# Are You Sure? Confidence in Prediction of Dependency Tree Edges

**Avihai Mejer**
Department of Electrical Engineering
Technion-Israel Institute of Technology
Haifa 32000, Israel
amejer@tx.technion.ac.il

**Koby Crammer**
Department of Electrical Engineering
Technion-Israel Institute of Technology
Haifa 32000, Israel
koby@ee.technion.ac.il

## Abstract

We describe and evaluate several methods for estimating the confidence in the per-edge correctness of a predicted dependency parse. We show empirically that the confidence is associated with the probability that an edge is selected correctly and that it can be used to detect incorrect edges very efficiently. We evaluate our methods on parsing text in 14 languages.

## 1 Introduction

Dependency parsers construct directed edges between words of a given sentence to their arguments according to syntactic or semantic rules. We use MSTParser of McDonald et al. (2005) and focus on non-projective dependency parse *trees* with non-typed (unlabeled) edges. MSTParser produces a parse tree for a sentence by constructing a full, directed and weighted graph over the words of the sentence, and then outputting the maximal spanning tree (MST) of the graph. A linear model is employed for computing the weights of the edges using features depending on the two words the edge connects. Example features are the distance between the two words, words identity and words part-of-speech. MSTParser is training a model using online learning and specifically the MIRA algorithm (Crammer et al., 2006). The output of MSTParser is the highest scoring parse tree, it is not accompanied by any additional information about its quality.

In this work we evaluate few methods for estimating the confidence in the correctness of the prediction of a parser. This information can be used in several ways. For example, when using parse trees as input to another system such as machine translation, the confidence information can be used to correct inputs with low confidence. Another example is to guide manual validation to outputs which are more likely to be erroneous, saving human labor. We adapt methods proposed by Mejer and Crammer (2010) in order to produce per-edge confidence estimations in the prediction. Specifically, one approach is based on sampling, and another on a generalization of the concept of margin. Additionally, we propose a new method based on combining both approaches, and show that is outperforms both.

## 2 Confidence Estimation In Prediction

MSTParser produces the highest scoring parse trees using the trained linear model with no additional information about the confidence in the predicted tree. In this work we compute per-edge confidence scores, that is, a numeric confidence value, for all edges predicted by the parser. Larger score values indicate higher confidence. We use three confidence estimation methods that were proposed for sequence labeling (Mejer and Crammer, 2010), adapted here for dependency parsing. A fourth method, described in Sec. 3, is a combination of the two best performing methods.

The first method, named **Delta**, is a margin-based method. For computing the confidence of each edge the method generates an additional parse-tree, which is the best parse tree that is forced not to contain the specific edge in question. The confidence score of the edge is defined as the difference in the scores be-

*2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 573–576,
Montréal, Canada, June 3-8, 2012. ©2012 Association for Computational Linguistics

tween the two parse trees. The score of a tree is the sum of scores of the edges it contains. These confidence scores are always positive, yet *not* limited to [0, 1]. Delta method does not require parameter tuning.

The second method, named **Weighted K-Best (WKB)**, is a deterministic method building on properties of the inference algorithm. Specifically, we use k-best Maximum Spanning Tree algorithm (Hall, 2007) to produce the K parse trees with the highest score. This collection of K-trees is used to compute the confidence in a predicted edge. The confidence score is defined to be the weighted-fraction of parse trees that contain the edge. The contribution of different trees to compute this fraction is proportional to their absolute score, where the tree with the highest score has the largest contribution. Only trees with positive scores are included. The computed score is in the range [0, 1]. The value of K was tuned using a development set (optimizing the average-precision score of detecting incorrect edges, see below) and for most datasets K was set to a value between $10 - 20$.

The third method, **K Draws by Fixed Standard Deviation (KD-Fix)** is a probabilistic method. Here we sample $K$ weight vectors using a Gaussian distribution, for which the mean parameters are the learned model and isotropic covariance matrix with fixed variance $s^2$. The value $s$ is tuned on a development set (optimizing the average-precision score of detecting incorrect edges). The confidence of each edge is the probability of this edge induced from the distribution over parameters. We approximate this quantity by sampling K parse trees, each obtained by finding the MST when scores are computed by one of K sampled models. Finally, the confidence score of each edge predicted by the model is defined to be the fraction of parse trees among the K trees that contain this edge. Formally, the confidence score is $\nu = j/K$ where $j$ is the number of parse trees that contain this edge ($j \in \{0...K\}$) so the score is in the range [0, 1]. We set $K = 50$.

Finally, we describe below a fourth method, we call **KD-Fix+Delta**, which is a weighted-linear combination of KD-Fix and Delta.

## 3 Evaluation

We evaluated the algorithms using 13 languages used in CoNLL 2006 shared task[1], and the English Penn Treebank. The number of training sentences is between 1.5-72K, with an average of $20K$ sentences and 50K-1M words. The test sets contain $\sim 400$ sentences and $\sim 6K$ words for all datasets, except English with $2.3K$ sentences and $55K$ words. Parameter tuning was performed on development sets with 200 sentences per dataset. We trained a model per dataset and used it to parse the test set. Predicted edge accuracy of the parser ranges from $77\%$ on Turkish to $93\%$ on Japanese, with an average of $85\%$. We then assigned each predicted edge a confidence score using the various confidence estimation methods.

**Absolute Confidence:** We first evaluate the accuracy of the actual confidence values assigned by all methods. Similar to (Mejer and Crammer, 2010) we grouped edges according to the value of their confidence. We used 20 bins dividing the confidence range into intervals of size 0.05. Bin indexed $j$ contains edges with confidence value in the range $[\frac{j-1}{20}, \frac{j}{20}]$ , $j = 1..20$. Let $b_j$ be the center value of bin $j$ and let $c_j$ be the fraction of edges predicted correctly from the edges assigned to bin $j$. For a good confidence estimator we expect $b_j \approx c_j$.

Results for 4 datasets are presented in Fig. 1. Plots show the measured fraction of correctly predicted edges $c_j$ vs. the value of the center of bin $b_j$. Best performance is obtained when a line corresponding to a method is close to the line $y = x$. Results are shown for KD-Fix and WKB; Delta is omitted as it produces confidence scores out of [0, 1]. In two of the shown plots (Chinese and Swedish) KD-Fix (circles) follows closely the expected accuracy line. In another plot (Danish) KD-Fix is too pessimistic with line above $y = x$ and in yet another case (Turkish) it is too optimistic. The distribution of this qualitative behavior among the 14 datasets is: too optimistic in 2 datasets, too pessimistic in 7 and close to the line $y = x$ in 5 datasets. The confidence scores produced by the WKB are in general worse than KD-Fix, too optimistic in some confidence range

[1] Arabic, Bulgarian, Chinese, Czech, Danish, Dutch, German, Japanese, Portuguese, Slovene, Spanish, Swedish and Turkish . See `http://nextens.uvt.nl/~conll/`

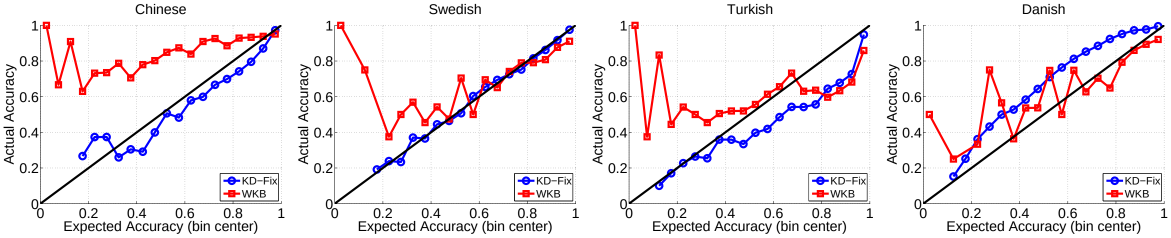

Figure 1: Evaluation of KD-Fix and WKB by comparing predicted accuracy vs. actual accuracy in each bin on 4 datasets. Best performance is obtained for curves close to the line *y*=*x* (black line). Delta method is omitted as its output is not in the range [0, 1].

| | KD<br>Fix | WKB | Delta | KD-Fix<br>+Delta | Random |
|---|---|---|---|---|---|
| Avg-Prec | **0.535** | 0.304 | 0.518 | **0.547** | 0.147 |
| Prec @10% | **0.729** | 0.470 | 0.644 | **0.724** | 0.145 |
| Prec @90% | 0.270 | 0.157 | **0.351** | **0.348** | 0.147 |
| RMSE | **0.084** | 0.117 | - | - | 0.458 |

Table 1: Row 1: Average precision in ranking all edges according confidence values. Rows 2-3: Precision in detection of incorrect edges when detected 10% and 90% of all the incorrect edges. Row 4: Root mean square error. All results are averaged over all datasets.

and too pessimistic in another range. We computed the root mean square-error (RMSE) in predicting the bin center value given by $\sqrt{(\sum_j n_j(b_j - c_j)^2)/(\sum_j n_j)}$ , where $n_j$ is the number of edges in the $jth$ bin. The results, summarized in the $4th$ row of Table 1, support the observation that KD-Fix performs better than WKB, with smaller RMSE.

**Incorrect Edges Detection:** The goal of this task is to efficiently detect incorrect predicted-edges. We ranked all predicted edges of the test-set (per dataset) according to their confidence score, ordering from low to high. Ideally, erroneous edges by the parser are ranked at the top. A summary of the average precision, computed at all ranks of erroneous edges, (averaged over all datasets, due to lack of space), for all confidence estimation methods is summarized in the first row of Table 1. The average precision achieved by random ordering is about equal to the error rate for each dataset. The Delta method improves significantly over both the random ordering and WKB. KD-Fix achieves the best performance in 12 of 14 datasets and the best average-performance. These results are consistent with the results obtained for sequence labeling by Mejer and Crammer (2010).

Average precision summarizes the detection of all incorrect edges into a single number. More refined analysis is encapsulated in Precision-Recall (PR) plots, showing the precision as more incorrect edges are detected. PR plots for three datasets are shown in Fig. 2. From these plots (applied also to other datasets, omitted due to lack of space) we observe that in most cases KD-Fix performs significantly better than Delta in the early detection stage (first 10-20% of the incorrect edges), while Delta performs better in late detection stages (last 10-20% of the incorrect edges). The second and third rows of Table 1 summarize the precision after detecting only 10% incorrect edges and after detecting 90% of the incorrect edges, averaged over all datasets. For example, in Czech and Portuguese plots of Fig. 2, we observe an advantage of KD-Fix for low recall and an advantage of Delta in high recall. Yet for Arabic, for example, KD-Fix outperforms Delta along the entire range of recall values.

KD-Fix assigns at most K distinct confidence values to each edge - the number of models that agreed on that particular edge. Thus, when edges are ranked according to the confidence, all edges that are assigned the same value are ordered randomly. Furthermore, large fraction of the edges, $\sim 70 - 80\%$, are assigned one of the top-three scores (i.e. *K-2, K-1, K*). As a results, the precision performance of KD-Fix drops sharply for recall values of 80% and above. On the other hand, we hypothesize that the lower precision of Delta at low recall values (diamond in Fig. 2) is because by definition Delta takes into account only two parses, ignoring additional possible parses with score close to the highest score. This makes Delta method more sensitive to small differences in score values compared to KD-Fix.

Based on this observation, we propose combining both KD-Fix and Delta. Our new method sets the confidence score of an edge to be a weighted mean of the score values of KD-Fix and Delta, with weights *a* and *1-a*, respectively. We use a value

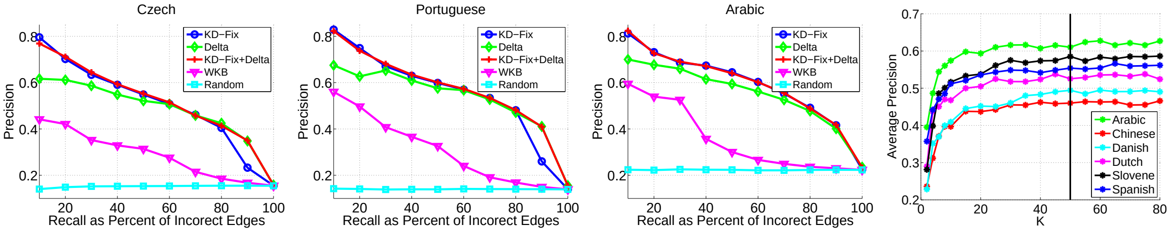

Figure 2: (Best shown in color.) Three left plots: Precision in detection of incorrect edges as recall increases. Right plot: Effect of $K$ value on KD-Fix method performance (for six languages, the remaining languages follow similar trend, omitted for clarity).

$a \approx 1$, so if the confidence value of two edges according to KD-Fix is different, the contribution of the score from Delta is negligible, and the final score is very close as score of only KD-Fix. On the other hand, if the score of KD-Fix is the same, as happens for many edges at high recall values, then Delta breaks arbitrary ties. In other words, the new method first ranks edges according to the confidence score of KD-Fix, then among edges with equal KD-Fix confidence score a secondary order is employed using Delta. Not surpassingly, we name this method **KD-Fix+Delta**. This new method enjoys the benefits of the two methods. From the first row of Table 1 we see that it achieves the highest average-precision averaged over the 14 datasets. It improves average-precision over KD-Fix in 12 of 14 datasets and over Delta in all 14 datasets. From the second and third row of the table, we see that it has Precision very close to KD-Fix for recall of 10% (0.729 vs. 0.724), and very close to Delta for recall of 90% (0.351 vs. 0.348). Moving to Fig. 2, we observe that the curve associated with the new method (red ticks) is in general as high as the curves associated with KD-Fix for low values of recall, and as high as the curves associated with Delta for large values of recall.

To illustrate the effectiveness of the incorrect edges detection process, Table 2 presents the number of incorrect edges detected vs. number of edges inspected for the English dataset. The test set for this task includes $55K$ words and the parser made mistake on $6,209$ edges, that is, accuracy of $88.8\%$. We see that using the ranking induced by KD-Fix+Delta method, inspection of 550, 2750 and 5500 edges $(1, 5, 10\%$ of all edges), allows detection of $6.6 - 46\%$ of all incorrect edges, over $4.5$ times more effective than random validation.

| Edges inspected<br>(% of total edges) | Incorrect edges detected<br>(% of incorrect edges) |
|---|---|
| 550 (1%) | 412 (6.6%) |
| 2,750 (5%) | 1,675 (27%) |
| 5,500 (10%) | 2,897 (46%) |

Table 2: Number of incorrect edges detected, and the corresponding percentage of *all mistakes*, after inspecting $1 - 10\%$ of all edges, using ranking induced by KD-Fix+Delta method.

**Effect of $K$ value on KD-Fix method performance** The right plot of Fig. 2 shows the average-precision of detecting incorrect edges on the test set using the KD-Fix method for $K$ values ranging between 2 and 80. We see that even with $K = 2$, only two samples per sentence, the average precision results are much better than random ranking in all tasks. As $K$ is increased the results improve until reaching maximal results at $K \approx 30$. Theoretical calculations, using concentration inequalities, show that accurate estimates based on the sampling procedure requires $K \approx 10^2 - 10^3$. Yet, we see that for practical uses, smaller $K$ values by $1 - 2$ order of magnitude is suffice.

## References

[Crammer et al.2006] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive-aggressive algorithms. *JMLR*, 7:551–585.

[Hall2007] Keith Hall. 2007. k-best spanning tree parsing. In *In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

[McDonald et al.2005] R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *HLT/EMNLP*.

[Mejer and Crammer2010] A. Mejer and K. Crammer. 2010. Confidence in structured-prediction using confidence-weighted models. In *EMNLP*.