Learning Preferences for Referring Expression Generation: Effects of Domain, Language and Algorithm

Ruud Koolen
Tilburg University
P.O. Box 90135
5000 LE Tilburg
The Netherlands
r.m.f.koolen@uvt.nl

Emiel Krahmer
Tilburg University
P.O. Box 90135
5000 LE Tilburg
The Netherlands
e.j.krahmer@uvt.nl

Mariët Theune
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands
m.theune@utwente.nl

Abstract

One important subtask of Referring Expression Generation (REG) algorithms is to select the attributes in a definite description for a given object. In this paper, we study how much training data is required for algorithms to do this properly. We compare two REG algorithms in terms of their performance: the classic Incremental Algorithm and the more recent Graph algorithm. Both rely on a notion of preferred attributes that can be learned from human descriptions. In our experiments, preferences are learned from training sets that vary in size, in two domains and languages. The results show that depending on the algorithm and the complexity of the domain, training on a handful of descriptions can already lead to a performance that is not significantly different from training on a much larger data set.

1 Introduction

Most practical NLG systems include a dedicated module for Referring Expression Generation (REG) in one form or another (Mellish et al., 2006). One central problem a REG module needs to address is deciding on the contents of a description. Jordan and Walker (2005), for example, studied human-produced descriptions in a furniture scenario, and found that speakers can refer to a target in many different ways ("the yellow rug", "the $150 rug", etc.). The question, then, is how speakers decide which attributes to include in a description, and how this decision process can be modeled in a REG algorithm.

When we focus on the generation of distinguishing descriptions (which is often done in REG), it is usually assumed that some attributes are more preferred than others: when trying to identify a chair, for example, its colour is probably more helpful than its size. It is precisely this intuition of preferred attributes which is incorporated in the Incremental Algorithm (Dale and Reiter, 1995), arguably one of the most influential REG algorithms to date. The Incremental Algorithm (IA) assumes the existence of a complete, ordered list of preferred attributes. The algorithm basically iterates through this list, adding an attribute (e.g., COLOUR) to the description under construction if its value (e.g., *yellow*) helps ruling out one or more of the remaining distractors.

Even though the IA is exceptional in that it relies on a complete ordering of attributes, most current REG algorithms make use of preferences in some way (Fabbrizio et al., 2008; Gervás et al., 2008; Kelleher, 2007; Spanger et al., 2008; Viethen and Dale, 2010). The graph-based REG algorithm (Krahmer et al., 2003), for example, models preferences in terms of costs, where cheaper is more preferred. Contrary to the IA, the graph-based algorithm assumes that preferences operate at the level of attribute-value pairs (or properties) rather than at the level of attributes; in this way it becomes possible to prefer a straightforward size (*large*) over a subtle colour (*mauve, taupe*). Moreover, the graph-based algorithm looks for the cheapest overall description, and may opt for a description with a single, relatively dispreferred property ("the man with the blue eyes") when the alternative would be to combine many, relatively preferred properties ("the large, balding man with the bow tie and the striped tuxedo"). This flexibility is arguably one of the

reasons why the graph-based REG approach works well: it was the best performing system in the most recent REG Challenge (Gatt et al., 2009).

But where do the preferences used in the algorithms come from? Dale and Reiter point out that preferences are domain dependent, and that determining them for a given domain is essentially an empirical question. Unfortunately, they do not specify how this particular empirical question should be answered. The general preference for colour over size is experimentally well-established (Pechmann, 1989), but for most other cases experimental data are not readily available. An alternative would be to look at human data, preferably in a "semantically transparent" corpus (van Deemter et al., 2006), that is: a corpus that contains the attributes and values of all domain objects, together with the attribute-value pairs actually included in a target reference. Such corpora are typically collected using human participants, who are asked to produce referring expressions for targets in controlled visual scenes. One example is the TUNA corpus, which is a publicly available data set containing 2280 human-produced descriptions in total, and which formed the basis of various REG Challenges. Clearly, building a corpus such as TUNA is a time consuming and labour intensive exercise, so it will not be surprising that only a handful of such corpora exists (and often only for English).

This raises an important question: how many human-produced references are needed to make a good estimate of which attributes and properties are preferred? Do we really need hundreds of instances, or is it conceivable that a few of them (collected in a semantically transparent way) will do? This is not an easy matter, since various factors might play a role: from which data set are example references sampled, what are the domains of interest, and, perhaps most importantly, which REG algorithm is considered? In this paper, we address these questions by systematically training two REG algorithms (the Incremental Algorithm and the graph-based REG algorithm) on sets of human-produced descriptions of increasing size and evaluating them on a held-out test set; we do this for two different domains (people and furniture descriptions) and two data sets in two different languages (TUNA and D-TUNA, the Dutch version of TUNA).

That size of the training set may have an impact on the performance of a REG algorithm was already suggested by Theune et al. (2011), who used the English TUNA corpus to determine preferences (costs) for the graph-based algorithm using a similar learning curve set-up as we use here. However, the current paper expands on Theune et al. (2011) in three major ways. Firstly, most importantly, where Theune et al. reported results for only one algorithm (the graph-based one), we directly compare the performance of the graph-based algorithm and the Incremental Algorithm (something which, somewhat surprisingly, has not been done before). Secondly, we test whether these algorithms perform differently in two different languages (English and Dutch), and thirdly, we use eight training set sizes, which is more than the six set sizes that were used by Theune et al.

Below we first explain in more detail which algorithms (Section 2) and corpora (Section 3) we used for our experiments. Then we describe how we derived costs and orders from subsets of these corpora (Section 4), and report the results of our experiments focusing on effects of domain, language and size of the training set (Section 5). We end with a discussion and conclusion (Section 6), where we also compare the performance of the IA trained on small set sizes with that of the classical Full Brevity and Greedy algorithms (Dale and Reiter, 1995).

2 The Algorithms

In this section we briefly describe the two algorithms, and their settings, used in our experiment. For details about these algorithms we refer to the original publications.

The Incremental Algorithm (IA) The basic assumption underlying the Incremental Algorithm (Dale and Reiter, 1995) is that speakers "prefer" certain attributes over others when referring to objects. This intuition is formalized in the notion of a list of attributes, ranked in order of preference. When generating a description for a target, the algorithm iterates through this list, adding an attribute to the description under construction if its value helps rule out any of the distractors not previously ruled out. There is no backtracking in the IA, which means that a selected attribute is always realized in

the final description, even if the inclusion of later attributes renders it redundant. In this way, the IA is capable of generating overspecified descriptions, in accordance with the human tendency to mention redundant information (Pechmann, 1989; Engelhardt et al., 2006; Arts et al., 2011). The TYPE attribute (typically realized as the head noun) has a special status in the IA. After running the algorithm it is checked whether TYPE is in the description; if not, it is added, so that TYPE is always included even if it does not rule out any distractors.

To derive preference orders from human-produced descriptions we proceeded as follows: given a set of n descriptions sampled from a larger corpus (where n is the set size, a variable we systematically control in our experiment), we counted the number of times a certain attribute occurred in the n descriptions. The most frequently occurring attribute was placed at the first position of the preferred attributes list, followed by the second most frequent attribute, etc. In the case of a tie (i.e., when two attributes occurred equally often, which typically is more likely to happen in small training sets), the attributes were ordered alphabetically. In this way, we made sure that all ties were treated in the same, comparable manner, which resulted in a complete ranking of attributes, as required by the IA.

The Graph-based Algorithm (Graph) In the graph-based algorithm (Krahmer et al., 2003), which we refer to as Graph, information about domain objects is represented as a labelled directed graph, and REG is modeled as a graph-search problem. The output of the algorithm is the cheapest distinguishing subgraph, given a particular *cost function* assigning costs to properties (i.e., attribute-value pairs). By assigning zero costs to some properties Graph is also capable of generating overspecified descriptions, including redundant properties. To ensure that the graph search does not terminate before the free properties are added, the search order must be explicitly controlled (Viethen et al., 2008). To ensure a fair comparison with the IA, we make sure that if the target's TYPE property was not originally selected by the algorithm, it is added afterwards.

In this study, both the costs and orders required by Graph are derived from corpus data. We base the property order on the frequency with which each attribute-value pair is mentioned in a training corpus, relative to the number of target objects with this property. The properties are then listed in order of decreasing frequency. Costs can be derived from the same corpus frequencies; here, following Theune et al. (2011), we adopt a systematic way of deriving costs from frequencies based on k-means clustering. Theune and colleagues achieved the best performance with $k = 2$, meaning that the properties are divided in two groups based on their frequency. The properties in the group with the highest frequency get cost 0. These 'free' properties are always included in the description if they help distinguish the target. The properties in the less frequent group get cost 1; of these properties, the algorithm only adds the minimum number necessary to achieve a distinguishing description. Ties due to properties occurring with the same frequency need not be resolved when determining the cost function, since Graph does not assume the existence of a complete ordering. Properties that did not occur in a training corpus were automatically assigned cost 1. Like we did for the IA, we listed attribute-value pairs with the same frequency in alphabetical order.

3 Corpora

Training and test data for our experiment were taken from two corpora of referring expressions, one English (TUNA) and one Dutch (D-TUNA).

TUNA The TUNA corpus (Gatt et al., 2007) is a semantically transparent corpus consisting of object descriptions in two domains (furniture and people). The corpus was collected in an on-line production experiment, in which participants were presented with visual scenes containing one target object and six distractor objects. These objects were ordered in a 5 $\times$ 3 grid, and the participants were asked to describe the target in such a way that it could be uniquely distinguished from its distractors. Table 1 shows the attributes and values that were annotated for the descriptions in the two domains.

There were two experimental conditions: in the +LOC condition, the participants were free to describe the target using any of its properties, including its location on the screen (represented

Furniture	
Attribute	**Possible values**
TYPE	*chair, desk, sofa, fan*
COLOUR	*green, red, blue, gray*
ORIENTATION	*front, back, left, right*
SIZE	*large, small*
X-DIMENSION	*1, 2, 3, 4, 5*
Y-DIMENSION	*1, 2, 3*
People	
Attribute	**Possible values**
TYPE	*person*
AGE	*old, young*
HAIRCOLOUR	*light, dark*
ORIENTATION	*front, left, right*
HASBEARD	*true, false*
HASGLASSES	*true, false*
HASSHIRT	*true, false*
HASSUIT	*true, false*
HASTIE	*true, false*
X-DIMENSION	*1, 2, 3, 4, 5*
Y-DIMENSION	*1, 2, 3*

Table 1: Attributes and values in the furniture and people domains. X- and Y-DIMENSION refer to an object's horizontal and vertical position in a scene grid and only occur in the English TUNA corpus.

in Table 1 as the X- and Y-DIMENSION), whereas in the -LOC condition they were discouraged (but not prevented) from mentioning object locations. However, some descriptions in the -LOC condition contained location information anyway.

D-TUNA For Dutch, we used the D-TUNA corpus (Koolen and Krahmer, 2010). This corpus uses the same visual scenes and annotation scheme as the TUNA corpus, but consists of Dutch instead of English target descriptions. Since the D-TUNA experiment was performed in laboratory conditions, its data is relatively 'cleaner' than the TUNA data, which means that it contains fewer descriptions that are not fully distinguishing and that its descriptions do not contain X- and Y-DIMENSION attributes. Although the descriptions in D-TUNA were collected in three different conditions (written, spoken, and face-to-face), we only use the written descriptions in this paper, as this condition is most similar to the data collection in TUNA.

4 Method

To find out how much training data is required to achieve an acceptable attribute selection performance for the IA and Graph, we derived orders and costs from different sized training sets. We then evaluated the algorithms, using the derived orders and costs, on a test set. Training and test sets were taken from TUNA and D-TUNA.

As Dutch training data, we used 160 furniture and 160 people items, randomly selected from the textual descriptions in the D-TUNA corpus. The remaining furniture and people descriptions (40 items each) were used for testing. As English training data, we took all -LOC data from the training set of the REG Challenge 2009 (Gatt et al., 2009): 165 furniture and 136 people descriptions. As English test data we used all -LOC data from the REG 2009 development set: 38 furniture and 38 people descriptions. We only used -LOC data to increase comparability to the Dutch data.

From the Dutch and English furniture and people training data, we selected random subsets of 1, 5, 10, 20, 30, 40 and 50 descriptions. Five different sets of each size were created, since the accidental composition of a training set could strongly influence the results. All training sets were built up in a cumulative fashion, starting with five randomly selected sets of size 1, then adding 4 items to each of them to create five sets of size 5, and so on, for each combination of language and domain. We used these different training sets to derive preference orders of attributes for the IA, and costs and property orders for Graph, as outlined above.

We evaluated the performance of the derived preference orders and cost functions on the test data for the corresponding domain and language, using the standard Dice and Accuracy metrics for evaluation. Dice measures the overlap between attribute sets, producing a value between 1 and 0, where 1 stands for a perfect match and 0 for no overlap at all. Accuracy is the percentage of perfect matches between the generated attribute sets and the human descriptions in the test set. Both metrics were used in the REG Generation Challenges.

	English furniture			
	IA		Graph	
Set size	Dice	Acc.(%)	Dice	Acc.(%)
1	0.764	36.8	0.693	24.7
5	0.829	55.3	0.756	33.7
10	0.829	55.3	0.777	39.5
20	0.829	55.3	0.788	40.5
30	0.829	55.3	0.782	40.5
40	0.829	55.3	0.793	45.3
50	0.829	55.3	0.797	45.8
All	0.829	55.3	0.810	50.0
	Dutch furniture			
	IA		Graph	
Set size	Dice	Acc.(%)	Dice	Acc.(%)
1	0.925	63.0	0.876	44.5
5	0.935	67.5	0.917	62.0
10	0.929	68.5	0.923	66.0
20	0.930	65.5	0.923	64.0
30	0.931	67.0	0.924	65.5
40	0.931	67.0	0.931	67.5
50	0.929	66.0	0.929	67.0
All	0.926	65.0	0.929	67.5

Table 2: Performance for each set size in the furniture domain. For sizes 1 to 50, means over five sets are given. The full sets are 165 English and 160 Dutch descriptions. Note that the scores of the IA for the English sets of sizes 1 to 30 were also reported in Theune et al. (2011).

	English people			
	IA		Graph	
Set size	Dice	Acc.(%)	Dice	Acc.(%)
1	0.519	7.4	0.558	12.6
5	0.605	15.8	0.617	14.5
10	0.682	21.1	0.683	20.0
20	0.710	22.1	0.716	24.7
30	0.682	15.3	0.716	26.8
40	0.716	26.3	0.723	26.3
50	0.718	27.9	0.727	26.3
All	0.724	31.6	0.730	28.9
	Dutch people			
	IA		Graph	
Set size	Dice	Acc.(%)	Dice	Acc.(%)
1	0.626	4.5	0.682	17.5
5	0.737	16.0	0.738	21.0
10	0.738	12.5	0.741	19.5
20	0.765	12.5	0.778	25.5
30	0.762	14.5	0.789	25.0
40	0.763	11.5	0.792	25.0
50	0.764	10.5	0.798	26.0
All	0.775	12.5	0.812	32.5

Table 3: Performance for each set size in the people domain. For sizes 1 to 50, means over five sets are given. The full sets are 136 English and 160 Dutch descriptions. Note that the scores of the IA for the English sets of sizes 1 to 30 were also reported in Theune et al. (2011).

5 Results

5.1 Overall analysis

To determine the effect of domain and language on the performance of REG algorithms, we applied repeated measures analyses of variance (ANOVA) to the Dice and Accuracy scores, using *set size* (1, 5, 10, 20, 30, 40, 50, all) and *domain* (furniture, people) as within variables, and *algorithm* (IA, Graph) and *language* (English, Dutch) as between variables.

The results show main effects of *domain* (Dice: $F_{(1,152)} = 56.10$, $p < .001$; Acc.: $F_{(1,152)} = 76.36$, $p < .001$) and *language* (Dice: $F_{(1,152)} = 30.30$, $p < .001$; Acc.: $F_{(1,152)} = 3.380$, $p = .07$). Regarding the two domains, these results indicate that both the IA and the Graph algorithm generally performed better in the furniture domain (Dice: M = .86, SD = .01; Acc.: M = .56, SD = .03) than in the people domain (Dice: M = .72, SD = .01; Acc.: M = .20, SD = .02). Regarding the two languages, the results show that both algorithms generally performed better on the Dutch data (Dice: M = .84, SD = .01; Acc.: M = .41, SD = .03) than on the English data (Dice: M = .74, SD = .01; Acc.: M = .34, SD = .03). There is no main effect of *algorithm*, meaning that overall, the two algorithms had an equal performance. However, this is different when we look separately at each domain and language, as we do below.

5.2 Learning curves per domain and language

Given the main effects of domain and language described above, we ran separate ANOVAs for the different domains and languages. For these four analyses, we used *set size* as a within variable, and *algorithm* as a between variable. To determine the effects of *set size*, we calculated the means of the scores of the five training sets for each set size, so that we could compare them with the scores of the entire set. The results are shown in Tables 2 and 3.

We made planned post hoc comparisons to test which is the smallest set size that does not perform significantly different from the entire training set in

terms of Dice and Accuracy scores (we call this the "ceiling"). We report results both for the standard Bonferroni method, which corrects for multiple comparisons, and for the less strict LSD method from Fisher, which does not. Note that with the Bonferroni method we are inherently less likely to find statistically significant differences between the set sizes, which implies that we can expect to reach a ceiling earlier than with the LSD method. Table 4 shows the ceilings we found for the algorithms, per domain and language.

	English furniture		**Dutch furniture**	
	Dice	Accuracy	Dice	Accuracy
IA	5 (5)	5 (5)	1 (1)	1 (1)
Graph	10 (5)	40 (5)	5 (1)	5 (1)
	English people		**Dutch people**	
	Dice	Accuracy	Dice	Accuracy
IA	10 (10)	40 (1)	20 (5)	1 (1)
Graph	20 (10)	20 (1)	30 (20)	5 (1)

Table 4: Ceiling set sizes computed using LSD, with Bonferroni between brackets.

The furniture domain Table 2 shows the Dice and Accuracy scores in the furniture domain. We found significant effects of *set size* for both the English data (Dice: $F_{(7,518)} = 15.59$, $p < .001$; Acc.: $F_{(7,518)} = 17.42$, $p < .001$) and the Dutch data (Dice: $F_{(7,546)} = 5.322$, $p < .001$; Acc.: $F_{(7,546)} = 5.872$, $p < .001$), indicating that for both languages, the number of descriptions used for training influenced the performance of both algorithms in terms of both Dice and Accuracy. Although we did not find a main effect of algorithm, suggesting that the two algorithms performed equally well, we did find several interactions between *set size* and *algorithm* for both the English data (Dice: $F_{(7,518)} = 1.604$, ns; Acc.: $F_{(7,518)} = 2.282$, $p < .05$) and the Dutch data (Dice: $F_{(7,546)} = 3.970$, $p < .001$; Acc.: $F_{(7,546)} = 3.225$, $p < .01$). For the English furniture data, this interaction implies that small set sizes have a bigger impact for the IA than for Graph. For example, moving from set size 1 to 5 yielded a Dice improvement of .18 for the IA, while this was only .09 for Graph. For the Dutch furniture data, however, a reverse pattern was observed; moving from set size 1 to 5 yielded an improvement of .01 (Dice) and .05 (Acc.) for the IA, while this was .11 (Dice) and .18 (Acc.) for Graph.

Post hoc tests showed that small set sizes were generally sufficient to reach ceiling performance: the general pattern for both algorithms and both languages was that the scores increased with the size of the training set, but that the increase got smaller as the set sizes became larger. For the English furniture data, Graph reached the ceiling at set size 10 for Dice (5 with the Bonferroni test), and at set size 40 for Accuracy (again 5 with Bonferroni), while this was the case for the IA at set size 5 for both Dice and Accuracy (also 5 with Bonferroni). For the Dutch furniture data, Graph reached the ceiling at set size 5 for both Dice and Accuracy (and even at 1 with the Bonferroni test), while this was at set size 1 for the IA (again 1 with Bonferroni).

The people domain Table 3 shows the Dice and Accuracy scores in the people domain. Again, we found significant effects of *set size* for both the English data (Dice: $F_{(7,518)} = 39.46$, $p < .001$; Acc.: $F_{(7,518)} = 11.77$, $p < .001$) and the Dutch data (Dice: $F_{(7,546)} = 33.90$, $p < .001$; Acc.: $F_{(7,546)} = 3.235$, $p < .01$). Again, this implies that for both languages, the number of descriptions used for training influenced the performance of both algorithms in terms of both Dice and Accuracy. Unlike we did in the furniture domain, we found no interactions between *set size* and *algorithm*, but we did find a main effect of algorithm for the Dutch people data (Dice: $F_{(1,78)} = .751$, ns; Acc.: $F_{(1,78)} = 5.099$, $p < .05$), showing that Graph generated Dutch descriptions that were more accurate than those generated by the IA.

As in the furniture domain, post hoc tests showed that small set sizes were generally sufficient to reach ceiling performance. For the English data, Graph reached the ceiling at set size 20 for both Dice and Accuracy (with Bonferroni: 10 for Dice, 1 for Accuracy), while this was the case for the IA at set size 10 for Dice (also 10 with Bonferroni), and at set size 40 for Accuracy (and even at 1 with Bonferroni). For the Dutch data, Graph reached the ceiling at set size 30 for Dice (20 with Bonferroni), and at set size 5 for Accuracy (1 with Bonferroni). For the IA, ceiling was reached at set size 20 for Dice (Bonferroni: 5), and already at 1 for Accuracy (Bonferroni: 1).

6 Discussion and Conclusion

Our main goal was to investigate how many human-produced references are required by REG algorithms such as the Incremental Algorithm and the graph-based algorithm to determine preferences (or costs) for a new domain, and to generate "human-like" descriptions for new objects in these domains. Our results show that small data sets can be used to train these algorithms, achieving results that are not significantly different from those derived from a much larger training set. In the simple furniture domain even one training item can already be sufficient, at least for the IA. As shown in Table 4, on the whole the IA needed fewer training data than Graph (except in the English people domain, where Graph only needed a set size of 10 to hit the ceiling for Dice, while the IA needed a set size of 20).

Given that the IA ranks attributes, while the graph-based REG algorithm ranks attribute-value pairs, the difference in required training data is not surprising. In any domain, there will be more attribute-value pairs than attributes, so determining an attribute ranking is an easier task than determining a ranking of attribute-value pairs. Another advantage of ranking attributes rather than attribute-value pairs is that it is less vulnerable to the problem of "missing data". More specifically, the chance that a specific attribute does not occur in a small training set is much smaller than the chance that a specific attribute-value pair does not occur. As a consequence, the IA needs fewer data to obtain complete attribute orderings than Graph needs to obtain costs for all attribute-value pairs.

Interestingly, we only found interactions between training set size and algorithm in the furniture domain. In the people domain, there was no significant difference between the size of the training sets required by the algorithms. This could be explained by the fact that the people domain has about twice as many attributes as the furniture domain, and fewer values per attribute (see Table 1). This means that for people the difference between the number of attributes (IA) and the number of attribute-value pairs (Graph) is not as big as for furniture, so the two algorithms are on more equal grounds.

Both algorithms performed better on furniture than on people. Arguably, the people pictures in the TUNA experiment can be described in many more different ways than the furniture pictures can, so it stands to reason that ranking potential attributes and values is more difficult in the people than in the furniture domain. In a similar vein, we might expect Graph's flexible generation strategy to be more useful in the people domain, where more can be gained by the use of costs, than in the furniture domain, where there are relatively few options anyway, and a simple linear ordering may be quite sufficient.

This expectation was at least partially confirmed by the results: although in most cases the differences are not significant, Graph tends to perform numerically better than the IA in the people domain. Here we see the pay-off of Graph's more fine-grained preference ranking, which allows it to distinguish between more and less salient attribute values. In the furniture domain, most attribute values appear to be more or less equally salient (e.g., none of the colours gets notably mentioned more often), but in the people domain certain values are clearly more salient than others. In particular, the attributes HASBEARD and HASGLASSES are among the most frequent attributes in the people domain when their value is *true* (i.e., the target object can be distinguished by his beard or glasses), but they hardly get mentioned when their value is *false*. Graph quickly learns this distinction, assigning low costs and a high ranking to <HASBEARD, *true*> and <HASGLASSES, *true*> while assigning high costs and a low ranking to <HASBEARD, *false*> and <HASGLASSES, *false*>. The IA, on the other hand, does not distinguish between the values of these attributes.

Moreover, the graph-based algorithm is arguably more generic than the Incremental Algorithm, as it can straightforwardly deal with relational properties and lends itself to various extensions (Krahmer et al., 2003). In short, the larger training investment required for Graph in simple domains may be compensated by its versatility and better performance on more complex domains. To test this assumption, our experiment should be repeated using data from a more realistic and complex domain, e.g., geographic descriptions (Turner et al., 2008). Unfortunately, currently no such data sets are available.

Finally, we found that the results of both algorithms were better for the Dutch data than for the English ones. We think that this is not so much an ef-

fect of the language (as English and Dutch are highly comparable) but rather of the way the TUNA and D-TUNA corpora were constructed. The D-TUNA corpus was collected in more controlled conditions than TUNA and as a result, arguably, it contains training data of a higher quality. Also, because the D-TUNA corpus does not contain any location properties (X- and Y-DIMENSION) its furniture and people domains are slightly less complex than their TUNA counterparts, making the attribute selection task a bit easier.

One caveat of our study is that so far we have only used the standard automatic metrics on REG evaluation (albeit in accordance with many other studies in this area). However, it has been found that these do not always correspond to the results of human-based evaluations, so it would be interesting to see whether the same learning curve effects are obtained for extrinsic, task based evaluations involving human subjects. Following Belz and Gatt (2008), this could be done by measuring reading times, identification times or error rates as a function of training set size.

Comparing IA with FB and GR We have shown that small set sizes are sufficient to reach ceiling for the IA. But which preference orders (PO's) do we find with these small set sizes? And how does the IA's performance with these orders compare to the results obtained by alternative algorithms such as Dale and Reiter's (1995) classic Full Brevity (FB) and Greedy Algorithm (GR)? – a question explicitly asked by van Deemter et al. (2012). In the furniture domain, all five English training sets of size 5 yield a PO for which van Deemter et al. showed that it causes the IA to significantly outperform FB and GR (i.e., either C(olor)O(rientation)S(ize) or CSO; note that here we abstract over TYPE which van Deemter and colleagues do not consider). When we look at the English people domain and consider set size 10 (ceiling for Dice), we find that four out of five sets have a preference order where HAIRCOLOUR, HASBEARD and HASGLASSES are in the top three (again disregarding TYPE); one of these is the best performing preference order found by van Deemter and colleagues (GBH), another performs slightly less well but still significantly better than FB and GR (BGH); the other two score statistically comparable to the classical algorithms. The fifth people PO includes X- and Y-DIMENSION in the top three, which van Deemter et al. ignore. In sum: in almost all cases, small set sizes (5 and 10 respectively) yield POs with which the IA performs at least as well as the FB and GR algorithms, and in most cases significantly better.

Conclusion We have shown that with few training instances, acceptable attribute selection results can be achieved; that is, results that do not significantly differ from those obtained using a much larger training set. Given the scarcity of resources in this field, we feel that this is an important result for researchers working on REG and Natural Language Generation in general. We found that less training data is needed in simple domains with few attributes, such as the furniture domain, and more in relatively more complex domains such as the people domain. The data set being used is also of influence: better results were achieved with D-TUNA than with the TUNA corpus, which probably not so much reflects a language difference, but a difference in the way the corpora were collected.

We found some interesting differences between the IA and Graph algorithms, which can be largely explained by the fact that the former ranks attributes, and the latter attribute-value pairs. The advantage of the former (coarser) approach is that overall, fewer training items are required, while the latter (more fine-grained) approach is better equipped to deal with more complex domains. In the furniture domain both algorithms had a similar performance, while in the people domain Graph did slightly better than the IA. It has to be kept in mind that these results are based on the relatively simple furniture and people domains, and evaluated in terms of a limited (though standard) set of evaluation metrics. We hope that in the near future semantically transparent corpora for more complex domains will become available, so that these kinds of learning curve experiments can be replicated.

Acknowledgments Krahmer and Koolen received financial support from The Netherlands Organization for Scientific Research (NWO Vici grant 27770007). We thank Albert Gatt for allowing us to use his implementation of the IA, and Sander Wubben for help with k-means clustering.

References

Anja Arts, Alfons Maes, Leo Noordman, and Carel Jansen. 2011. Overspecification facilitates object identification. *Journal of Pragmatics*, 43(1):361–374.

Anja Belz and Albert Gatt. 2008. Intrinsic vs. extrinsic evaluation measures for referring expression generation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL'08)*, pages 197–200.

Robert Dale and Ehud Reiter. 1995. Computational interpretation of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.

Paul E. Engelhardt, Karl G.D Bailey, and Fernanda Ferreira. 2006. Do speakers and listeners observe the Gricean Maxim of Quantity? *Journal of Memory and Language*, 54:554–573.

Giuseppe Di Fabbrizio, Amanda Stent, and Srinivas Bangalore. 2008. Trainable speaker-based referring expression generation. In *Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, pages 151–158.

Albert Gatt, Ielka van der Sluis, and Kees van Deemter. 2007. Evaluating algorithms for the generation of referring expressions using a balanced corpus. In *Proceedings of the 11th European Workshop on Natural Language Generation (ENLG 2007)*, pages 49–56.

Albert Gatt, Anja Belz, and Eric Kow. 2009. The TUNA-REG Challenge 2009: Overview and evaluation results. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, pages 174–182.

Pablo Gervás, Raquel Hervás, and Carlos Léon. 2008. NIL-UCM: Most-frequent-value-first attribute selection and best-scoring-choice realization. In *Proceedings of the 5th International Natural Language Generation Conference (INLG 2008)*, pages 215–218.

Pamela W. Jordan and Marilyn Walker. 2005. Learning content selection rules for generating object descriptions in dialogue. *Journal of Artificial Intelligence Research*, 24:157–194.

John Kelleher. 2007. DIT - frequency based incremental attribute selection for GRE. In *Proceedings of the MT Summit XI Workshop Using Corpora for Natural Language Generation: Language Generation and Machine Translation (UCNLG+MT)*, pages 90–92.

Ruud Koolen and Emiel Krahmer. 2010. The D-TUNA corpus: A Dutch dataset for the evaluation of referring expression generation algorithms. In *Proceedings of the 7th international conference on Language Resources and Evaluation (LREC 2010)*.

Emiel Krahmer, Sebastiaan van Erk, and André Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29(1):53–72.

Chris Mellish, Donia Scott, Lynn Cahill, Daniel Paiva, Roger Evans, and Mike Reape. 2006. A reference architecture for natural language generation systems. *Natural Language Engineering*, 12:1–34.

Thomas Pechmann. 1989. Incremental speech production and referential overspecification. *Linguistics*, 27:98–110.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating NLG systems. *Computational Linguistics*, 35(4):529–558.

Philipp Spanger, Takehiro Kurosawa, and Takenobu Tokunaga. 2008. On "redundancy" in selecting attributes for generating referring expressions. In *COLING 2008: Companion volume: Posters*, pages 115–118.

Mariët Theune, Ruud Koolen, Emiel Krahmer, and Sander Wubben. 2011. Does size matter – How much data is required to train a REG algorithm? In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 660–664, Portland, Oregon, USA.

Ross Turner, Somayajulu Sripada, Ehud Reiter, and Ian P. Davy. 2008. Using spatial reference frames to generate grounded textual summaries of georeferenced data. In *Proceedings of the 5th International Natural Language Generation Conference (INLG)*, pages 16–24.

Kees van Deemter, Ielka van der Sluis, and Albert Gatt. 2006. Building a semantically transparent corpus for the generation of referring expressions. In *Proceedings of the 4th International Natural Language Generation Conference (INLG 2006)*, pages 130–132.

Kees van Deemter, Albert Gatt, Ielka van der Sluis, and Richard Power. 2012. Generation of referring expressions: Assessing the Incremental Algorithm. *Cognitive Science, to appear*.

Jette Viethen and Robert Dale. 2010. Speaker-dependent variation in content selection for referring expression generation. In *Proceedings of the 8th Australasian Language Technology Workshop*, pages 81–89.

Jette Viethen, Robert Dale, Emiel Krahmer, Mariët Theune, and Pascal Touset. 2008. Controlling redundancy in referring expressions. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, pages 239–246.