Multi-Label Learning with Weak Label

Yu-Yin Sun  Yin Zhang  Zhi-Hua Zhou∗
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210093, China
{sunyy, zhangyin, zhouzh}@lambda.nju.edu.cn

Abstract

Multi-label learning deals with data associated with multiple labels simultaneously. Previous work on multi-label learning assumes that for each instance, the “full” label set associated with each training instance is given by users. In many applications, however, to get the full label set for each instance is difficult and only a “partial” set of labels is available. In such cases, the appearance of a label means that the instance is associated with this label, while the absence of a label does not imply that this label is not proper for the instance. We call this kind of problem “weak label” problem. In this paper, we propose the WELL (WEak Label Learning) method to solve the weak label problem. We consider that the classification boundary for each label should go across low density regions, and that each label generally has much smaller number of positive examples than negative examples. The objective is formulated as a convex optimization problem which can be solved efficiently. Moreover, we exploit the correlation between labels by assuming that there is a group of low-rank base similarities, and the appropriate similarities between instances for different labels can be derived from these base similarities. Experiments validate the performance of WELL.

Introduction

In previous multi-label studies, a basic assumption is that all the proper labels of every training instance are given. For example, if a training image contains the concepts tiger, trees and forest, the user should provide the labels tiger, trees and forest for the image. In many applications, however, this assumption hardly holds since getting all the proper labels is difficult, and generally only a “partial” label set is available. For example, the user may only tag the image with the label tiger while missing the labels trees and forest. In such scenario, if the user provides a label for the instance, we know that this is a proper label for this instance; while for a label which has not been assigned to the instance, we could not conclude that this is not a proper label for the instance. It is evident that this scenario is quite different from the classic multi-label setting where all proper labels for training instances are assumed to be given. We call this kind of multi-label problem the “weak label” problem.

The weak label problem is related to but different from the PU-learning (Positive and Unlabeled data Learning) problem (Li and Liu 2003; Liu et al. 2003; Fung et al. 2006; Elkan and Noto 2008). If all labels are independent, a weak label problem can be decomposed into a series of PU-learning problems, each corresponding to a label. Such simple decomposition, however, ignores the correlation between the labels that can be very useful. For example, the appearance of the label computer in an image strongly implies the existence of the label desk and the nonexistence of the label tiger. In weak label problem, since the label information of training examples is incomplete, we may want to exploit the label correlation rather than simply treating the labels independently. Moreover, PU-learning methods generally do not consider class imbalance, while class imbalance inherently exists in the weak label problem because there are multiple labels and for each label, the number of positive examples is usually much smaller than the number of negative examples.

In this paper, we study the weak label problem and propose the WELL (WEak Label Learning) method. We require the classification boundary for each label to go across low density regions, and explicitly consider the inherent class imbalance in the weak label problem. We formulate the objective as a convex optimization problem which can be solved by quadratic programming efficiently. To exploit the label correlation, we assume that there is a group of low-rank similarities. Experiments validate the performance of WELL.

∗This research was supported by the National Fundamental Research Program of China (2010CB327903), the National Science Foundation of China (60975043, 60721002), the Jiangsu 333 High-Level Talent Cultivation Program and the MSRA IST Program.

Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
positive examples are sampled completely randomly from unlabeled data. Based on the assumption that the labeled examples for each label is usually much smaller than that of negative examples. There were some efforts for relaxing the problem caused by class imbalance. For example, Zhang and Zhou (2007) considered label prior probabilities gained from the k-nearest neighbors of the instance and utilized maximum a posteriori (MAP) principle to determine proper labels in their ML-kNN method. Another difficulty is on the exploitation of the correlation among class labels. Many multi-label learning methods have tried to consider the label correlation in different ways. Examples include methods based on probabilistic generative models (McCallum 1999; Ueda and Saito 2003), maximum entropy methods (Ghramaw and McCallum 2005; Zhu et al. 2005), hypergraph spectral learning (Sun, Ji, and Ye 2008), shared subspace classification (Ji et al. 2008), models-shared subspace boosting (Yan, Tesić, and Smith 2007), maximizing the dependence between features and labels (Zhang and Zhou 2010), etc. Some multi-label learning methods work by transforming the task into a ranking problem, trying to rank the proper labels before other labels for each instance. Representative methods include BoosTexter (Schapire and Singer 2000), RankSVM (Elisseeff and Weston 2002), etc.

PU-learning, also known as partially supervised learning (Li and Liu 2003), studies the problem where a small positive example set (P) and a large unlabeled example set (U) are given for training. This is a special kind of semi-supervised learning where there is no labeled negative examples. Many existing PU-learning methods (Li and Liu 2003; Liu et al. 2003; Fung et al. 2006) first try to obtain a set of labeled negative examples, by considering the instances which are with the highest confidence to be negative, and then train a supervised or semi-supervised classifier. Some methods (Lee and Liu 2003; Liu et al. 2003) treat all the unlabeled instances as negative and assign different costs or weights to different kinds of errors, where the costs associated with labeled data are larger than those associated with unlabeled data. Based on the assumption that the labeled positive examples are sampled completely randomly from all the potential positive examples, Elkan and Noto (2008) showed that a PU-learner predicts probabilities that differ by only a constant factor from the true conditional probabilities of being positive, and this factor can be estimated on validation sets.

The WELL Method

Let X denotes the feature space and suppose there is a label set Θ containing m different labels. The proper labels associated with an instance x ∈ X compose a subset of Θ, which can be represented as an m-dimensional binary label vector, with 1 indicating that the instance belongs to the concept corresponding to the dimension and 0 otherwise. All the labels consist of the label space Y = {0, 1}^m. In the classic multi-label learning setting, for n labeled instances we have a full label matrix Y ∈ {0, 1}^{n × m} where Y_{ik} = 1 means the k-th label is a proper label while Y_{ik} = 0 means the k-th label is not a proper label for the i-th instance. In the weak label problem, Y is unknown and instead we are given a partial label matrix Ŷ ∈ {0, 1}^{n × m} where Ŷ_{ik} ≤ Y_{ik}. Different from the full label matrix, Ŷ_{ik} = 0 tells us nothing. We want to learn a predicted label matrix F ∈ {0, 1}^{n × m} from Ŷ to approximate Y.

Problem Formulation

We observe that in most multi-label applications, for each label the number of positive examples is much smaller than that of negative examples. Therefore, we want the predicted positive instances to be as sparse as possible for each label, i.e., we want to minimize 1^T F_k, where 1 is all-one column vector and F_k is the k-th column of F. Similar to graph-based unsupervised/semi-supervised learning methods (Shi and Malik 2000; Belkin, Niyogi, and Sindhwani 2006), we construct a PSD (positive semi-definite) similarity matrix W = W_{ij} ∈ R^{n × n}, where W_{ij} is the similarity between the i-th and the j-th instances. Minimizing \sum_{i,j} (F_{ik} - F_{jk})^2 W_{ij} is equivalent to requiring the classification boundary for each label to go across low density regions. Thus, the prediction of the k-th label, F_{ik}, is the solution to the optimization problem

$$\min_f 1^T f + \alpha \sum_{i,j} (f_i - f_j)^2 W_{ij} + \beta \sum_{Ŷ_{ik}=1} \ell(\hat{Y}_{ik}, f_i)$$

s.t. $f \in \{0, 1\}^n$, (1)

where \alpha and \beta are the controlling parameters, \ell(·, ·) is the loss function occurring only on the location (i, k) where \hat{Y}_{ik} = 1. Solving Eq. 1 is hard and therefore, we relax the domain of f from \{0, 1\}^n to [0, 1]^n. If we define \ell(·, ·) as a convex function, this problem is a convex optimization problem. Denote $D = \text{diag}(d_1, d_2, \ldots, d_n)$ a diagonal matrix with diagonal elements $d_i = \sum_j W_{ij}$ and $L = D - W$ the Laplacian matrix. Adopting the squared loss for the loss function, we can transform Eq. 1 to the following QP (quadratic programming) problem

$$\min_f f^T (\alpha L + \beta Y_k) f - 2(\beta \hat{Y}_{ik} + 1)^T f$$

s.t. $f \in [0, 1]^n$, (2)

where $Y_k = \text{diag}(\hat{Y}_{1k}, \hat{Y}_{2k}, \ldots, \hat{Y}_{nk})$ is a diagonal matrix.
For each label, after obtaining the continuous \( f \), we get the ranking of instances. To get the prediction we need to learn a threshold \( t \) to discretize \( f \) as \( F_{ik} = \delta(f_i \geq t) \), where \( \delta \) is the indicator function which takes 1 when \( f_i \geq t \) and 0 otherwise. Since we only have positive data, we could not adopt regression methods to train a model predicting \( f \) like in (Elisseeff and Weston 2002). Note that as \( t \) varies, \( F_k \) can only take \( n+1 \) different values. So, a simple strategy is to choose the \( F_k \) that minimizes Eq. 1.

Shared Base Similarities

In Eq. 1 all the labels share the same similarity matrix \( W \) derived from the feature space. In many applications, however, there is a gap between the similarity for features and the similarity for semantic meanings. Similarity which is helpful for the classification should be dependent on the corresponding label. Take image annotation task for example. Assume that we have three images, where the first image has labels car and road, the second has labels people and road, and the third has labels people, computer and office. When the label road is concerned, the second image is similar to the first one and dissimilar to the third one because it shares road with the first image while the third image has no road. While when the label people is concerned, the second image is similar to the third one and dissimilar to the first one. It is evident that it is better to learn different similarity matrices for different labels, while simply using the same similarity matrix may lose useful information.

In order to embed the label information, we learn the new similarity matrix for the \( k \)-th label, denoted as \( W^k \), by maximizing the kernel alignment with \( \hat{Y}_{ik} \), i.e.,

\[
\hat{Y}_{ik} W^k \hat{Y}_{ik} / \| W^k \|_F.
\]

Note that when \( \hat{Y}_{ik} = 0 \), we have \( \hat{Y}_{ik} W^k \hat{Y}_{ik} = 0 \) for any \( j \), which means that the uncertainty of \( \hat{Y}_{ik} \) will not contribute to the kernel alignment. In the meanwhile, similar to (Liu et al. 2009), we treat the original similarity matrix as a noisy observation of \( W^k \) and require \( W^k \) be in the neighborhood of \( W \). Thus, we have

\[
\begin{align*}
\min_{W^k} & \quad -\hat{Y}_{ik}^T W^k \hat{Y}_{ik} + \gamma \| W^k - W \|^2_F \\
\text{s.t.} & \quad \| W^k \|_F = C, \quad W^k \succeq 0,
\end{align*}
\]

where \( \gamma \) is the controlling parameter, \( C \) is a constant used to control the Frobenius norm of \( W^k \), and \( W^k \succeq 0 \) means \( W^k \) is a PSD matrix. However, contrary to the extreme where all the labels share the same similarity matrix in Eq. 1, Eq. 3 goes to the other extreme where the label correlation is not taken into consideration. In order to get a balance, we require all the \( W^k \)’s share something. Similar strategies have been used before. For example, in (Yan, Tesić, and Smith 2007), a group of base models is shared by all the models corresponding to different labels; in (Ji et al. 2008), a common subspace is shared by all the classification tasks. Here, we assume that there is a group of low-rank base similarities, and the appropriate similarities between instances for different labels can be derived from these base similarities, i.e., \( W^k = \sum_i \lambda_i^k W_i \) where \( W_i = v_i v_i^T \) and \( v_i \) is the \( i \)-th orthonormal eigenvector of \( W \) as \( W = \sum_i \eta_i v_i v_i^T \). This is related to some studies in the MIML (multi-instance multi-label learning) framework where the high-level concept is derived from a set of sub-concepts (Zhou et al. 2008). To ensure \( W^k \succeq 0 \), we require \( \lambda_i^k \succeq 0 \) for each \( k \) and \( i \). Denote \( \lambda^k = (\lambda_1^k, \ldots, \lambda_n^k) \), we have \( \| W^k \|_F = \| \lambda^k \|_2 \) and the constraint becomes \( \| \lambda^k \|_2 = C \). For simplicity, we define \( C = \sum_i \eta_i \) and replace the \( \ell_2 \)-norm constraint on \( \lambda^k \) by the \( \ell_1 \)-norm constraint. Thus Eq. 3 becomes

\[
\begin{align*}
\min_{\lambda} & \quad -\hat{Y}_{ik}^T \left( \sum_i \lambda_i W_i \right) \hat{Y}_{ik} + \gamma \| \sum_i \lambda_i W_i - W \|^2_F \\
\text{s.t.} & \quad \| \lambda \|_1 = C, \quad \lambda_i \succeq 0.
\end{align*}
\]

Note that

\[
\hat{Y}_{ik}^T \left( \sum_i \lambda_i W_i \right) \hat{Y}_{ik} = \sum_i \lambda_i \left( \hat{Y}_{ik}^T W_i \hat{Y}_{ik} \right) = \sum_i \lambda_i \left( \hat{Y}_{ik}^T v_i v_i^T \hat{Y}_{ik} \right) = u^T \lambda,
\]

where \( u_i = \hat{Y}_{ik}^T v_i v_i^T \hat{Y}_{ik} \). We have

\[
\begin{align*}
\left\| \sum_i \lambda_i W_i - W \right\|^2_F &= \sum_{i,j} \lambda_i \lambda_j \text{tr} (W_i W_j) - 2 \sum_i \lambda_i \text{tr} (W_i W) \\
&= \sum_{i,j} \lambda_i \lambda_j v_i^T v_j v_j^T v_i - 2 \sum_{i,j} \lambda_i \eta_j v_i^T v_j v_j^T v_i \\
&= \lambda^T I \lambda - 2 \eta^T \lambda,
\end{align*}
\]

where \( \text{tr}(\cdot) \) is the trace of a matrix and \( I \) is the identity matrix. Overall, the objective function for the \( k \)-th label becomes

\[\text{Input:}\]
\[Y : n \times m \text{ partial label matrix}\]
\[W : n \times n \text{ PSD similarity matrix}\]
\[\text{Output:}\]
\[F : n \times m \text{ predicted label matrix}\]
\[\text{Process:}\]
\[1 \quad \text{Decompose } W = \sum_i \eta_i v_i v_i^T.\]
\[2 \quad C = \sum_i \eta_i, \quad W_i = v_i v_i^T.\]
\[3 \quad \text{for } 1 \leq k \leq m\]
\[4 \quad u_k = \hat{Y}_{ik}^T v_i v_i^T \hat{Y}_{ik} .\]
\[5 \quad \text{Solve } \min_{\lambda} \gamma \lambda^T I \lambda - (u + 2\eta)^T \lambda \quad \text{s.t. } \lambda_i \geq 0 \]
\[6 \quad W^k = \sum_i \lambda_i W_i, \quad L^k = D^k - W^k \]
\[7 \quad \text{Solve } \min_{f} \left( \alpha L^k + \beta T_s \right) f - 2(\beta \hat{Y}_{ik} + 1)^T f \quad \text{s.t. } f \in [0, 1]^n \]
\[8 \quad \text{Vary the threshold and choose the discretized } f \text{ minimizing the above object function as } F_k .\]
\[9 \quad \text{end for}\]

Figure 1: Pseudo-code of the WELL method

\[1\text{For simplicity of discussion, we drop the superscript } k \text{ for } \lambda^k .\]
\[2\text{In the derivation we omit the terms being constant with } \lambda .\]
Table 1: Experimental results (mean±std) on Yeast data. ↑ indicates “the larger, the better”; ↓ indicates “the smaller, the better”. The best performance and its comparable performances are bolded (statistical significance examined via pairwise t-tests at 95% significance level).

<table>
<thead>
<tr>
<th>WL Ratio</th>
<th>WELL</th>
<th>WELL-Minus</th>
<th>ML-kNN</th>
<th>Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>20%</td>
<td>0.197±0.001</td>
<td>0.203±0.008</td>
<td>0.297±0.002</td>
<td>0.207±0.001</td>
</tr>
<tr>
<td>30%</td>
<td>0.162±0.000</td>
<td>0.167±0.005</td>
<td>0.275±0.005</td>
<td>0.169±0.000</td>
</tr>
<tr>
<td>40%</td>
<td>0.139±0.001</td>
<td>0.144±0.005</td>
<td>0.260±0.010</td>
<td>0.150±0.001</td>
</tr>
<tr>
<td>50%</td>
<td>0.083±0.000</td>
<td>0.084±0.002</td>
<td>0.210±0.005</td>
<td>0.088±0.002</td>
</tr>
<tr>
<td>60%</td>
<td>0.074±0.000</td>
<td>0.075±0.002</td>
<td>0.198±0.001</td>
<td>0.080±0.001</td>
</tr>
</tbody>
</table>

\[
\min_{\lambda} \gamma \lambda^T I \lambda - (u + 2\eta)^T \lambda \\
s.t. \quad 1^T \lambda = C, \quad \lambda_i \geq 0.
\]

Note that Eq. 5 has only equality constraint and $I$ is very sparse. So, this QP problem can be solved very efficiently by SMO (sequential minimal optimization) (Platt 1999). The pseudo-code of the WELL is summarized in Figure 1.

**Empirical Study**

We compare the WELL method with state-of-the-art multi-label learning methods RankSVM (Elisseeff and Weston 2002) and ML-kNN (Zhang and Zhou 2007), and PU-learning methods Elkan08 (abbreviated for the method in (Elkan and Noto 2008)), iter-SVM and bias-SVM (Liu et al. 2003). We also evaluate a degenerated version of WELL, denoted as WELL-Minus, where the original similarity matrix $W$ is shared for all the labels, to study the utility of exploiting label correlation.

We use multi-label classification criteria Hamming Loss, Macro-F1 and Micro-F1 to measure the performance. Hamming Loss evaluates how many times an instance-label pair is misclassified; Macro-F1 averages the F1 measure on the predictions of different labels as a whole. Details can be found in (Zhang and Zhou 2010). Three real-world tasks, i.e., gene functional analysis, text classification and image annotation, are included in experiments. On each data set, we vary the weak label ratio (WL ratio), defined as $||Y_1||_1/||Y||_1$, of each instance from 20% to 60% with 10% as interval to study the performance of different methods.

For WELL, $\alpha$ and $\beta$ are fixed as 100 and 10, respectively. This setting is sub-optimal and we will study how to set the parameters better in the future. We observed that the performance does not change much as $\alpha$ and $\beta$ vary around the fixed values. Another parameter, $\gamma$, is tuned from $\{10^i | i = 0, 1, \cdots, 4\}$ based on the best performance on kernel alignment using five-fold cross-validation on training data. The original similarity matrix used in WELL and the kernel used in SVM-based methods are rbf kernels and the kernel width is fixed to 1. The SVM with rbf kernel used in all SVM-based methods is implemented by libSVM (Lin, Lin, and Weng 2007). Parameters of WELL-Minus are set to the same values as those for WELL. For ML-kNN, we set $k = 10$ as suggested in (Zhang and Zhou 2007). For other parameters of the compared methods, we choose from the pool of $\{10^i | i = -4, -3, \cdots, 3, 4\}$ according to the best performance on Hamming Loss on the ground-truth. Note that by such a parameter setting, the comparison is unfavorable to our WELL method; however, we will see that even in such setting the performance of WELL is still superior to the compared methods.

**Yeast Gene Functional Analysis**

The first task is to predict the gene functional classes of the Yeast *Saccharomyces cerevisiae*. The Yeast data set investigated in (Elisseeff and Weston 2002) is used. The data set we used here contains 1,500 examples and 14 class labels. The average number of labels for each instance is $4.2\pm1.57$.

Results are shown in Table 1. It can be seen that WELL performs significantly better than all the other approaches except WELL-Minus on Hamming Loss when the WL ratio is larger than 50%. As the WL ratio decreases, the advantage of WELL to other methods becomes more apparent. One reason is that WELL uses not only the similarity between instances but also the similarity between labels.

**Text Classification**

The second task is a text classification task in SIAM Text Mining Competition (TMC) 2007. This data set is a subset of the Aviation Safety Reporting System (ASRS) data.
set, which contains a huge number of documents. Each document is an aviation safety report documenting one or more problems that occurred on certain flights. The goal is to label the documents with respect to what types of problems they describe. Each document may belong to more than one class. Here we use the pre-processed version.

The data set we used here contains 1,000 examples and 15 class labels. The average number of labels for each instance is 3.57 ± 0.73. The dimensionality of this data set is high (30,438), and therefore we first perform PCA to reduce the dimensionality to 7,000.

Results are summarized in Table 2. It can be seen that WELL performs significantly better than all other methods on all the evaluation criteria.

### Image Annotation

The third task is image annotation. The data set we used was released by Microsoft Research Asia (MSRA). Each image is described by seven descriptors (i.e., 7 groups of features) including color moment, correlogram, EDH, face, HSV, RGB and wavelet texture. The total number of features is 899. All the labels are annotated by human. The data set we used here contains 1,000 examples and 15 class labels. The average number of labels for each instance in this subset is 6.76 ± 0.94.

Results are summarized in Table 3. It can be seen that WELL always performs the best on all the criteria. Figure 2 shows two examples of the results. The PU-learning methods, Elkan08 and bias-SVM, predict much more labels than the ground-truth. It is probably because those methods are designed to handle balanced data which is unusual in the case of multi-label learning. Also, note that although bias-SVM and Elkan08 predict many labels, bias-SVM misses clothing for the first image and Elkan08 misses leaf for the second image. clothing is related to women and leaf is related to jungle; this implies that those methods have not utilized the label correlation well. The outputs of iter-SVM are as same as the inputs. This may be caused by that it treats the unlabeled data as negative and uses a self-training strategy, while the empirical loss is small at the beginning and thus the outputs will not change greatly during the iterations. Methods designed for classic multi-label learning, ML-kNN and RankSVM, predict almost as same as the inputs. This is not difficult to understand since they are designed for full label setting and tend to regard unassigned labels as negative. WELLMINUS also outputs more labels than the ground-truth, which may be caused by that it assumes all labels share the same similarity matrix and so the boundaries corresponding to different labels may be similar.

### Conclusion

In this paper, we study the weak label problem which is a new kind of multi-label learning problem, where only a partial label set associated with each training example is provided. We propose the WELL method which considers the inherent class imbalance of the weak label problem and enforces the classification boundary for each label to go across low density regions. We formulate the objective as a quadratic programming problem which can be solved efficiently. To exploit label correlations, we assume that there is a group of low-rank base similarities, and the appropriate similarities between instances for different labels can be derived from these base similarities. Improving the efficiency of our method and applying it to large scale database are interesting issues for future study.

### References

Table 3: Experimental results (mean±std) on MSRA data. † indicates “the larger, the better”; ‡ indicates “the smaller, the better”. The best performance and its comparable performances are bolded (statistical significance examined via pairwise t-tests at 95% significance level).

<table>
<thead>
<tr>
<th>WL</th>
<th>Ratio</th>
<th>WELL</th>
<th>WELLMINUS</th>
<th>ML-κNN</th>
<th>RankSVM</th>
<th>Ellkan08</th>
<th>iter-SVM</th>
<th>bias-SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hamming Loss†</td>
<td>20%</td>
<td>0.249±0.004</td>
<td>0.315±0.000</td>
<td>0.434±0.001</td>
<td>0.316±0.000</td>
<td>0.549±0.000</td>
<td>0.318±0.000</td>
<td>0.316±0.001</td>
</tr>
<tr>
<td></td>
<td>30%</td>
<td>0.224±0.007</td>
<td>0.261±0.002</td>
<td>0.419±0.003</td>
<td>0.277±0.000</td>
<td>0.499±0.072</td>
<td>0.278±0.000</td>
<td>0.277±0.000</td>
</tr>
<tr>
<td></td>
<td>40%</td>
<td>0.198±0.007</td>
<td>0.212±0.004</td>
<td>0.380±0.004</td>
<td>0.237±0.000</td>
<td>0.515±0.028</td>
<td>0.238±0.000</td>
<td>0.237±0.000</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>0.155±0.001</td>
<td>0.155±0.000</td>
<td>0.312±0.013</td>
<td>0.174±0.000</td>
<td>0.371±0.004</td>
<td>0.174±0.000</td>
<td>0.174±0.000</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>0.127±0.003</td>
<td>0.128±0.001</td>
<td>0.265±0.001</td>
<td>0.142±0.000</td>
<td>0.284±0.078</td>
<td>0.142±0.000</td>
<td>0.142±0.000</td>
</tr>
<tr>
<td>Macro-F1‡</td>
<td>20%</td>
<td>0.634±0.009</td>
<td>0.473±0.001</td>
<td>0.068±0.002</td>
<td>0.464±0.000</td>
<td>0.619±0.000</td>
<td>0.460±0.000</td>
<td>0.585±0.171</td>
</tr>
<tr>
<td></td>
<td>30%</td>
<td>0.676±0.015</td>
<td>0.591±0.003</td>
<td>0.129±0.016</td>
<td>0.553±0.000</td>
<td>0.588±0.044</td>
<td>0.559±0.001</td>
<td>0.553±0.000</td>
</tr>
<tr>
<td></td>
<td>40%</td>
<td>0.726±0.012</td>
<td>0.695±0.006</td>
<td>0.276±0.010</td>
<td>0.644±0.000</td>
<td>0.554±0.036</td>
<td>0.642±0.000</td>
<td>0.644±0.000</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>0.800±0.002</td>
<td>0.799±0.000</td>
<td>0.488±0.033</td>
<td>0.761±0.000</td>
<td>0.645±0.008</td>
<td>0.760±0.000</td>
<td>0.762±0.000</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>0.842±0.003</td>
<td>0.841±0.001</td>
<td>0.606±0.002</td>
<td>0.814±0.000</td>
<td>0.729±0.049</td>
<td>0.814±0.000</td>
<td>0.814±0.000</td>
</tr>
<tr>
<td>Micro-F1‡</td>
<td>20%</td>
<td>0.643±0.009</td>
<td>0.472±0.001</td>
<td>0.075±0.003</td>
<td>0.460±0.000</td>
<td>0.621±0.000</td>
<td>0.455±0.000</td>
<td>0.384±0.176</td>
</tr>
<tr>
<td></td>
<td>30%</td>
<td>0.688±0.014</td>
<td>0.599±0.002</td>
<td>0.141±0.016</td>
<td>0.556±0.000</td>
<td>0.593±0.040</td>
<td>0.555±0.000</td>
<td>0.556±0.000</td>
</tr>
<tr>
<td></td>
<td>40%</td>
<td>0.732±0.012</td>
<td>0.699±0.006</td>
<td>0.292±0.011</td>
<td>0.643±0.000</td>
<td>0.560±0.035</td>
<td>0.641±0.000</td>
<td>0.643±0.000</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>0.802±0.002</td>
<td>0.801±0.000</td>
<td>0.504±0.033</td>
<td>0.760±0.000</td>
<td>0.650±0.007</td>
<td>0.759±0.000</td>
<td>0.760±0.000</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>0.843±0.003</td>
<td>0.842±0.001</td>
<td>0.618±0.002</td>
<td>0.813±0.000</td>
<td>0.732±0.048</td>
<td>0.813±0.000</td>
<td>0.814±0.000</td>
</tr>
</tbody>
</table>

Figure 2: Examples from MSRA data set. The ground-truth, input and predicted labels are shown on the right side of each image.