Discriminative features in reversible stochastic attribute-value grammars

Daniël de Kok
University of Groningen
d.j.a.de.kok@rug.nl

Abstract

Reversible stochastic attribute-value grammars (de Kok et al., 2011) use one model for parse disambiguation and fluency ranking. Such a model encodes preferences with respect to syntax, fluency, and appropriateness of logical forms, as weighted features. Reversible models are built on the premise that syntactic preferences are shared between parse disambiguation and fluency ranking.

Given that reversible models also use features that are specific to parsing or generation, there is the possibility that the model is trained to rely on these directional features. If this is true, the premise that preferences are shared between parse disambiguation and fluency ranking does not hold.

In this work, we compare and apply feature selection techniques to extract the most discriminative features from directional and reversible models. We then analyse the contributions of different classes of features, and show that reversible models do rely on task-independent features.

1 Introduction

Reversible stochastic attribute-value grammars (de Kok et al., 2011) provide an elegant framework that fully integrates parsing and generation. The most important contribution of this framework is that it uses one conditional maximum entropy model for fluency ranking and parse disambiguation. In such a model, the probability of a derivation d is conditioned on a set of input constraints c that restrict the set of derivations allowed by a grammar to those corresponding to a particular sentence (parsing) or logical form (generation):

$$p(d|c) = \frac{1}{Z(c)} exp \sum_i w_i f_i(c, d) \quad (1)$$

$$Z(c) = \sum_{d' \in \Omega(c)} exp \sum_i w_i f_i(c, d') \quad (2)$$

Here, $\Omega(c)$ is the set of derivations for input c, $f_i(c, d)$ the value of feature f_i in derivation d of c, and w_i is the weight of f_i. Reversibility is operationalized during training by imposing a constraint on a given feature f_i with respect to the sentences T in the parse disambiguation treebank and logical forms L in the fluency ranking treebank. This constraint is:

$$\sum_{c \in C} \sum_{d \in \Omega(c)} \tilde{p}(c) p(d|c) f_i(c, d) - \tilde{p}(c, d) f_i(c, d) = 0 \quad (3)$$

Where $C = T \cup L$, $\tilde{p}(c)$ is the empirical probability of a set of constraints c, and $\tilde{p}(c, d)$ the joint probability of a set of constraints c and a derivation d.

Reversible stochastic-attribute grammars rest on the premise that preferences are shared between language comprehension and production. For instance, in Dutch, subject fronting is preferred over direct object fronting. If models for parse disambiguation and fluency ranking do not share preferences with respect to fronting, it would be difficult for a parser

Proceedings of the UCNLG+Eval: Language Generation and Evaluation Workshop, pages 54–63,
Edinburgh, Scotland, UK, July 31, 2011.

to recover the logical form that was the input to a generator.

Reversible models incorporate features that are specific to parse disambiguation and fluency ranking, as well as features that are used for both tasks. Previous work (Cahill et al., 2007; de Kok, 2010) has shown through feature analysis that task-independent features are indeed useful in directional models. However, since reversible models assign just one weight to each feature regardless the task, one particular concern is that much of their discriminatory power is provided by task-specific features. If this is true, the premise that similar preferences are used in parsing and generation does not hold.

In this work, we will isolate the most discriminative features of reversible models through feature selection, and make a quantitative and qualitative analysis of these features. Our aim is to to verify that reversible models do rely on features used both in parsing and generation.

To find the most effective features of a model, we need an effective feature selection method. Section 2 describes three such methods: *grafting*, *grafting-light*, and *gain-informed selection*. These methods are compared empirically in Section 4 using the experimental setup described in Section 3. We then use the best feature selection method to perform quantitative and qualitative analyses of reversible models in Sections 5 and 6.

2 Feature selection

Feature selection is a procedure that attempts to extract a subset of discriminative features $S \subset F$ from a set of features F, such that a model using S performs comparable to a model using F and $|S| \ll |F|$.

As discussed in De Kok (2010), a good feature selection method should handle three kinds of redundancies in feature sets: features that rarely change value; features that overlap; and noisy features. Also, for a qualitative evaluation of fluency ranking, it is necessary to have a ranking of features by discriminative power.

De Kok (2010) compares frequency-based selection, correlation selection, and a gain-informed selection method. In that work, it was found that the gain-informed selection method outperforms frequency-based and correlation selection. For this reason we exclude the latter two methods from our experiments. Other commonly used selection methods for maximum entropy models include ℓ_1 regularization (Tibshirani, 1996), grafting (Perkins et al., 2003; Riezler and Vasserman, 2004), and grafting-light (Zhu et al., 2010). In the following sections, we will give a description of these selection methods.

2.1 $\ell 1$ regularization

During the training of maximum entropy models, regularization is often applied to avoid unconstrained feature weights and overfitting. If $L(w)$ is the objective function that is minimized during training, a regularizer $\Omega_q(w)$ is added as a penalty for extreme weights (Tibshirani, 1996):

$$C(w) = L(w) + \Omega_q(w) \tag{4}$$

Given that the maximum entropy training procedure attempts to minimize the negative log-likelihood of the model, the penalized objective function is:

$$C(w) = -\sum_{c,d} \tilde{p}(c,d) log(p(d|c)) + \Omega_q(w) \tag{5}$$

The regularizer has the following form:

$$\Omega_q(w) = \lambda \sum_{i=1}^{n} |w_i|^q$$

Setting $q = 1$ in the regularizer function gives a so-called ℓ_1 regularizer and amounts to applying a double-exponential prior distribution with $\mu = 0$. Since the double-exponential puts much of its probability mass near its mean, the ℓ_1 regularizer has a tendency to force weights towards zero, providing integral feature selection and avoiding unbounded weights. Increasing λ strengthens the regularizer, and forces more feature weights to be zero.

Given an appropriate value for λ, ℓ_1 regularization can exclude features that change value infrequently, as well as noisy features. However, it does not guarantee to exclude overlapping features, since

the weight mass can be distributed among overlapping features. ℓ_1 regularization also does not fulfill a necessary characteristic for the present task, in that it does not provide a ranking based on the discriminative power of features.

2.2 Grafting

Grafting (Perkins et al., 2003) adds incremental feature selection during the training of a maximum entropy model. The selection process is a repetition of two steps: 1. a gradient-based heuristic selects the most promising feature from the set of unselected features Z, adding it to the set of selected features S, and 2. a full optimization of weights is performed over all features in S. These steps are repeated until a stopping condition is triggered.

During the first step, the gradient of each unselected feature $f_i \in Z$ is calculated with respect to the model p_S, that was trained with the set of selected features, S:

$$\left|\frac{\partial L(w_S)}{\partial w_i}\right| = p_S(f_i) - \tilde{p}(f_i) \tag{6}$$

The feature with the largest gradient is removed from Z and added to S.

The stopping condition for grafting integrates the ℓ_1 regularizer in the grafting method. Note that when ℓ_1 regularization is applied, a feature is only included (has a non-zero weight) if its penalty is outweighted by its contribution to the reduction of the objective function. Consequently, only features for which $\left|\frac{\partial L(w_S)}{\partial w_i}\right| > \lambda$ holds are eligible for selection. This is enforced by stopping selection if for all f_i in Z

$$\left|\frac{\partial L(w_S)}{\partial w_i}\right| \leq \lambda \tag{7}$$

Although grafting uses ℓ_1 regularization, its iterative nature avoids selecting overlapping features. For instance, if f_1 and f_2 are identical, and f_1 is added to the model p_S, $\left|\frac{\partial L(w_S)}{\partial w_2}\right|$ will amount to zero.

Performing a full optimization after each selected feature is computationally expensive. Riezler and Vasserman (2004) observe that during the feature step selection a larger number of features can be added to the model (n-best selection) without a loss of accuracy in the resulting model. However, this so-called *n-best grafting* may introduce overlapping features.

2.3 Grafting-light

The grafting-light method (Zhu et al., 2010) operates using the same selection step as grafting, but improves performance over grafting by applying one iteration of gradient-descent during the optimization step rather than performing a full gradient-descent. As such, grafting-light gradually works towards the optimal weights, while grafting always finds the optimal weights for the features in S during each iteration.

Since grafting-light does not perform a full gradient-descent, an additional stopping condition is required, since the model may still not be optimal even though no more features can be selected. This additional condition requires that change in value of the objective function incurred by the last gradient-descent is smaller than a predefined threshold.

2.4 Gain-informed selection

Gain-informed feature selection methods calculate the gain $\Delta L(S, f_i)$ of adding a feature $f_i \in Z$ to the model. If $L(w_S)$ is the negative log-likelihood of p_S, $\Delta L(S, f_i)$ is defined as:

$$\Delta L(S, f_i) \equiv L(w_S) - L(w_{S \cup f_i}) \tag{8}$$

During each selection step, the feature that gives the highest gain is selected. The calculation of $L(p_{S \cup f_i})$ requires a full optimization over the weights of the features in $S \cup f_i$. Since it is computationally intractable to do this for every f_i in Z, Berger et al. (1996) propose to estimate the weight w_i of the candidate feature f_i, while assuming that the weights of features in S stay constant. Under this assumption, w_i can be estimated using a simple line search method.

However, Zhou et al. (2003) observe that, despite this simplification, the gain-informed selection method proposed by Berger et al. (1996) still recalculates the weights of all the candidate features during every cycle. They observe that the gains of candidate features rarely increase. If it is assumed that the gain of adding a feature does indeed never increase as a result of adding another feature, the gains of features during the previous iteration can be kept.

To account for features that become ineffective, the gain of the highest ranked feature is recalculated. The highest ranked feature is selected if it remains the best feature after this recalculation. Otherwise, the same procedure is repeated for the next best feature.

De Kok (2010) modifies the method of Zhou et al. (2003) for ranking tasks. In the present work, we also apply this method, but perform a full optimization of feature weights in p_S every n cycles.

Since this selection method uses the gain of a feature in its selection criterion, it excludes noisy and redundant features. Overlapping features are also excluded since their gain diminishes after selecting one of the overlapping features.

3 Experimental setup and evaluation

3.1 Treebanks

We carry out our experiments using the Alpino dependency parser and generator for Dutch (van Noord, 2006; de Kok and van Noord, 2010). Two newspaper corpora are used in the experiments. The training data consists of the cdbl part of the Eindhoven corpus[1] (7,154 sentences). Syntactic annotations are available from the Alpino Treebank[2] (van der Beek et al., 2002). Part of the Trouw newspaper of 2001 is used for evaluation[3]. Syntactic annotations are part of LASSY[4] (van Noord et al., 2010), part WR-P-P-H (2,267 sentences).

3.2 Features

In our experiments, we use the features described in De Kok et al. (2011). In this section, we provide a short summarization of the types of features that are used.

Word adjacency. Word and Alpino part-of-speech tag trigram models are used as auxiliary distributions (Johnson and Riezler, 2000). In both models, linear interpolation smoothing is applied to handle unknown trigrams, and Laplacian smoothing for unknown unigrams. The trigram models have been trained on the Twente Nieuws Corpus (approximately 100 million words), excluding the Trouw 2001 corpus. In parsing, the value of the word trigram model is constant across derivations of a given input sentence.

Lexical frames. The parser applies lexical analysis to find all possible subcategorization frames for tokens in the input sentence. Since some frames occur more frequently in good parses than others, two feature templates record the use of frames in derivations. An additional feature implements an auxiliary distribution of frames, trained on a large corpus of automatically annotated sentences (436 million words). The values of lexical frame features are constant for all derivations in sentence realization, unless the frame is underspecified in the logical form.

Dependency relations. Several templates describe aspects of the dependency structure. For each dependency relation multiple dependency features are extracted. These features list the dependency relation, and characteristics of the head and dependent, such as their roots or part of speech tags. Additionally, features are used to implement auxiliary distributions for selectional preferences (van Noord, 2007). In generation, the values of these features are constant across derivations corresponding to a given logical form.

Syntactic features. Syntactic features include features that record the application of grammar rules, as well as the application of a rule in the context of another rule. Additionally, there are features describing more complex syntactic patterns, such as fronting of subjects and other noun phrases, orderings in the middle field, long-distance dependencies, and parallelism of conjuncts in coordinations.

3.3 Parse disambiguation

To create training and evaluation data for parse disambiguation, the treebanks described in section 3.1 are parsed, extracting the first 3000 derivations. On average, there are about 649 derivations for the sentences in the training data, and 402 derivations for the sentences in the test data.

Since the parser does not always yield the correct parse, the concept accuracy (CA) (van Noord,

[1] http://www.inl.nl/corpora/eindhoven-corpus

[2] http://www.let.rug.nl/vannoord/trees/

[3] http://hmi.ewi.utwente.nl/TwNC

[4] http://www.inl.nl/corpora/lassy-corpus

2006) of each derivation is calculated to estimate its quality. The highest scoring derivations for each input are marked as correct, all other derivations are marked as incorrect. Features are then extracted from each derivation.

The concept accuracy is calculated based on the named dependency relations of the candidate and correct parses. If $D_p(t)$ is the bag of dependencies produced by the parser for sentence t and $D_g(t)$ is the bag of dependencies of the correct (gold-standard) parse, concept accuracy is defined as:

$$\text{CA} = \frac{\sum_{t \in T} |D_p(t) \cap D_g(t)|}{\sum_{t \in T} \max(|D_p(t)|, |D_g(t)|)} \quad (9)$$

The procedure outlined above gives examples of correct and incorrect derivations to train the model, and derivations to test the resulting model.

3.4 Fluency ranking

For training and evaluation of the fluency ranker, we use the same treebanks as in parse disambiguation. We assume that the sentence that corresponds to a dependency structure in the treebank is the correct realization of that dependency structure. We parse each sentence in the treebank, extracting the dependency structure that is the most similar to that in the treebank. We perform this step to assure that it is possible to generate from the given dependency structure. We then use the Alpino chart generator to make all possible derivations and realizations conforming to that dependency structure. Due to a limit on generation time, some longer sentences and corresponding dependency structures are excluded from the data. The average sentence length was 15.7 tokens, with a maximum of 26 tokens.

Since the sentence in the treebank cannot always be produced exactly, we estimate the quality of each realization using the General Text Matcher (GTM) method (Melamed et al., 2003). The best-scoring derivations are marked as correct, the other derivations are marked as incorrect. Finally, features are extracted from these derivations.

The General Text Matcher method marks all corresponding tokens of a candidate realization and the correct realization in a grid, and finds the maximum matching (the largest subset of marks, such that no marks are in the same row or column). The size of the matching M is then determined using the lengths of runs r in the matching (a run is a diagonal of marks), rewarding longer runs:

$$size(M) = \sqrt{\sum_{r \in M} length(r)^2} \quad (10)$$

This method has been shown to have the highest correlation with human judgments in a related language (German), using a comparable system (Cahill, 2009).

3.5 Training

Models are trained by extracting an informative sample of $\Omega(c)$ for each c in the training data (Osborne, 2000). This informative sample consists of at most 100 randomly selected derivations.

We then apply feature selection on the training data. We let each method select 1711 features. This number is derived from the number of non-zero features that training a model with a ℓ_1 norm coefficient of 0.0002 gives. Grafting and grafting-light selection are applied using TinyEst[5]. For gain-informed selection, we use FeatureSqueeze[6]. For all three methods, we add 10 features to the model during each selection step.

3.6 Evaluation

We evaluate each selection method stepwise. We train and evaluate a model on the best-n features according to each selection method, for $n = [0..1711]$. In each case, the feature weights are estimated with TinyEst using a ℓ_1 norm coefficient of 0.0002. This stepwise evaluation allows us to capture the effectiveness of each method.

Parse disambiguation and fluency ranking models are evaluated on the WR-P-P-H corpus that was described in Section 3.1, using CA and GTM scores respectively.

4 Evaluation of feature selection methods

4.1 Incremental feature selection

Figure 1 shows the performance of the feature selection methods for parse disambiguation. This graph shows that that both grafting methods are far more

[5] http://github.com/danieldk/tinyest

[6] https://github.com/rug-compling/featuresqueeze

effective than gain-informed selection. We can also see that only a small number of features is required to construct a competitive model. Selecting more features improves the model only gradually.

Figure 2 shows the performance of the feature selection methods in fluency ranking. Again, we see the same trend as in parse disambiguation. The grafting and grafting-light methods outperform gain-informed selection, with the grafting method coming out on top. In feature selection, even a smaller number of features is required to train an effective model. After selecting more than approximately 50 features, adding features only improves the model very gradually.

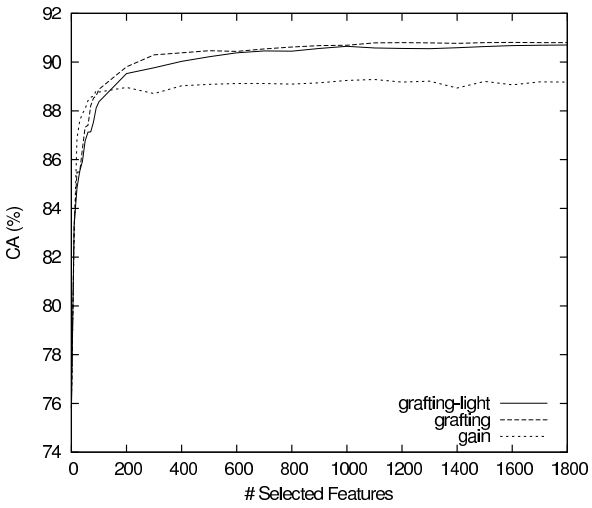

Figure 1: Application of feature selection methods to parse disambiguation

4.2 Selection using an ℓ_1 prior

During our experiments, we also evaluated the effect of using an ℓ_1 prior in Alpino to see if it is worthwile to replace feature selection using a frequency cut-off (Malouf and van Noord, 2004). Using Alpino's default configuration with a frequency cut-off of 2 and an ℓ_2 prior with $\sigma^2 = 1000$ the system had a CA-score of 90.94% using 25237 features. We then trained a model, applying an ℓ_1 prior with a norm coefficient of 0.0002. With this model, the system had a CA-score of 90.90% using 2346 features.

In generation, Alpino uses a model with the same frequency cut-off and ℓ_2 prior. This model has 1734 features features and achieves a GTM score of 0.7187. Applying the ℓ_1 prior reduces the number

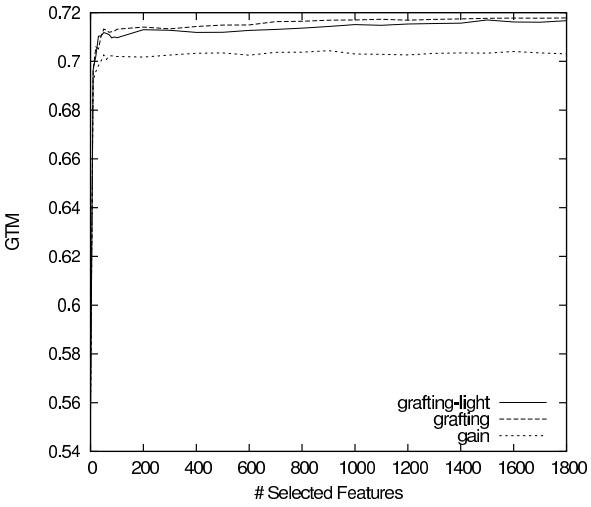

Figure 2: Effectiveness of feature selection methods in fluency ranking. Both grafting methods outperform gain-based ranking

of features to 607, while mildly increasing the GTM score to 0.7188.

These experiments show that the use of ℓ_1 priors can compress models enormously, even compared to frequency-based feature selection, while retaining the same levels of accuracy.

5 Quantitative analysis of reversible models

For a quantitative analysis of highly discriminative features, we extract the 300 most effective features of the fluency ranking, parse disambiguation, and reversible models using grafting. We then divide features into five classes: *dependency* (enumeration of dependency triples), *lexical* (readings of words), *n-gram* (word and tag trigram auxiliary distributions), *rule* (identifiers of grammar rules), and *syntactic* (abstract syntactic features). Of these classes, *rule* and *syntactic* features are active during both parse disambiguation and fluency ranking.

In the quantitative analyses, we train a model for each selection step. The models contain the 1 to 300 best features. Using these models, we can calculate the contribution of feature f_i to the improvement according to some evaluation function e

$$c(f_i) = \frac{e(p_{0..i}) - e(p_{0..i-1})}{e(p_{0..n}) - e(p_0)} \qquad (11)$$

where $p_{0..i}$ is a model trained with the i most dis-

criminative features, p_0 is the uniform model, and $n = 300$.

5.1 Parse disambiguation

Table 1 provides class-based counts of the 300 most discriminative features for the parse disambiguation and reversible models. Since the n-gram features are not active during parse disambiguation, they are not selected for the parse disambiguation model. All other classes of features are used in the parse disambiguation model. The reversible model uses all classes of features.

Class	Directional	Reversible
Dependency	93	84
Lexical	24	24
N-gram	0	2
Rule	156	154
Syntactic	27	36

Table 1: Per-class counts of the best 300 features according to the grafting method.

Contributions per feature class in parse disambiguation are shown in table 2. In the directional parse disambiguation model, parsing-specific features (*dependency* and *lexical*) account for 55% of the improvement over the uniform model.

In the reversible model, there is a shift of contribution towards task-independent features. When applying this model, the contribution of parsing-specific features to the improvement over the uniform model is reduced to 45.79%.

We can conclude from the per-class feature contributions in the directional parse disambiguation model and the reversible model, that the reversible model does not put more emphasis on parsing-specific features. Instead, the opposite is true: task-independent features are more important in the reversible model than the directional model.

5.2 Fluency ranking

Table 3 provides class-based counts of the 300 most discriminative features of the fluency ranking and reversible models. During fluency ranking, dependency features and lexical features are not active.

Table 4 shows the per-class contribution to the improvement in accuracy for the directional and reversible models. Since the dependency and lexical features are not active during fluency ranking, it may come as a surprise that their contribution is negative in the reversible model. Since they are used for parse disambiguation, they have an effect on weights of task-independent features. This phenomenon did not occur when using the reversible model for parse disambiguation, because the features specific to fluency ranking (n-gram features) were selected as the most discriminative features in the reversible model. Consequently, the reversible models with one and two features were uniform models from the perspective of parse disambiguation.

Class	Directional	Reversible
Dependency	21.53	13.35
Lexical	33.68	32.62
N-gram	0.00	0.00
Rule	37.61	47.35
Syntactic	7.04	6.26

Table 2: Per-class contribution to the improvement of the model over the base baseline in parse disambiguation.

Class	Directional	Reversible
Dependency	0	84
Lexical	0	24
N-gram	2	2
Rule	181	154
Syntactic	117	36

Table 3: Per-class counts of the best 300 features according to the grafting method.

Class	Directional	Reversible
Dependency	0.00	-4.21
Lexical	0.00	-1.49
N-gram	81.39	83.41
Rule	14.15	16.45
Syntactic	3.66	4.59

Table 4: Per-class contribution to the improvement of the model over the baseline in fluency ranking.

Since active features compensate for this loss in the reversible model, we cannot directly compare per-class contributions. To this end, we normalize the contribution of all positively contributing features, leading to table 5. Here, we can see that the reversible model does not shift more weight towards task-specific features. On the contrary, there is a

mild effect in the opposite direction here as well.

Class	Directional	Reversible
N-gram	81.39	79.89
Rule	14.15	15.75
Syntactic	3.66	4.39

Table 5: Classes giving a net positive distribution, with normalized contributions.

6 Qualitative analysis of reversible models

While the quantitative evaluation shows that task-independent features remain important in reversible models, we also want to get an insight into the actual features that were used. Since it is unfeasible to study the 300 best features in detail, we extract the 20 best features.

Grafting-10 is too course-grained for this task, since it selects the first 10 features solely by their gradients, while there may be overlap in those features. To get the most accurate list possible, we perform grafting-1 selection to extract the 20 most effective features. We show these features in table 6 with their polarities. The polarity indicates whether a feature is an indicator for a good (+) or bad (-) derivation.

We now provide a description of these features by category.

Word/tag trigrams. The most effective features in fluency ranking are the n-gram auxiliary distributions (1, 3). The word n-gram model settles preferences with respect to fixed expressions and common word orders. It also functions as a (probabilistic) filter of archaic inflections and incorrect inflections that are not known to the Alpino lexicon. The tag n-gram model help picking a sequence of part-of-speech tags that is plausible.

Frame selection. Various features assist in the selection of proper subcategorization frames for words. This currently affects parse disambiguation mostly. There is virtually no ambiguity of frames during generation, and a stem/frame combination normally only selects one inflection. The most effective feature for frame selection is (2), which is an auxiliary distribution of words and corresponding frames based on a large automatically annotated corpus. Other effective features indicate that readings as an adjective (9) and as an intransitive verb (20) are not preferred.

Rank	Polarity	Feature
1	+	ngram_lm
2	+	z_f2
3	+	ngram_tag
4	-	r1(np_n)
5	+	r2(np_det_n,2,n_n_pps)
6	-	p1(pardepth)
7	+	r2(vp_mod_v,3,vproj_vc)
8	-	r2(vp_arg_v(np),2,vproj_vc)
9	-	f1(adj)
10	+	r2(vp_mod_v,2,optpunct(e))
11	-	s1(non_subj_np_topic)
12	+	r1(n_adj_n)
13	+	dep23(prep,hd/pc,verb)
14	+	r1(optpunct(e))
15	+	dep34(van,prep,hd/mod,noun)
16	+	dep23(noun,hd/su,verb)
17	+	p1(par)
18	-	r1(vp_v_mod)
19	+	dep23(prep,hd/mod,verb)
20	-	f1(verb(intransitive))

Table 6: The twenty most discriminative features of the reversible model, and their polarities.

Modifiers. Feature 5 indicates the preference to attach prepositional phrases to noun phrases. However, if a modifier is attached to a verb, we prefer readings and realizations where the modifier is left-adjoining rather than right-adjoining (7, 18, 19). For instance, *zij heeft met de hond gelopen* (*she has with the dog walked*) is more fluent than *zij heeft gelopen met de hond* (*she has walked with the dog*). Finally, feature 15 gives preference to analyses where the preposition *van* is a modifier of a noun.

Conjunctions. Two of the twenty most discriminative features involve conjunctions. The first (6) is a dispreference for conjunctions where conjuncts have a varying depth. In conjunctions, the model prefers derivations where all conjuncts in a conjunctions have an equal depth. The other feature (17) gives a preferences to conjunctions with parallel conjuncts — conjunctions where every conjunct is constructed using the same grammar rule.

Punctuation. The Alpino grammar is very generous in allowing optional punctuation. An empty punctuation sign is used to fill grammar rule slots when no punctuation is used or realized. Two features indicate preferences with respect to optional punctuation. The first (10) gives preference to filling the second daughter slot of the *vp_mod_v* with the empty punctuation sign. This implies that derivations are preferred where a modifier and a verb are not separated by punctuation. The second feature (14) indicates a general preference for the occurrence of empty optional punctuation in the derivation tree.

Subjects/objects. In Dutch, subject fronting is preferred over object fronting. For instance, *Spanje won de wereldbeker* (*Spain won the World Cup*) is preferred over *de wereldbeker won Spanje* (*the World Cup won spain*). Feature 8 will in many cases contribute to the preference of having topicalized noun phrase subjects. It disprefers having a noun phrase left of the verb. For example, *zij heeft met de hond gelopen* (*she has with the dog walked*) is preferred over *met de hond heeft zij gelopen* (*with the dog she has walked*). Feature 11 encodes the preference for subject fronting, by penalizing derivations where the topic is a non-subject noun phrase.

Other syntactic preferences. The remaining features are syntactic preferences that do not belong to any of the previous categories. Feature 4 indicates a dispreference for derivations where bare nouns occur. Feature 12 indicates a preference for derivations where a noun occurs along with an adjective. Finally, feature 13 gives preference to the prepositional complement (*pc*) relation if a preposition is a dependent of a verb and lexical analysis shows that the verb can combine with that prepositional complement.

We can conclude from this description of features that many of the features that are paramount to parse disambiguation and fluency ranking are task-independent, modeling phenomena such as subject/object fronting, modifier adjoining, parallelism and depth in conjunctions, and the use of punctuation.

7 Conclusion

In this work we have used feature selection techniques for maximum entropy modeling to analyze the hypothesis that the models in reversible stochastic attribute-value grammars use task-independent features. To this end, we have first compared three feature selection techniques, namely gain-informed selection, grafting, and grafting-light. In this comparison we see that grafting outperforms both grafting-light and gain-informed selection in parse disambiguation and fluency ranking tasks.

We then used grafting to select the most effective features for parse disambiguation, fluency ranking, and reversible models. In the quantitative analysis we have shown that the reversible model does not put more emphasis on task-specific features. In fact, the opposite is true: in the reversible model task-independent features become more defining than in the directional models.

We have also provided a qualitative analysis of the twenty most effective features, showing that many of these features are relevant to both parsing and generation. Effective task-independent features for Dutch model phenomena such as subject/object fronting, modifier adjoining, parallelism and depth in conjunctions, and the use of punctuation.

8 Future work

An approach for testing the reversibility of models that we have not touched upon in this work, is to evaluate such models using tasks that combine parsing and generation. For instance, a good word graph parser should choose a fluent sentence with a syntactically plausible reading. If reversible models integrate parsing-specific, generation-specific, and task-independent features properly, they should be competitive to models specifically trained for that task. In the future, we hope to evaluate reversible stochastic attribute-value grammars in the light of such tasks.

9 Acknowledgments

This work was funded by the DAISY project of the STEVIN program. The author would also like to thank Yan Zhao, Barbara Plank, and Gertjan van Noord for the many valuable discussions on maximum entropy modeling and feature selection.

References

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):71.

Aoife Cahill, Martin Forst, and Christian Rohrer. 2007. Designing features for parse disambiguation and realisation ranking. In *The Proceedings of the LFG '07 Conference*, pages 128–147. CSLI Publications.

Aoife Cahill. 2009. Correlating human and automatic evaluation of a german surface realiser. In *Proceedings of the ACL-IJCNLP 2009 Conference - Short Papers*, pages 97–100.

Daniël de Kok and Gertjan van Noord. 2010. A sentence generator for Dutch. In *Proceedings of the 20th Computational Linguistics in the Netherlands conference (CLIN)*, pages 75–90.

Daniël de Kok, Barbara Plank, and Gertjan van Noord. 2011. Reversible stochastic attribute-value grammars. In *Proceedings of the ACL HLT 2011 Conference - Short Papers*.

Daniël de Kok. 2010. Feature selection for fluency ranking. In *Proceedings of the 6th International Natural Language Generation Conference (INLG)*, pages 155–163.

Mark Johnson and Stefan Riezler. 2000. Exploiting auxiliary distributions in stochastic unification-based grammars. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 154–161, Seattle, Washington.

Robert Malouf and Gertjan van Noord. 2004. Wide coverage parsing with stochastic attribute value grammars. In *IJCNLP-04 Workshop: Beyond shallow analyses - Formalisms and statistical modeling for deep analyses*. JST CREST, March.

I. Dan Melamed, Ryan Green, and Joseph Turian. 2003. Precision and recall of machine translation. In *HLT-NAACL*.

Miles Osborne. 2000. Estimation of stochastic attribute-value grammars using an informative sample. In *Proceedings of the 18th conference on Computational linguistics (COLING)*, pages 586–592.

Simon Perkins, Kevin Lacker, and James Theiler. 2003. Grafting: Fast, incremental feature selection by gradient descent in function space. *The Journal of Machine Learning Research*, 3:1333–1356.

Stefan Riezler and Alexander Vasserman. 2004. Incremental feature selection and l1 regularization for relaxed maximum-entropy modeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP04), Barcelona, Spain*.

Robert Tibshirani. 1996. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Leonoor van der Beek, Gosse Bouma, Robert Malouf, and Gertjan van Noord. 2002. The Alpino dependency treebank. In *Computational Linguistics in the Netherlands (CLIN)*.

Gertjan van Noord, Ineke Schuurman, and Gosse Bouma. 2010. Lassy syntactische annotatie, revision 19053. `http://www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf`.

Gertjan van Noord. 2006. **A**t **L**ast **P**arsing **I**s **N**ow **O**perational. In *TALN 2006 Verbum Ex Machina, Actes De La 13e Conference sur Le Traitement Automatique des Langues naturelles*, pages 20–42, Leuven.

Gertjan van Noord. 2007. Using self-trained bilexical preferences to improve disambiguation accuracy. In *Proceedings of the International Workshop on Parsing Technology (IWPT)*, ACL 2007 Workshop, pages 1–10, Prague. Association for Computational Linguistics, ACL.

Yaqian Zhou, Lide Wu, Fuliang Weng, and Hauke Schmidt. 2003. A fast algorithm for feature selection in conditional maximum entropy modeling. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, EMNLP '03, pages 153–159, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jun Zhu, Ni Lao, and Eric P. Xing. 2010. Grafting-light: fast, incremental feature selection and structure learning of Markov random fields. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 303–312. ACM.