We use all the features of the SVM-based and ME-based argument classification models. All SVM classifiers are realized using SVM-Light with a polynomial kernel of degree 2. The ME-based model is implemented based on Zhang's MaxEnt toolkit[1] and L-BFGS (Nocedal and Wright, 1999) method to perform parameter estimation.

2.3 Classification Model Incorporation

We now explain how we incorporate the SVM-based and ME-based argument classification models. After argument classification, we acquire two scoring matrices, $\mathbf{P}_{ME}$ and $\mathbf{P}_{SVM}$, respectively. Incorporation of these two models is realized by weighted summation of $\mathbf{P}_{ME}$ and $\mathbf{P}_{SVM}$ as follows:

$$\mathbf{P}' = w_{ME}\mathbf{P}_{ME} + w_{SVM}\mathbf{P}_{SVM}$$

We use $\mathbf{P}'$ for the objective coefficients of the ILP described in Section 2.4.

2.4 Integer Linear Programming (ILP)

To represent full parsing information as features, there are still several syntactic constraints on a parsing tree in the SRL problem. For example, on a path of the parsing tree, there can be only one constituent annotated as a non-null argument. However, it is difficult to encode this constraint in the argument classification models. Therefore, we apply integer linear programming to resolve inconsistencies produced in the argument classification stage.

According to Punyakanok et al. (2004), given a set of constituents, **S**, and a set of semantic role labels, **A**, the SRL problem can be formulated as an ILP as follows:

Let z_{ia} be the indicator variable that represents whether or not an argument, a, is assigned to any $S_i \in \mathbf{S}$; and let $p_{ia} = \text{score}(S_i = a)$. The scoring matrix **P** composed of all p_{ia} is calculated by the argument classification models. The goal of this ILP is to find a set of assignments for all z_{ia} that maximizes the following function:

$$\sum_{S_i \in \mathbf{S}} \sum_{a \in \mathbf{A}} p_{ia} z_{ia} .$$

Each $S_i \in \mathbf{S}$ should have one of these argument types, or no type (null). Therefore, we have

$$\sum_{a \in \mathbf{A}} z_{ia} = 1 .$$

Next, we show how to transform the constraints in the filter function into linear equalities or inequalities, and use them in this ILP.

Constraint I: No overlapping or embedding

For arguments $S_{j_1}, \ldots, S_{j_k}$ on the same path in a full parsing tree, only one argument can be assigned to an argument type. Thus, at least $k - 1$ arguments will be *null*, which is represented by ϕ in the following linear equality:

$$\sum_{i=1}^{k} z_{j_i \phi} \geq k - 1 .$$

Constraint II: No duplicate argument classes

Within the same sentence, A0-A5 cannot appear more than once. The inequality for A0 is therefore:

$$\sum_{i=1}^{k} z_{i\text{A0}} \leq 1 .$$

Constraint III: R-XXX arguments

The linear inequalities that represent A0 and its reference type R-A0 are:

$$\forall m \in \{1,\ldots,M\} : \sum_{i=1}^{k} z_{i\text{A0}} \geq z_{m\text{R-A0}} .$$

Constraint IV: C-XXX arguments

The continued argument XXX has to occur before C-XXX. The linear inequalities for A0 are:

$$\forall m \in \{2,\ldots,M\} : \sum_{i=1}^{m-1} z_{j_i\text{A0}} \geq z_{m\text{C-A0}} .$$

Constraint V: Illegal arguments

For each verb, we look up its allowed roles. This constraint is represented by summing all the corresponding indicator variables to 0.

3 Experiment Results

3.1 Data and Evaluation Metrics

The data, which is part of the PropBank corpus, consists of sections from the Wall Street Journal part of the Penn Treebank. All experiments were carried out using Section 2 to Section 21 for training, Section 24 for development, and Section 23 for testing. Unlike CoNLL-2004, part of the Brown corpus is also included in the test set.

3.2 Results

Table 1 shows that our system makes little difference to the development set and Test WSJ. However, due to the intrinsic difference between the WSJ and Brown corpora, our system performs better on Test WSJ than on Test Brown.

[1] http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html