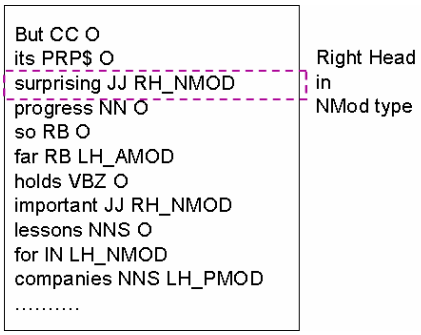

Figure 3: Sequential tagging model for neighbor parse

Unfortunately, for some languages, like Chinese and Czech, training with CRF is because of the large number of features and the head relations. To make it practical, we focus on just three types: left head, right head, and out-of-neighbor. This effectively reduces most of the feature space for the CRF. The training time for the neighbor parser with only three categories is less than 5 minutes while it takes three days with taking all the relation tag into account.

2.2 Root Parser

After the neighbor parse, the tagged labels are good features for the root parse. In the second stage, the root parser identifies the *root* words in the sentence. Nevertheless, for some languages, such as Arabic and Czech, the roots might be several types as against to Chinese and English in which the number of labels of roots is merely one. Similar to the neighbor parser, we also take the root label into account. As noted, for Chinese and English, the goal of the root parser can be reduced to determine whether the current word is root or not.

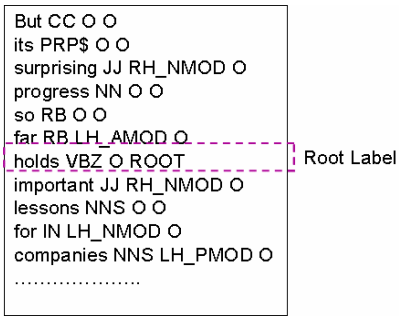

Figure 4: Sequential tagging model for neighbor parse

Similar to the neighbor parse, the root parsing task can also be treated as a sequential tagging problem. Figure 4 shows the basic concept of the root parser. The third column is mainly derived from the neighbor parser, while the fourth column represents whether the current word is a root with relation or not.

2.3 Parsing Algorithm

After adding the neighbor and root parser output as features, in the final stage, the modified Yamada's shift-reduce parsing algorithm (Yamada and Matsumoto, 2003) is then run. This method is deterministic and can deal with projective data only. There are three basic operation (action) types: Shift (S), Left (L), and Right (R). The operation is mainly determined via the classifier according to the selected features (see 2.4). Each time, the operation is applied to two unparsed words, namely, focus and next. If there exists an arc between the two words (either left or right), then the head of focus or next word is found; otherwise (i.e., shift), next two words are considered at next stage. This method could be economically performed via maintaining two pointers, focus, and next without an explicit stack. The parse operation is iteratively run until no more relation can be found in the sentence.

In 2006, Chang et al. (2006) further reported that the use of "step-back" in comparison to the original "stay". Furthermore, they also add the "wait-left" operations to prevent the "too early reduce" problems. In this way, the parse actions can be reduced to be bound in $3n$ where n is the number of words in a sentence.

Now we compare the adopted parsing algorithm in this year to the one we employed last year (Wu et al., 2006a). The common characteristics are:

1. the same number of parse operations (4)
2. shift-reduce
3. linearly scaled
4. deterministic and projective

On the contrary, their parse actions are quite different. Therefore these two methods have different run time. This gives the two methods rise to different iterative times. The main reason is that the step-back might trace back to previous words, which can be viewed as pop the top words on the stack back to the unparsed strings, while the Nivre's method does not trace-back any two words