Abstract—The emergence of social tagging and crowdsourcing systems provides a unique platform where multiple weak labelers can form a crowd to fulfill a labeling task. Yet, crowd labelers are often noisy, inaccurate, and have limited labeling knowledge, and worst of all, they act independently without seeking complementary knowledge from each other to improve labeling performance. In this paper, we propose a Self-Taught Active Learning (STAL) paradigm, where imperfect labelers are able to learn complementary knowledge from one another to expand their knowledge sets and benefit the underlying active learner. We employ a probabilistic model to characterize the knowledge of each labeler through which a weak labeler can learn complementary knowledge from a stronger peer. As a result, the self-taught active learning process eventually helps achieve high classification accuracy with minimized labeling costs and labeling errors.

Keywords—active learning; crowd; self-taught;

I. INTRODUCTION

In a traditional active learning setting, an omniscient oracle is required to provide a correct answer to each query [1, 2]. This is, unfortunately, hardly the case for many applications, such as social tagging and crowdsourcing systems, where plenty of users can form abundant weak labeling resources [3]. These emerging Web-based applications have raised a new active learning problem involving multiple nonexpert labelers with imperfect labels for the same set of queried instances [4, 5, 6]. Existing omniscient oracle based active learning cannot take the risk of incorrect information provided by weak labelers into account [5, 7, 8]. Researchers have observed this interesting problem and several works have been reported recently ([7, 9, 10, 11]) for extracting useful labeling information from multiple imperfect labelers. Nevertheless, for all these existing methods, they assume that imperfect labelers’ knowledge sets are fixed and labelers are unable to learn complementary knowledge from one another [6, 11]. This has motivated us to study a new active learning problem, that is, enabling imperfect labelers to learn labeling knowledge from one another to refine their knowledge sets during the active learning process.

In this paper, we propose a Self-Taught Active Learning (STAL) paradigm, where a crowd of imperfect labelers are able to form a self-taught learning system and learn complementary knowledge from one another to expand their knowledge and benefit the underlying active learner. To implement such a self-taught active learning process, we have three challenges to address:

- **Instance Selection:** Identifying the most informative instance for labeling is difficult, mainly because each weak labeler may provide incorrect/noisy labels for the query. We need to identify the most needed instance by taking all labelers as a whole instead of treating them separately;
- **Labeler Selection:** Identifying the most reliable labeler for each selected query is difficult. We need to properly characterize the strength/weakness of each labeler and select the one with the most reliable knowledge for the queried instance;
- **Self-Taught Learning:** While existing methods treat weak labelers as independent individuals, we need to promote self-taught learning between labelers. For specified knowledge or a specific concept, we should know which labeler is good at it and which labeler needs to learn that knowledge.

A conceptual view between existing multi-labeler based active learning methods and our new paradigm is shown in Figure 1. Our framework, STAL, employs a probabilistic knowledge-concept model to explicitly characterize the knowledge of different labelers. We consider that making a query is subject to a certain amount of costs in a multiple-
labeled setting, so each query only involves answers from one selected labeler (instead of asking all labelers to label the queried instance). To properly select the instance-labeler pair in each active learning iteration, we use four random variables \( \mathcal{X}, \mathcal{A}, \mathcal{Y}, \) and \( \mathcal{Z} \) to represent instances, the knowledge of the labelers, the observed labels from the labelers, and the ground truth labels of the instances respectively. So the probability value \( P(\mathcal{Z}|\mathcal{x}) \) can capture the global uncertainty of an unlabeled instance \( \mathcal{x} \) with respect to all labelers, and \( P(\mathcal{A}|\mathcal{x}) \) represents a labeler’s knowledge in labeling instance \( \mathcal{x} \). As a result, we can identify the most informative instance for labeling, and also use the queried instance \( \mathcal{x} \) and its label gained from the most reliable labeler to teach the most unreliable labeler (i.e. self-taught learning).

Experiments from both real-world and benchmark data sets demonstrate a clear performance gain of STAL, compared to a number of baseline algorithms.

II. PROBLEM DEFINITION

We consider active learning in a multiple-labeler scenario where a total of \( M \) labelers/oracles \( \{l_1, \cdots, l_M\} \) exist to provide labeling information for some instances selected from a candidate pool, \( \mathcal{X} = \{ \mathcal{x}_1, \cdots, \mathcal{x}_N \} \), containing \( N \) instances. For any selected instance \( \mathcal{x}_i \) and a labeler \( l_j \), the label provided by \( l_j \) is denoted by \( y_{i,j} \) whereas the ground truth label of \( \mathcal{x}_i \) is denoted by \( z_i \). To clearly characterize a labeler’s labeling capability, we assume that each labeler’s reliability in labeling an instance \( \mathcal{x}_i \) is determined by whether the labeler has the knowledge set, which covers the instance \( \mathcal{x}_i \). We provide more specific definitions as follows.

**Definition 1 Concept:** A concept represents a set of instances sharing the same semantic categorization. For example, \( \text{sports} \) is a concept to represent a number of news documents (i.e. instances) related to sports. Given a data set, a group of concepts, such as \( \{ c_1 = \text{sports}, c_2 = \text{entertainment}, c_3 = \text{politics} \} \), may exist to represent the whole concept space \( \mathcal{C} \) of the data set.

**Definition 2 Knowledge Set:** A knowledge set of a labeler \( l_j \), denoted by \( \mathcal{K}_j \subseteq \mathcal{C} \), represents a set of concepts on which \( l_j \) has the labeling knowledge. For example \( \mathcal{K}_{c_2} = \{ c_1 = \text{sports}, c_2 = \text{entertainment} \} \) indicates that labeler \( l_2 \)’s knowledge set \( \mathcal{K}_2 \) includes two concepts.

**Definition 3 Label Error:** If an instance within a labeler \( l_j \)’s knowledge set was submitted to \( l_j \), the labeler \( l_j \) can provide the ground truth label for the instance, otherwise, \( l_j \) can only guess the label (according to his/her existing knowledge) for the queried instance. The guessed label may be incorrect, which, in turn, introduces a label error.

Given multiple weak labelers and a fixed budget (in terms of the number of queries to the labelers), the aim of self-taught active learning is to query the most informative instances from the candidate pool \( \mathcal{X} \) such that the classifier trained from the labeled instances has the highest classification accuracy.

![Figure 2. The graphical model for modeling instances \( \mathcal{X} \) and their ground truth labels \( \mathcal{Z} \), the reliability of labelers \( \mathcal{A} \), and the actual labels \( \mathcal{Y} \) provided by the labelers. \( \mathcal{X} \) and \( \mathcal{Y} \) can be observed whereas \( \mathcal{A} \) and \( \mathcal{Z} \) are unobservable.](image)

III. MODELING MULTIPLE LABELERS WITH RELIABILITY

Given an instance \( \mathcal{x} \) submitted for labeling and a number of weak labelers, each of which has its own knowledge set, we assume that the ground truth label \( z \) of \( \mathcal{x} \) can be estimated and different labelers can estimate the label \( y \) by using their own knowledge (which is subject to different reliability values with respect to the underlying concepts). More formally, we define a graphical model, as shown in Figure 2, with four random variables \( \mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{Z} \), where \( \mathcal{x}_i \in \mathcal{X} \) represents an instance and \( \mathcal{Y} = \{ y_{i,j}, j \in \mathcal{M} \} \) denotes the labels provided by all labelers. Instance \( \mathcal{x} \) and the label \( y \) provided by each labeler can be observed, whereas variables capturing the labelers \( a \) and the ground truth label \( z \) are unobservable. Then given a set of training data \( \mathcal{X} = \{ \mathcal{x}_1, \cdots, \mathcal{x}_N \} \) and a set of labelers \( \{ l_1, \cdots, l_M \} \), our model should estimate the ground truth labels, defined as \( Z \) and the reliability of labelers, defined as \( A \), for each instance \( \mathcal{x} \). This graphical model can be represented by the joint distribution defined in Eq.(1).

\[
p(\mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{Z}) = \prod_{i}^{N} p(z_i | \mathcal{x}_i) \prod_{i}^{N} \prod_{j}^{M} p(a_{i,j} | \mathcal{x}_i) p(y_{i,j} | z_i, a_{i,j})
\]

From the above model, we can estimate the ground truth label \( z \) for each instance \( \mathcal{x} \) (i.e. \( P(z | \mathcal{x}) \)), and we can also estimate the reliability of each labeler with respect to each instance (i.e. \( P(a | \mathcal{x}) \)).

A. Inference

In a multiple-weak-labeler setting, labelers may have different reliability for each instance, depending on each labeler’s knowledge. In our model, we explicitly use \( a \) to denote the uncertainty of a labeler in labeling a queried instance \( \mathcal{x}_i \). (In the following sections, we use uncertainty and reliability interchangeably to characterize each labeler with uncertainty inversely proportional to the reliability of each labeler.) The lower the \( a \) value, the more confident the labeler will be in labeling the instance. Because the actual label \( y \) provided by each labeler is an offset of the instance’s genuine label \( z \), subject to the labeler’s uncertainty

859
a, we define the following model to capture the relationship between each instance's genuine label and its actual label provided by a labeler.

\[ p(y_{i,j}|a_{i,j}, z_i) = N(z_i, a_{i,j}) \]  

(2)

where label \( y_{i,j} \) provided by labeler \( l_j \) is subject to a Gaussian distribution whose mean is the ground truth label of instance \( x_i \) and the variance is the labeler’s uncertainty in labeling \( x_i \).

The model in Figure 2 indicates that the ground truth label of instance \( x_i \) solely depends on the instance itself. We use a logistic regression model to capture the relationship between \( x \) and \( z \) as follows:

\[ p(z_i|x_i) = (1 + \exp(-w^T x_i - \lambda))^{-1} \]  

(3)

As one of the important aspects of the graphical model in Figure 2, we need to clearly model the knowledge of different labelers and the uncertainty of each labeler with respect to different concepts and different query instances. Because knowledge sets of different labelers might be different (or overlapping), we use a weighted concept representation to represent the knowledge set of a labeler \( l_j \) as follows:

\[ K_j = \{\alpha^t_{c_1}, \ldots, \alpha^t_{c_T}\} \]  

(4)

where \( \alpha^t_{c_i} \) indicates the confidence of labeler \( l_j \) for labeling concept \( c_i \). \( \alpha^t_{c_i} = 0 \) indicates the labeler does not have knowledge to label concept \( c_i \). Because an instance \( x \) may belong to one or multiple concepts, we use \( p(c_i|x) \) to represent \( x \)'s membership of belonging to concept \( c_i \). Accordingly, given a total of \( T \) concepts in the data set, an instance’s membership with respect to each concept set is given as follows:

\[ M_x = \{p(c_1|x), p(c_2|x), \ldots, p(c_T|x)\} \]  

(5)

Then for labeler \( l_j \), its uncertainty in labeling instance \( x_i \) is given in Eq.(6).

\[ p(a_{i,j}|x) = (1 + \exp(\sum_{t=1}^{T} \alpha^t_{c_i} p(c_i|x) + q))^{-1} \]  

(6)

After deriving the knowledge set model, we consider the set of concepts \( \mathcal{C} \). In our pool-based setting, we can assume that each instance belongs to one or multiple concepts, where each concept can be represented by a Gaussian distribution. As a result, the whole data set \( \mathcal{X} \) can be represented by using a mixture Gaussian model with \( T \) concepts

\[ p(x) = \sum_{t=1}^{T} w_t g(x|\mu_t, \Sigma_t) \]  

(7)

where \( w_t (t = 1, \ldots, T) \) is a mixture weight, and each \( g(x|\mu_t, \Sigma_t) \) is a component Gaussian density. Each component is a D-variable Gaussian function of the form

\[ g(x|\mu_t, \Sigma_t) = \frac{1}{(2\pi)^{D/2} |\Sigma_t|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_t)^T \Sigma_t^{-1} (x - \mu_t) \right\} \]  

(8)

with a mean vector \( \mu_t \) and a covariance matrix \( \Sigma_t \). The mixture weights satisfy the constant that \( \sum_{t=1}^{T} w_t = 1 \). Given all concepts in a data set \( \mathcal{X} \), the membership of an instance \( x \), with respect to concept \( c_t \), is given as follows:

\[ p(c_t|x) = N(x|\mu_t, \Sigma_t) \]  

(9)

### B. Maximum Likelihood Estimation

Given observed variables, i.e., instances and their class labels provided by labelers, we would like to infer hidden values. Our training process is to learn two groups of model parameters \( \Theta = \{T, \Psi\} \), where \( T = \{w, \lambda\} \) and \( \Psi = \{K_j, q_j\}_{j=1}^{M} \). This can be solved by using a traditional EM process as follows:

**E-step:** Compute the expectation of the log data likelihood with respect to the distribution of the latent variables derived from the current estimation of the model parameters.

Assume that we have a current estimate of the labeler parameters. We compute the posterior on the estimated ground truth:

\[ \hat{p}(z_i) = p(z_i|x_i, A, Y) \propto p(z_i, A, Y|x_i) \]  

(10)

where

\[ p(z_i, A, Y|x_i) = \prod_j p(a_{i,j}|x_i)p(y_{i,j}|z_i, a_{i,j})p(z_i|x_i) \]  

(11)

**M-step:** To estimate the model parameters, we maximise the expectation of the logarithm of the posterior on \( z \) with respect to \( \hat{p}(z_i) \) from the E-step:

\[ \Theta^* = \arg\max_{\Theta} Q(\Theta, \hat{\Theta}) \]  

(12)

where \( \hat{\Theta} \) is the estimate from the previous iteration and

\[ Q(\Theta, \hat{\Theta}) = E_z[\log(p(x_i, A, Y|z_i))] \]

\[ = \sum_{i,j} \mathbb{E}_{z_i}[\log p(a_{i,j}|x_i) + \log p(y_{i,j}|z_i, a_{i,j}) + \log p(z_i|x_i)] \]

We can compute the updated parameters by using the LBFGS quasi-Newton method [12] to solve the above optimization problem, which does not require second derivatives.

### IV. SELF-TAUGHT ACTIVE LEARNING

#### A. Instance Selection

The goal of active learning is to learn the most accurate classifier with the least number of labeled instances. We employ commonly used uncertainty sampling, by using posterior probability \( p(z|x) \) trained from our graphical model, to select the most informative instance as follows:

\[ x^* = \arg\max_{x_i \in \mathcal{X}} H(z_i|x_i) \]  

(13)

where

\[ H(z_i|x_i) = -\sum_{z_i} p(z_i|x_i) \log (z_i|x_i) \]  

(14)
In Algorithm 1, Step 5 represents the most informative instance selection process.

B. Labeler Selection

Given an instance selected from Eq.(13), labeler selection intends to identify the most reliable labeler who can provide the most accurate label for the queried instance. Because mislabeled instances will severely reduce the accuracy of the classifier trained from the labeled set [13], the reliability of each labeler, with respect to each instance, can be computed using Eq.(6), where $p(a_{i,j}|x_i)$ represents the uncertainty of labeler $l_j$ with respect to the queried instance $x_i$. Accordingly, we can simply rank the conditional probability values from Eq.(6) in an ascending order and select the labeler with the lowest uncertainty score to label the queried instance, as given in Eq.(15).

$$j^* = \arg\min_{j \in M} p(a_{i,j}|x_i) \quad (15)$$

It is worth noting that the uncertainty calculated in Eq.(6) involves two important components: (1) the labeler’s knowledge set, and (2) the memberships of each instance belonging to different concepts. As a result, the uncertainty of a labeler with respect to an input instance $x_i$ is determined by the labeler’s knowledge with respect to each concept in the data set, as defined in Eq.(4), and by the membership of each instance belonging to each concept as defined in Eq.(6).

In Algorithm 1, Steps 6-7 represents the labeler selection process.

C. Self-Taught Learning between Labelers

The above instance and label pair selection process provides solutions to identify the most informative instance and selects the most reliable labeler to label the instance. A self-taught learning process intends to use the knowledge gained from the most reliable labeler to teach the most unreliable labeler so that a weaker labeler can gain knowledge from its stronger peer.

In our multiple-weak-labeler setting, we use $X^{l_j}$ to denote instances which can be accurately labeled by labeler $l_j$. The instances in $X^{l_j}$ essentially form the knowledge set which determines the labeling capability of $l_j$. If we can expand $X^{l_j}$ by using high quality instances labeled by other labelers, it will eventually enhance the knowledge of $l_j$ and improve its labeling capability. Accordingly, we can include instances labeled from the most reliable labeler to improve a weak labeler’s knowledge as given in Eq.(16).

$$X^{l_j}_w \leftarrow X^{l_j}_w \cup (x^*, y^{x*,j*}) \quad \text{where } j^w = \arg\max_{j \in M} p(a_{i,j}|x_i) \quad (16)$$

Please note that the self-taught active learning process between labelers in Eq.(16) only uses knowledge gained from the most reliable labeler (according to Eq.(15)) to teach the most unreliable labeler. This is because even the most reliable labeler can be incorrectly identified, so the label provided by the most reliable labeler might be incorrect. While it is possible to propagate the knowledge to all weak labelers, the pairwise self-taught learning between the strongest and the weakest labelers ensures that error knowledge does not flood all labelers that would eventually deteriorate active learning.

In Algorithm 1, Steps 8-9 represent the self-taught learning process.

Algorithm 1 Self-Taught Active Learning from Crowds

**Input:** (1) Candidate pool $X$; (2) Multiple weak labelers $l_1, \cdots, l_M$; and (3) The number (or the percentage) of queries allowed by the labelers ($reqQueries$)

**Output:** Labeled instance set $L$

1. Initialize model by randomly labeling a small portion of instances from $X$ and compute the initial parameters $\Theta$; 
2. numQueries $\leftarrow 0$; 
3. $X^{l_j} \leftarrow$ initial knowledge of each labeler $l_j, j \in M$; 
4. while numQueries $\leq$ reqQueries do 
5. $x^* \leftarrow$ most informative instance from candidate pool $X$ (Eq.(13)); 
6. $j^* \leftarrow$ most reliable labeler for instance $x^*$ (Eq.(15)); 
7. $(x^*, y^{x^*,j^*}) \leftarrow$ request instance $x^*$’s label from labeler $l_{j^*}$; 
8. $j^w \leftarrow$ most unreliable labeler (Eq.(16)); 
9. $X^{l_j}_w \leftarrow X^{l_j}_w \cup (x^*, y^{x^*,j^*})$ (self-taught learning); 
10. $L \leftarrow L \cup (x^*, y^{x^*,j^*})$; 
11. $\Theta \leftarrow$ retrain model using the updated labeled data and its label (Sec. IV.C); 
12. numQueries $\leftarrow$ numQueries + 1; 
13. end while

V. Experiments

We evaluate the performance of the proposed STAL algorithm based on two data sets and implement the following baselines for experimental comparisons:

- **Multi-Labeler Active Learning:** it uses our active learning model to select the most informative instance and the most reliable labeler for the labeling process. There is, however, no self-taught learning mechanism between labelers.

- **Random Sampling Self-Taught:** it does not use active learning but randomly chooses an instance for querying and uses the most reliable labeler to label this instance. After querying the class label, it will let a weak labeler learn from the most reliable labeler.

- **Multi-Labeler Random Sampling:** it does not use active learning but randomly chooses an instance for querying and uses the most reliable labeler based on our multiple-weak-labeler probabilistic graph model to label this instance. There is, however, no self-taught learning between labelers.

In our experiments, we use 10-fold cross-validation and report the average results. In each fold, we randomly label
a small subset of instances to initialize the active learning process and use logistic regression for classification.

A. A Real-World Data Set

Our real-world test-bed includes a publicly available corpus of 1000 sentences from scientific texts annotated by multiple annotators [14]. We use its polarity labels in our experiments. We set the fragments as the instances and their polarities are treated as labels. We collect the fragments segmented from sentences on which all five experts break in the same way. Meanwhile, we also remove fragments with less than 10 characters. Similar to the tf-idf, we use the term frequency and its inverse document frequency for those fragments to extract the most common words. As the results of the above preprocessing process, we construct 504 instances each containing 153 features. Because we do not know the actual concepts of the data set, we use k-means clustering to generate a number of clusters as concepts (7 clusters in our experiments). Then we can calculate each instance’s membership with respect to each individual concept/cluster.

To demonstrate that the proposed STAL method is effective to help each labeler improve its labeling knowledge, we report the knowledge propagation map for different labelers (with respect to the concepts in the data set) in Figures 3(b) to 3(d). In each of the propagation maps, the x-axis denotes the concepts and the y-axis represents the labelers. The intensity in each cell, $I_{u,v}, (u = 1, \ldots, 7; v = 1, \ldots, 5)$ represents the average uncertainty of a labeler with respect to all instances belonging to that specific concept, defined as

$$I_{u,v} = 255 \times \frac{\sum_{i=1}^{\mid c_u \mid} p(a_{i,v} | x_i)}{\mid c_u \mid}$$

(17)

where $\mid c_u \mid$ represents the number of instances in concept $c_u$ and 255 is used to normalize the intensity into a [0,255] range. The lower the intensity value, the less is the uncertainty of the labeler on the instances.

B. A Benchmark Data Set

We also validate the performance of our algorithm on a publicly available UCI benchmark data set: Vertebral Column [15]. Because this data set was not labeled by multiple labelers, we generate several synthetic labelers each having its own knowledge set, to simulate multi-labeler scenarios. For each scenario, use k-means clustering to generate 7 clusters and compute each instance’s membership with respect to each cluster. We assume that each simulated labeler has knowledge to accurately label instances in one or two clusters, so different labelers have different labeling knowledge. Meanwhile we also simulate that labelers have overlapped knowledge between each other as follows.

For the 7 clusters generated from the data set, we randomly select two clusters and remove all instances in the two clusters, so we have five clusters and five labelers in total. For each of the five cluster $c_u$, we assign one labeler $l_v$ to the cluster and assume $l_v$ is fully capable of labeling all instances in $c_u$. Meanwhile, for each labeler $l_v$, we also randomly choose 35% instances from other four clusters and let $l_v$ have those instances’ labeling information. By doing so, we are allowing $l_v$ to have partial knowledge of labeling concepts outside of $l_v$’s cluster. We repeat the same process for all labelers to make sure that each labeler has partial knowledge to label instances outside of its own knowledge set. In addition, for the two clusters removed at the beginning, we choose one labeler and let the labeler have labeling information for one of the two clusters (so the selected labeler has knowledge to label two concepts). For the cluster which is not selected, its labeling information is evenly divided by 5 labelers so each labeler knows 20% instances in the cluster. By using the above process, we can simulate 5 labelers with different yet overlapping labeling knowledge. This simulation of multiple labelers with complementary knowledge can closely simulate real-world applications with weak labelers and was also used in a previous study [6].

In Figures 3(a) and 3(e) we report the learning curves of the classifiers trained from the instance sets labeled by different active learning methods, which demonstrate that STAL results in better performance than all the other methods. Multi-Labeler Random Sampling has the worst performance and is followed by Multi-Labeler Active Learning and Random Sampling Self-Taught. Clearly, random sampling does not choose informative labeled data to train and this results in the worst performance. On the other hand, while Multi-Labeler Active Learning does choose informative instances to label, the inherent limitation of the weak labelers does not allow active learners to improve themselves further. By properly modeling the knowledge of multiple labelers and enabling knowledge propagation (self-taught learning) between labelers, STAL achieves the best performance for the benchmark data set.

In Figures 3(b) to 3(d) and 3(f) to 3(h), we also report each labeler’s knowledge propagation for the two data sets. Overall, the results clearly show that the knowledge of the labelers can be significantly improved during the active learning process. The most interesting results are from the labeling knowledge propagation for concept 7. At the beginning, each labeler only has very little knowledge to label instances in this concept. However, as the query process and the self-taught active learning between labelers continue, all labelers gain a significant amount of knowledge to label instances in this concept, as shown in the last column of each map.

VI. Conclusion

In this paper, we formulated a new active learning problem, called Self-Taught Active Learning (STAL), where
Hence, can learn complementary knowledge from one another to refine their knowledge to benefit active learning. The proposed STAL framework consists of two alternating steps. First, it combines instance uncertainty and labelers’ knowledge to select an instance-labeler pair for labeling. Second, it encourages labelers to learn from each other’s labeling information. Experimental results demonstrated that the proposed STAL method can accurately capture labelers’ strength/weakness and select the most reliable labeler for each query. The results also showed that the quality of the labeled instance set is better than those of the other baseline methods and validated that self-taught active learning does help improve the labeling capability of each labeler over time.

ACKNOWLEDGMENTS

This work is supported by the Australian Research Council (ARC) Future Fellowship under grant FT100100971, the US NASA Research Award (NNX09AK86G), the US National Science Foundation under grant CCF-0905337, and the National 863 Program of China under grant 2012AA011005.

REFERENCES