values between each entity instance and the species prototype, we construct the following $(n+c) \times (d+b)$ feature matrix for both the training instances and prototypes:

$$\mathbf{F} = \begin{bmatrix} \mathbf{X} & \theta\mathbf{X}_1 + (1-\theta)\mathbf{X}_2 \\ \mathbf{0} & \mathbf{P} \end{bmatrix}, \qquad (2)$$

where $\mathbf{X}_1$ and $\mathbf{X}_2$ are constructed in the same way as $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{X}}_2$ but for training instances. Their corresponding label vector is $\boldsymbol{l} = [\boldsymbol{y}^T, 1, 2, \ldots, c]^T$.

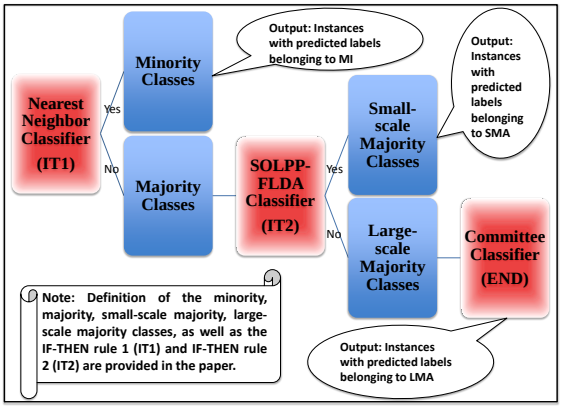

Figure 1: Structure of the proposed hierarchical classification system

4.2 Hierarchical Classification

Multi-stage or hierarchical classification (Giusti et al., 2002; Podolak, 2007; Kurzyński, 1988) is widely used in many complex multi-category classification tasks. Existing research shows such techniques can potentially achieve right trade-off between accuracy and resource allocation (Giusti et al., 2002; Podolak, 2007). Our proposed hierarchical system has a tree-like structure with three different types of classifier at nodes (see Figure 1). Different classes are organized in a hierarchical order to be classified based on the corresponding numbers of available training instances. Letting n_i denote the number of training instances available in the $i$the class excluding the created prototypes, we categorize the classes as follows:

- **Minority Classes (MI):** Classes with less training instances than the threshold: MI $= \{i : \frac{n_i}{n} < \sigma_1, i \in \{1, 2 \ldots, c\}\}$.

- **Majority Classes (MA):** Classes with more training instances than the threshold: MA $= \{i : \frac{n_i}{n} \geq \sigma_1, i \in \{1, 2 \ldots, c\}\}$.
- **Small-scale Majority Classes (SMA):** Majority Classes with less training instances than the threshold: SMA $= \{i : \frac{n_i}{n} < \sigma_2, i \in \text{MA}\}$.
- **Large-scale Majority Classes (LMA):** Majority Classes with more training instances than the threshold: LMA $= \{i : \frac{n_i}{n} \geq \sigma_2, i \in \text{MA}\}$.

Here, $0 < \sigma_1 < 1$ and $0 < \sigma_2 < 1$ are size thresholds set by users. We have MI $\cap$ MA $= \emptyset$, SMA $\cap$ LMA $= \emptyset$, and SMA $\cup$ LMA $=$ MA.

The tree-like hierarchical structure of our system is determined by MI, MA, SMA, and LMA. We propose two IF-THEN rules to control the system: Given a query instance $\hat{\boldsymbol{x}}_i$, the level 1 classifier $\mathfrak{C}_1$ is used to predict whether $\hat{\boldsymbol{x}}_i$ belongs to MA or a specific class in MI, which wer call IF-THEN rule 1 (IT1). If $\hat{\boldsymbol{x}}_i$ belongs to MA, the level 2 classifier $\mathfrak{C}_2$ is used to predict whether $\hat{\boldsymbol{x}}_i$ belongs to LMA or a specific class in SMA, called IF-THEN rule 2 (IT2). If $\hat{\boldsymbol{x}}_i$ belongs to LMA, the level 3 classifier $\mathfrak{C}_3$ finally predicts the specific class in LMA $\hat{\boldsymbol{x}}_i$ belongs to. We explain in the following sections how the classifiers $\mathfrak{C}_1$, $\mathfrak{C}_2$, and $\mathfrak{C}_3$ work in detail.

4.2.1 Nearest Neighbour Classifier

The goal of the nearest neighbour classifier, denoted by $\mathfrak{C}_1$, is to decide whether the nearest-neighbour prototype of the query instance belongs to MI. The only used training instances are our created dictionary-based prototypes $\{\boldsymbol{p}_i\}_{i=1}^{c}$ with the label vector $[1, 2, \ldots, c]^T$. The nearest-neighbour prototype of the query instance $\hat{\boldsymbol{x}}_i$ possesses the maximum similarity to $\hat{\boldsymbol{x}}_i$:

$$NN(\hat{\boldsymbol{x}}_i) = \arg \max_{j=1,2,\ldots,c} \hat{s}_{ij}, \tag{3}$$

where $\hat{s}_{ij}$ is obtained by Eq. (1). Consequently, the output of the classifier $\mathfrak{C}_1$ is given as

$$\mathfrak{C}_1(\hat{\boldsymbol{x}}_i) = \begin{cases} NN(\hat{\boldsymbol{x}}_i), & \text{If } NN(\hat{\boldsymbol{x}}_i) \in \text{MI}, \\ 0, & \text{Otherwise.} \end{cases} \tag{4}$$

The IF-THEN rule 1 can then be expressed as

$$\text{Action}^{(\text{IT1})} = \begin{cases} \text{Go to } \mathfrak{C}_2, & \text{If } \mathfrak{C}_1(\hat{\boldsymbol{x}}_i) = 0, \\ \text{Stop}, & \text{Otherwise.} \end{cases}$$

4.2.2 SOLPP-FLDA Classifier

The goal of the SOLPP-FLDA classifier, denoted by $\mathfrak{C}_2$, is to predict whether the query instance belongs to LMA or a specific class in SMA. In this classifier, the used training instances are the original training entities and the dictionary-based prototypes, both belonging to MA. The feature matrix $\mathbf{F}$ and the label vector $\boldsymbol{l}$ defined in Section 4.1 are used, but with instances from MI removed (we use $\tilde{n}$ to denote the number of remaining training instances, and the same symbol $\mathbf{F}$ for feature matrix). The used label vector $\tilde{\boldsymbol{l}}$ to train $\mathfrak{C}_2$ should be re-defined as $\tilde{l}_i = l_i$ if $l_i \in$ SMA, and 0 otherwise.

First, we propose to implement orthogonal locality preserving projection (OLPP) (Kokiopoulou and Saad, 2007) in a supervised manner, leading to SOLPP, to obtain a smaller set of more powerful features for classification. Also, we conduct SOLPP in a similarity-based feature space computed from $(d + 2b)$ original features by employing dot-product based similarity, given by $\mathbf{F}\mathbf{F}^T$. As explained later, to compute the new features from $\mathbf{F}\mathbf{F}^T$ instead of the original features $\mathbf{F}$ achieves reduced computational cost. An $\tilde{n} \times k$ projection matrix $\mathbf{V} = [v_{ij}]$ is optimized in this n-dimensional similarity-based feature space. The optimal projections are obtained by minimizing the weighted distances between the lower-dimensional embeddings so that "similar" instances are mapped together in the projected feature space. Mathematically, this leads to the following constrained optimization problem:

$$\min_{\substack{\mathbf{V} \in R^{\tilde{n} \times k}, \\ \mathbf{V}^T\mathbf{V} = \mathbf{I}_{k \times k}}} \text{tr}[\mathbf{V}^T\mathbf{F}^T\mathbf{F}(\mathbf{D} - \mathbf{W})\mathbf{F}\mathbf{F}^T\mathbf{V}], \tag{5}$$

where $\mathbf{W} = [w_{ij}]$ denotes the $n \times n$ weight matrix with w_{ij} defining the degree of "closeness" or "similarity" between the ith and jth instances, $\mathbf{D}$ is a diagonal matrix with $\{d_i = \sum_{j=1}^{\tilde{n}} w_{ij}\}_{i=1}^{\tilde{n}}$ as the diagonal elements.

Usually, the weight matrix $\mathbf{W}$ is defined by an adjacency graph constructed from the original

data, e.g. for OLPP. One common way to define the adjacency is by including the K-nearest neighbors (KNN) of a given node to its adjacency list, which is also called the KNN-graph (Kokiopoulou and Saad, 2007). There are two common ways to define the weight matrix: constant value, where $w_{ij} = 1$ if the ith and jth samples are adjacent, while $w_{ij} = 0$ otherwise, and Gaussian kernel. We will denote in the rest of the paper such a weight matrix computed only from the features as $\mathbf{W}_X$. Ideally, if the features can accurately describe all the discriminating characteristics, the samples that are close or similar enough to each other should have the same label vectors. However, when processing real dataset, what may happen is that, in the d-dimensional feature space, the data points that are close to each other may belong to different categories, while on the contrary, the data points that are in a distant to each other may belong to the same category. In the k-dimensional projected feature space obtained by OLPP, one may have the same problem. Because OLPP solves the constrained optimization problem in Eq. (5) using $\mathbf{W}_X$: if two instances are close or similar to each other in the original feature space, they will be the same close or similar to each other in the projected space. To solve this problem, we decide to modify the "closeness" or "similarity" between instances in the projected feature space by considering the label information. The following computation of a supervised weight matrix is used for our SOLPP:

$$\mathbf{W} = (1-\alpha)\mathbf{W}_X + \alpha\mathbf{L}\mathbf{L}^T, \tag{6}$$

where $0 \leq \alpha \leq 1$ is a user-defined parameter controlling the tradeoff between the label-based and feature-based neighborhood structures, and $\mathbf{L} = [l_{ij}]$ is an $\tilde{n} \times c$ binary label matrix with $l_{ij} = 1$ if the ith instance belongs to the jth class, and $l_{ij} = 0$ otherwise.

The optimal solution of Eq. (5) is the top $(k+1)$th eigenvectors of the $\tilde{n} \times \tilde{n}$ symmetric matrix $\mathbf{F}^T\mathbf{F}(\mathbf{D} - \mathbf{W})\mathbf{F}\mathbf{F}^T$, corresponding to the $k+1$ smallest eigenvalues, but with the top one eigenvector removed, denoted by $\mathbf{V}^*$. It is worth to mention that if the original feature matrix $\mathbf{F}$ is used as the input of SOLPP, one needs to compute the eigen-decomposition of the $(d+b) \times (d+b)$ symmetric matrix $\mathbf{F}^T(\mathbf{D} - \mathbf{W})\mathbf{F}$. The corresponding computation complexity increases in $O((d+b)^3)$, which is unacceptable in practical when $d + b \gg \tilde{n}$. The projected features for the training instances are computed by

$$\mathbf{Z} = \mathbf{F}\mathbf{F}^T\mathbf{V}^*. \tag{7}$$

Given a different set of m query instances with an $m \times (d+b)$ feature matrix,

$$\hat{\mathbf{F}} = [\hat{\mathbf{X}},\ \theta\hat{\mathbf{X}}_1 + (1-\theta)\hat{\mathbf{X}}_2], \tag{8}$$

their embeddings can be easily obtained by

$$\hat{\mathbf{Z}} = \hat{\mathbf{F}}\hat{\mathbf{F}}^T\mathbf{V}^*. \tag{9}$$

Then, the projected feature matrix $\mathbf{Z}$ and label vector $\tilde{\boldsymbol{l}}$ are used to train a multi-class classifier. By employing the one-against-all scheme, different binary classifiers $\{\mathfrak{C}_i^{(2)}\}_{i \in \text{SMA} \cup \{0\}}$ with label space $\{+1, -1\}$ are trained. For the ith class ($i \in \text{SMA} \cup \{0\}$), the training instances belonging to it are labeled as positive, otherwise negative. In each binary classifier $\mathfrak{C}_i^{(2)}$, a separating function

$$f_i^{(2)}(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{w}_i^{(2)} + b_i^{(2)} \tag{10}$$

is constructed, of which the optimal values of the weight vector $\boldsymbol{w}_i^{(2)}$ and bias $b_i^{(2)}$ are computed using Fisher's linear discriminant analysis (FLDA) (Fisher, 1936; Mu, 2008). Finally, the output of the classifier $\mathfrak{C}_2$ can be obtained by assigning the most confident class label to the query instance $\hat{\boldsymbol{x}}_i$, with the confidence value indicated by the value of separating function:

$$\mathfrak{C}_2(\hat{\boldsymbol{x}}_i) = \arg \max_{j \in \text{SMA} \cup \{0\}} f_j^{(2)}(\hat{\boldsymbol{x}}_i). \tag{11}$$

The IF-THEN rule 2 can then be expressed as

$$\text{Action}^{(\text{IT2})} = \begin{cases} \text{Go to } \mathfrak{C}_3, & \text{If } \mathfrak{C}_2(\hat{\boldsymbol{x}}_i) = 0, \\ \text{Stop}, & \text{Otherwise.} \end{cases}$$

4.2.3 Committee Classifier

The goal of the committee classifier, denoted by $\mathfrak{C}_3$, is to predict the specific class in LMA the query instance belongs to. The used training

instances are entities and dictionary-based prototypes only belonging to LMA. With the same one-against-all scheme, there are large number of positive and negative training instances to train a binary classifier for a class in LMA. To accelerate the training procedure without sacrificing the accuracy, the following scheme is designed.

Letting n_e denote the number of experts in committee, all the training instances are averagely divided into n_e+1 groups each containing similar numbers of training instances from the same class. The instances in the ith and the $(i+1)$th groups are used to train the ith expert classifier. This achieves overlapped training instances between expert classifiers. The output value of $\mathfrak{C}_i^{(3)}$ is not the class index as used in $\mathfrak{C}_2$, but the value of the separating function of the most confident class, denoted by $f_i^{(3)}$. Different from the commonly used majority voting rule, we only trust the most confident expert. Thus, the output of $\mathfrak{C}_3$ for a query instance $\hat{\boldsymbol{x}}_i$ can be obtained by

$$\mathfrak{C}_3(\hat{\boldsymbol{x}}_i) = \arg \max_{j=1,2,...,n_e} f_j^{(3)}(\hat{\boldsymbol{x}}_i). \qquad (12)$$

By using $\mathfrak{C}_3$, different expert classifiers can be trained in parallel. The total training time is equal to that of the slowest expert classifier. The more expert classifiers are used, the faster the system is, however, the less accurate the system may become due to the decrease of used training instances for each expert, especially the positive instances in the case of imbalanced classification. This is also the reason we do not apply the committee scheme to SMA classes.

5 Experiments

5.1 System Evaluation and Baseline

We evaluate the proposed method using 5-fold cross validation, with around 4,980 instances for training, and 1,245 instances for test in each trial. We compute the F_1 score for each species, and employ macro- and micro- average scheme to compute performance for all species. Three baselines for comparison include:

- **Baseline 1 (B1)** : A maximum entropy model trained with training data only.
- **Baseline 1 (B2)** : Combination of B1 and the species dictionary using rules employed in Wang et al. (2010).
- **Baseline 2 (B3)**: The "hybrid" system combining B1, the dictionary and a relation model [4] using rules (Wang et al., 2010).

Our hierarchical classification system were implemented in two ways:

- **HC**: Only the training data on its own is used to train the system.
- **HC/D**: Both the training data and the dictionary-based prototypes are used to train the system.

5.2 Results and Analysis

The proposed system was implemented with $\theta = 0.8$, $\alpha = 0.8$, $n_e = 4$, and $k = 1000$. The species 9606, 10090, 7227, and 4932 were categorized as LMA, the species 10116 as SMA, and the rest species as MI. To compute the supervised weight matrix, the percentage of the used KNN in the KNN-graph was 0.6. Parameters were not fine tuned, but set based on our empirical experience on previous classification research. As shown in Table 1: HC and B1 were trained with the same instances and features, and HC outperformed B1 in both macro and micro F1. Both HC and B1 obtained zero F_1 scores for most minority species, showing that it is nearly impossible to correctly label the query instances of minority classes, due to lack of training data. By learning from a related resource, HC/D, B2, and B3 yielded better macro performance. In particular, while HC/D and B2 learned from the same dictionary and training data, HC/D outperformed B2 by 19.1% in macro and 2.5% in micro F1. B3 aimed at improving the macro performance by employing computationally expensive syntactic parsers and also by training an extra relation classifier. With the same goal, HC/D integrated the cue word information into the ML classifier in a more general way, and yielded an 8.3% improvement over B3, as measured by macro-F1.

[4] This is an SVM model predicting relations between entities and nearby species words with positive output indicates species words bear the semantic label of entities.

Species Name	Cat.	No.	HC	HC/D	B1	B2	B3
Homo sapiens (9606)	LMA	3220	87.39	**87.48**	86.06	85.43	86.48
Mus musculus (10090)	LMA	1709	79.99	79.98	79.59	80.00	**80.41**
Drosophila melanogaster (7227)	LMA	641	86.62	86.35	**87.96**	87.02	87.37
Saccharomyces cerevisiae (4932)	LMA	499	**90.24**	**90.24**	83.35	81.64	84.64
Rattus norvegicus (10116)	SMA	50	55.07	**69.23**	48.42	64.41	59.41
Escherichia coli K-12 (83333)	MI	18	0.00	0.00	0.00	0.00	0.00
Xenopus tropicalis (8364)	MI	8	0.00	40.00	0.00	**41.67**	36.36
Caenorhabditis elegans (6239)	MI	7	0.00	22.22	0.00	**28.57**	**22.22**
Oryctolagus cuniculus (9986)	MI	3	0.00	0.00	0.00	**20.00**	0.00
Bos taurus (9913)	MI	3	0.00	50.00	0.00	0.00	**100.00**
Arabidopsis thaliana (3702)	MI	2	0.00	0.00	0.00	0.00	**66.67**
Arthropoda (6656)	MI	1	0.00	**100.00**	0.00	50.00	0.00
Martes zibellina (36722)	MI	1	0.00	**50.00**	0.00	28.57	0.00
Micro-average	N/A	N/A	85.03	**85.13**	83.59	83.04	83.80
Macro-average	N/A	N/A	30.72	**51.96**	29.42	43.64	47.97

Table 1: Performance is compared in F1 (%), where "No." denotes the number of training instances and "Cat." denotes the category of species class as defined in Section 4.2.

6 Conclusions and Future Work

Disambiguating bio-entities presents a challenge for traditional supervised learning methods, due to the high number of semantic classes and lack of training instances for some classes. We have proposed a hierarchical framework for imbalanced learning, and evaluated it on the species disambiguation task. Our method automatically builds training instances for the minority or missing classes from a cue word dictionary, under the assumption that cue words in the surrounding context of an entity strongly indicate its semantic category. Compared with previous work (Wang et al., 2010; Hatzivassiloglou et al., 2001), our method provides a more general way to integrate the cue word information into a ML framework without using deep linguistic information.

Although the species disambiguation task is specific to bio-text, the difficulties caused by imbalanced frequency of different senses are common in real application of sense disambiguation. The proposed technique can also be applied to other domains, providing the availability of a cue word dictionary that encodes semantic information regarding the target semantic classes. Building such a dictionary from scratch can be challenging, but may be easier compared to manual annotation. In addition, such dictionaries may already exist in specialised domains.

Acknowledgment

The authors would like to thank the biologists who annotated the species corpus, and National Centre for Text Mining. Funding: Pfizer Ltd.; Joint Information Systems Committee (to UK National Centre for Text Mining)

References

Agirre, E. and D. Martinez. 2004. Unsupervised WSD based on automatically retrieved examples: The importance of bias. In *Proceedings of EMNLP*.

Bunescu, R. and M. Paşca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188.

Giusti, N., F. Masulli, and A. Sperduti. 2002. Theoretical and experimental analysis of a two-stage system for classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):893–904.

Haibo, H. and E. A. Garcia. 2009. Learning from imbalanced data. *IEEE Trans. on Knowledge and Data Engineering*, 21(9):1263–1284.

Hakenberg, J., C. Plake, R. Leaman, M. Schroeder, and G. Gonzalez. 2008. Inter-species normalization of gene mentions with GNAT. *Bioinformatics*, 24(16).

Hatzivassiloglou, V., PA Duboué, and A. Rzhetsky. 2001. Disambiguating proteins, genes, and RNA in text: a machine learning approach. *Bioinformatics*, 17(Suppl 1).

Kokiopoulou, E. and Y. Saad. 2007. Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(12):2143–2156.

Krallinger, M., A. Morgan, L. Smith, F. Leitner, L. Tanabe, J. Wilbur, L. Hirschman, and A. Valencia. 2008. Evaluation of text-mining systems for biology: overview of the second biocreative community challenge. *Genome Biology*, 9(Suppl 2).

Kurzyński, M. W. 1988. On the multistage bayes classifier. *Pattern Recognition*, 21(4):355–365.

Mu, T. 2008. *Design of machine learning algorithms with applications to breast cancer detection*. Ph.D. thesis, University of Liverpool.

Okazaki, N., S. Ananiadou, and J. Tsujii. 2010. Building a high quality sense inventory for improved abbreviation disambiguation. *Bioinformatics*, doi:10.1093/bioinformatics/btq129.

Pan, S. J. and Q. Yang. 2009. A survey on transfer learning. *IEEE Trans. on Knowledge and Data Engineering*.

Podolak, I. T. 2007. Hierarchical rules for a hierarchical classifier. *Lecture Notes in Computer Science*, 4431:749–757.

Provost, F. 2000. Machine learning from imbalanced data sets 101. In *Proc. of Learning from Imbalanced Data Sets: Papers from the Am. Assoc. for Artificial Intelligence Workshop*. (Technical Report WS-00-05).

Wain, H., E. Bruford, R. Lovering, M. Lush, M. Wright, and S. Povey. 2002. Guidelines for human gene nomenclature. *Genomics*, 79(4):464–470.

Wang, X., J. Tsujii, and S. Ananiadou. 2010. Disambiguating the species of biomedical named entities using natural language parsers. *Bioinformatics*, 26(5):661667.