A Comparative Study of Weighting Schemes for the Interpretation of Spoken Referring Expressions

Su Nam Kim, Ingrid Zukerman, Thomas Kleinbauer and **Masud Moshtaghi**
Clayton School of Information Technology, Monash University
Clayton, Victoria 3800, Australia
`firstname.lastname@monash.edu`

Abstract

This paper empirically explores the influence of two types of factors on the interpretation of spoken object descriptions: (1) descriptive attributes, e.g., colour and size; and (2) interpretation stages, e.g., syntax and pragmatics. We also investigate two schemes for combining attributes when estimating the goodness of an interpretation: Multiplicative and Additive. Our results show that the former scheme outperforms the latter, and that the weights assigned to the attributes of a description and the stages of an interpretation influence interpretation accuracy.

1 Introduction

Referring expressions have been the topic of considerable research in *Natural Language Generation* (*NLG*) and psychology. In particular, attention has been paid to the usage of descriptive attributes, such as lexical item, colour, size, location and orientation (Section 2).

In this paper, we present an empirical study that examines the contribution of two types of factors to the understanding of spoken descriptions: (1) *descriptive attributes*, such as colour and size; and (2) *stages of an interpretation*, e.g., syntax and pragmatics. Our study was conducted in the context of *Scusi?*, a *Spoken Language Understanding* (*SLU*) system that interprets descriptions of household objects (Zukerman et al., 2008) (Section 3). Given a description such as "the *large blue* mug", where the descriptive attributes pertain to colour and size, in the absence of such a mug, should an SLU system prefer a large pink mug or a small blue mug? A preference for the former favours size over colour, while preferring the latter has the opposite effect. Similarly, considering the stages of an interpretation, if an *Automatic Speech Recognizer* (*ASR*) produces the text "the *played* inside the microwave" when a speaker says "the *plate* inside the microwave", should an SLU system prefer interpretations comprising objects inside the microwave or interpretations where "played" is considered a verb? A preference for the former favours pragmatics, while a preference for the latter favours the heard text.

We represent the contribution of a factor by assigning it a weight — factors with a higher weight are more influential than those with a lower weight; and investigate two methods for learning the weights of the factors pertaining to descriptive attributes and to interpretation stages: steepest ascent hill climbing and a genetic algorithm (Section 4). In addition, we consider two schemes for combining descriptive attributes, viz *Multiplicative* and *Additive* (Section 3.1). Our contribution pertains to the idea of empirically determining the influence of different factors on the interpretation accuracy of an SLU module, the methods for doing so, and the analysis of our results.

The rest of this paper is organized as follows. Next, we discuss related work. In Section 3, we outline our SLU system and the schemes for combining descriptive attributes. The learning algorithms appear in Section 4, and the results of our evaluation experiments in Section 5, followed by concluding remarks.

2 Related Work

The use and importance of different attributes in object descriptions has been studied both in psychology and in *NLG* (Krahmer and van Deemter, 2012), but there is little related research in *Natural Language Understanding* (*NLU*). Further, we have found no work on the relative importance of the different interpretation stages, e.g., is pragmatics more important than parsing?

Several studies have found that people tend

Su Nam Kim, Ingrid Zukerman, Thomas Kleinbauer and Masud Moshtaghi. 2014. A Comparative Study of Weighting Schemes for the Interpretation of Spoken Referring Expressions. In *Proceedings of Australasian Language Technology Association Workshop*, pages 50−58.

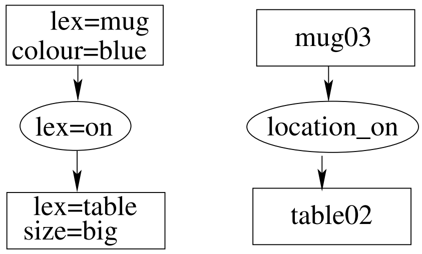
to include in their descriptions attributes that do not add discriminative power, e.g., (Dale and Reiter, 1995; Levelt, 1989, p. 129–134), which can be partly explained by the incremental nature of human language production and understanding (Pechmann, 1989; Kruijff et al., 2007). The incrementality of human speech was also considered by van der Sluis and Krahmer (**?**), in combination with object salience, when generating multimodal object references; while van Deemter (2006) and Mitchell *et al.* (2011) studied the generation of descriptions that employ gradable attributes, obtained from numerical data, focusing on size-related modifiers.

Gatt *et al.* (2007) compared the performance of several generation algorithms with respect to a combination of features, viz colour, position (restricted to placement in a grid), orientation and size. Their algorithm produced descriptions similar to those generated by people when the priority order of the attributes was *colour* ≻ *orientation* ≻ *size*. In contrast, Herrmann and Deutsch (1976) found that the choice of discriminative attributes is perceptually driven, but posited that there is no universally applicable priority ordering of attributes. This view was extended by Dale and Reiter (1995, p. 20), who suggested investigations to determine the priority order of attributes for different domains.

In this paper, we apply Dale and Reiter's suggestion to the understanding of spoken descriptions. However, rather than finding a priority order of attributes like Gatt *et al.* (2007), we learn weights that reflect the importance of descriptive attributes, and consider two schemes for combining these attributes. In addition, we extend this idea to the processing stages employed when interpreting descriptions.

3 SLU Systems and Case Study

The study described in this paper was conducted in the context of our SLU system *Scusi?* (Zukerman et al., 2008). However, what is important is not the specifics of a particular system, but the features of SLU systems to which our study is relevant. Specifically, the systems in question must have several processing stages, e.g., ASR, syntax, semantics and pragmatics; each processing stage must produce an N-best list of outputs (interpretations), e.g., N parse trees; and each interpretation generated at each stage must be assigned a score or probability that reflects its goodness.



(a) UCG (b) ICG

Figure 1: Sample UCG and ICG for "the blue mug on the big table".

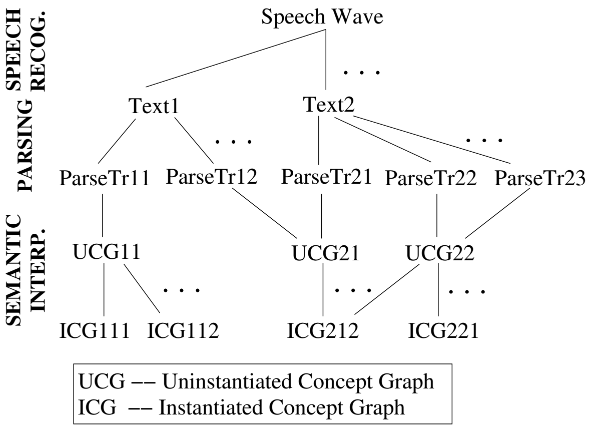


Figure 2: *Scusi?*'s processing stages.

Scusi? has four interpretation stages: ASR (Microsoft Speech SDK 6.1) produces candidate texts from a speech wave; Syntax (Charniak's probabilistic parser) generates parse trees; Semantics produces *uninstantiated Concept Graphs* (*UCGs*) (Sowa, 1984); and Pragmatics generates *instantiated Concept Graphs* (*ICGs*). Each of these outputs is assigned a probability. A UCG contains descriptive attributes (lexical item, colour and size of concepts, and positional relations between concepts) extracted from its "parent" parse tree. An ICG contains candidate objects within the current context (e.g., a room) and positional relations that reflect those specified in its parent UCG. For instance, Figure 1(a) shows one of the UCGs returned for the description "the blue mug on the big table", and Figure 1(b) displays one of the ICGs generated for this UCG. Note that the concepts in the UCG have generic names, e.g., `mug`, while the ICG contains specific objects, e.g., `mug03`, which is a candidate match for `lex=mug, colour=blue`.

Most interpretation stages can produce multiple outputs for a given input, e.g., up to 50 parse trees for a given ASR output. The graph of all possible interpretations (Figure 2) is usually too large to explore exhaustively in a practical spoken dialogue system. Therefore, *Scusi?* initially generates a promising ICG by selecting the top-ranked interpretation for each stage, and then performs a prescribed number of iterations as follows: in each iteration, an interpretation type (speech wave, text, parse tree or UCG) is selected probabilistically for further processing, giving preference to types produced by later interpretation stages in order to increase the specificity of the generated interpretations. An interpretation of the selected type is then probabilistically chosen for expansion, giving preference to more promising (higher scoring) interpretations, e.g., if a text is chosen, then a new parse tree for this text is added to the list of available parse trees.

3.1 Probability of an interpretation

The scores produced by *Scusi?* are in the $[0, 1]$ range, allowing them to be interpreted as *subjective probabilities* (Pearl, 1988). After making conditional independence assumptions, the probability of an ICG is estimated as follows (Zukerman et al., 2008):

$$\begin{aligned}\Pr(I|S,\mathcal{C}) \quad = \quad & \Pr(T|S)^{W_t}\Pr(P|T)^{W_p} \\ & \Pr(U|P)^{W_u}\Pr(I|U,\mathcal{C})^{W_i}\end{aligned} \tag{1}$$

where S, T, P, U and I denote speech wave, textual interpretation, parse tree, UCG and ICG respectively, and $\mathcal{C}$ denotes the current context (e.g., a room). The weights W_t, W_p, W_u and W_i reflect the importance of the outcome of each interpretation stage, i.e., ASR (text), Syntax (parse tree), Semantics (UCG) and Pragmatics (ICG).

The first two probabilities in Equation 1 are obtained from the ASR and the parser. The third probability, which reflects the complexity of a semantic interpretation, is estimated as the reciprocal of the number of nodes in a UCG. The last probability, viz the probability of an ICG I given UCG U and context $\mathcal{C}$, reflects the goodness of the match between ICG I and its parent UCG U in context $\mathcal{C}$. Specifically, the probability of I is estimated by a combination of functions that calculate how well the actual attributes of the objects in I (lexical item, colour, size and positional relation) match those specified in its parent UCG U.

We studied two schemes for combining these functions: *Multiplicative* and *Additive*.

Multiplicative scheme. This scheme is similar to that used in Equation 1:

$$\begin{aligned}\text{SC}_{\text{MULT}}(I|U,\mathcal{C}) = \prod_{i=1}^{N}\prod_{j=1}^{N}\Pr(\text{loc}(k_i,k_j))^{W_{loc}} \times \quad (2) \\ \prod_{i=1}^{N}\Pr(u_{i,lex}|k_i)^{W_{lex}}\Pr(u_{i,col}|k_i)^{W_{col}}\Pr(u_{i,siz}|k_i)^{W_{siz}},\end{aligned}$$

where N is the number of objects in ICG I, and the weights W_{lex}, W_{col}, W_{siz} and W_{loc} reflect the importance of lexical item, colour, size and location respectively. The second line in Equation 2 represents how well each object k_i in ICG I matches the lexical item, colour and size specified in its parent concept u_i in UCG U; and the first line represents how well the relative locations of two objects k_i and k_j in context $\mathcal{C}$ (e.g., a room) match their specified locations in U (e.g., $on(k_i, k_j)$). For instance, given the ICG in Figure 1(b), the second line in Equation 2 estimates the probability that `mug03` could be called “mug” and its colour could be called “blue” (no size was specified), and the probability that `table02` could be called “table” and considered “big” (no colour was specified). The first line estimates the probability that `mug03` could be said to be on `table02` (if the mug is elsewhere, this probability is low).

This scheme is rather unforgiving of partial matches or mismatches, e.g., the probability of a lexical match between “mug” and `cup01`, which is less than 1, is substantially reduced when raised to an exponent greater than 1; and a mismatch of a single attribute in an ICG significantly lowers the probability of the ICG.

Additive scheme. This more forgiving scheme employs the following formulation to estimate the probability of an ICG I given its parent UCG U and context $\mathcal{C}$:

$$\begin{aligned}\text{SC}_{\text{ADD}}(I|U,\mathcal{C}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\Pr(\text{loc}(k_i,k_j))W_{loc} + \quad (3) \\ \sum_{i=1}^{N}\{\Pr(u_{i,lex}|k_i)W_{lex} + \Pr(u_{i,col}|k_i)W_{col} + \\ \Pr(u_{i,siz}|k_i)W_{siz}\}\ .\end{aligned}$$

In principle, this scheme could also be applied to combining the probabilities of the interpretation stages. However, we did not explore this option owing to its inferior performance with respect to descriptive attributes (Section 5).

Probabilities from different sources

There are large variations in the probabilities returned by the different interpretation stages. In particular, the probabilities returned by the parser are several orders of magnitude smaller than those returned by the other stages. To facilitate the learning of weights, we adjust the probabilities returned by the different interpretation stages so that they are of a similar magnitude. To this effect, we adopt two approaches: (1) adjusting the probabilities returned by the parser by calculating their standardized score z_i, and (2) normalizing the probabilities of the ICGs by introducing a factor that depends on the weights assigned to different descriptive attributes. The second approach takes advantage of specific information about ICGs, which is not available about parse trees.

Adjusting parse-tree probabilities. Given a probability p_i returned by the parser, we calculate its z-score z_i for an input value x_i as follows: $z_i = (x_i - \mu)/\sigma$, where μ is the mean and σ the standard deviation of the probabilities returned by the parser for our development corpus (Section 5.2). The z_i scores are then transformed to the $[0, 1]$ range using a sigmoid function $z_i^{Norm} = \frac{1}{1+e^{-z_i}}$.

Normalizing ICG probabilities. The ICG scores obtained by the Multiplicative scheme are often in a small band in a very low range, while the ICG scores obtained by the Additive scheme are typically greater than 1. In order to expand the range of the former, and map the latter into the $[0, 1]$ range, we incorporate the following normalizing factor φ into their formulation:

$$\varphi = \sum_{i=1}^{M}\sum_{j=1}^{M} W_{loc} + \sum_{i=1}^{M}\{W_{lex} + W_{col} + W_{siz}\}\,,$$

where the weights correspond to the descriptive attributes that were mentioned.

This factor is incorporated into the Multiplicative and Additive schemes as follows:

- **Multiplicative scheme.**

$$\text{Pr}_{\text{MULT}}(I|U,\mathcal{C}) = \text{SC}_{\text{MULT}}(I|U,\mathcal{C})^{1/\varphi} \qquad (4)$$

- **Additive scheme.**

$$\text{Pr}_{\text{ADD}}(I|U,\mathcal{C}) = \frac{1}{\varphi}\text{SC}_{\text{ADD}}(I|U,\mathcal{C}) \qquad (5)$$

4 Learning Weights

In this section, we describe the algorithms used to learn the weights for the interpretation stages (W_t, W_p, W_u, W_i) and the descriptive attributes (W_{lex}, W_{col}, W_{siz}, W_{loc}), and our evaluation metrics.

4.1 Learning algorithms

In order to learn the values of the weights, an SLU system must be run on the entire training corpus each time a set of weights is tried. To control the search time, *Scusi?* was set to perform 150 iterations. In addition, we investigated only irrevocable search strategies: steepest ascent hill climbing and a genetic algorithm, employing two ranges of integer weights: a small range of $[1, 4]$ for our cross-validation experiment (Section 5), and a larger range of $[1, 20]$ for our development-dataset experiment (Section 5.2). In general, the algorithms produced low weights, except for the genetic algorithm in the development-dataset experiment, where it generated high weights for most descriptive attributes.

The fitness function for both search strategies is the system's average performance on a training corpus using the *NDCG*@10 metric. This metric, which is described in Section 4.2, was chosen due to its expressiveness, and the value 10 was selected because a response generation system may plausibly inspect the top 10 interpretations returned by an SLU module.

Steepest ascent hill climbing (SA). All weights are initially set to 1. In each iteration, a weight is increased by 1 while keeping the other weights at their previous value;[1] the weight configuration that yields the best performance is retained. This process is repeated until performance no longer improves after one round of changes.

Genetic algorithm (GA). One set of weights is considered a gene. Owing to the relatively long processing time for training corpus runs, we restrict the gene population size to 15, initialized with random values for all weights. In each iteration, the 10 best performing genes are kept, and the other five genes are replaced with offspring of the retained genes. Offspring are generated by selecting two genes probabilistically, with better genes having a higher selection probability, and probabilistically choosing between mutation and

[1] In preliminary experiments, we considered increments of 0.5, but this did not affect the results.

crossover, and between the parent weights to be retained in a crossover operation. This process is repeated until the performance of the population does not improve four times in a row.

4.2 Evaluation metrics

Scusi?'s performance is evaluated using two measures: *Fractional Recall* $@K$ (*FRecall*$@K$) and *Normalized Discounted Cumulative Gain* $@K$ (*NDCG*$@K$) (Järvelin and Kekäläinen, 2002).

FRecall$@K$ is a variant of Recall that accounts for the fact that an N-best system ranks equiprobable interpretations arbitrarily:

$$\mathit{FRecall}@K(d) = \frac{\sum_{j=1}^{K} \mathit{fc}(I_j)}{|C(d)|}\,,$$

where $C(d)$ is the set of correct interpretations for description d, I_j is a candidate interpretation for d, and *fc* is the fraction of correct interpretations among those with the same probability as I_j (it is a proxy for the probability that I_j is correct).

DCG$@K$ is similar to *FRecall*, but provides a finer-grained account of rank by discounting interpretations with higher (worse) ranks. This is done by dividing $\mathit{fc}(I_j)$ by a logarithmic penalty that reflects I_j's rank:

$$\mathit{DCG}@K(d) = \mathit{fc}(I_1) + \sum_{j=2}^{K} \frac{\mathit{fc}(I_j)}{\log_2 j}\,.$$

DCG$@K$ is normalized to the $[0, 1]$ range by dividing it by the *DCG*$@K$ score of an ideal N-best result, where the $|C(d)|$ correct interpretations of description d are ranked in the first $|C(d)|$ places:

$$\mathit{NDCG}@K(d) = \frac{\mathit{DCG}@K(d)}{1 + \sum_{j=2}^{\min\{|C(d)|,K\}} \frac{1}{\log_2 j}}\,.$$

5 Evaluation

In this section, we describe two experiments where we compare the performance of three versions of *Scusi?*: (1) with weights learned by SA, (2) with weights learned by GA, and (3) with Unity weights (all descriptive attributes and interpretation stages have a weight of 1).

As mentioned in Section 4.1, the entire training corpus must be processed for each weight configuration, resulting in long training times, in particular for GA. To reduce these times, rather than trying to learn all the weights at once, we first learned the weights of descriptive attributes, and then used the results of this experiment to learn the weights of the interpretation stages. Further, the former weights were learned from manually transcribed texts, while the latter were learned from actual ASR outputs. This was done because descriptive attributes in head nouns (lexical item, colour and size) were often mis-heard by the ASR, which hampers a learning system's ability to determine their contribution to the performance of an SLU module.

Table 1: Distribution of descriptive attributes over the 341-dataset.

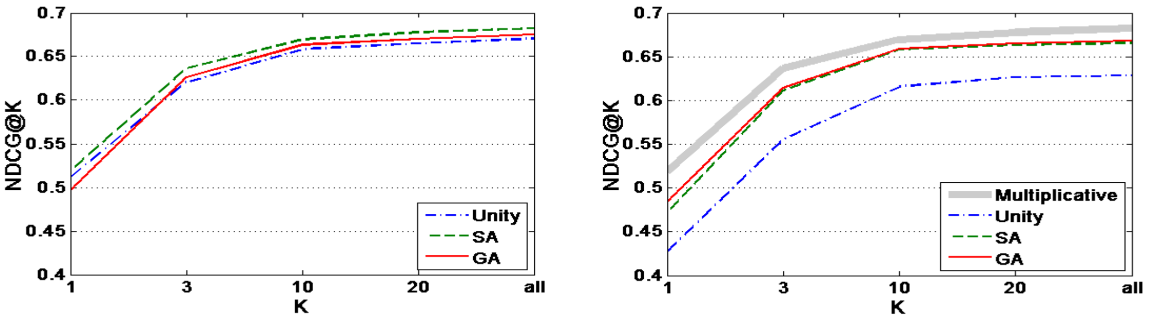
Attribute	**Number (%)**
Lexicon, Colour	14 (4.11%)
Lexicon, Colour, Position	150 (43.99%)
Lexicon, Colour, Size	3 (0.88%)
Lexicon, Colour, Position, Size	20 (5.87%)
Lexicon, Position	152 (44.57%)
Lexicon, Position, Size	2 (0.59%)
Lexicon, Size	0 (0.00%)
Total	341 (100%)

The resultant versions of *Scusi?* were evaluated using the corpus described in (Kleinbauer et al., 2013), denoted *341-dataset*, which consists of 341 free-form, spoken descriptions generated by 26 trial subjects for 12 objects within four diverse scenes (three objects per scene, where a scene contains between 9 and 17 objects). The descriptions are annotated with Gold standard ICGs. Table 1 displays the details of the descriptions and their attributes.

5.1 Experiment 1 – Cross-validation

Owing to run-time restrictions, we performed only three-fold cross validation, where the search algorithms were trained on 120 descriptions and tested on 221 descriptions. Both the training set and the test set were selected by stratified sampling according to the distribution of the descriptive attributes. Note that the training corpora comprise 360 descriptions in total, i.e., there are 19 extra descriptions in the training data because, as seen in Table1, descriptions containing size, e.g., "the *large* brown desk", were quite rare, and hence were included in more than one training set.

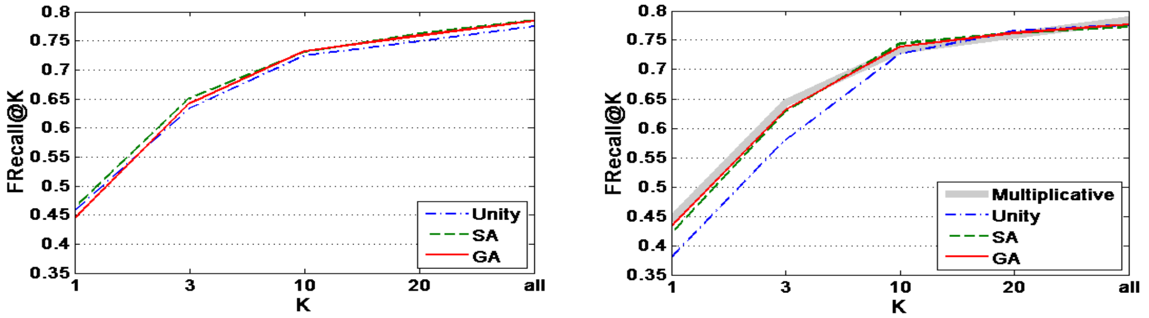
Each algorithm learned weight configurations for the interpretation stages and the descriptive attributes for each validation fold. The weights learned by SA generally differed from those learned by GA, and there were some differences in



(a) Multiplicative scheme

(b) Additive scheme

Figure 3: Average *NDCG*@K obtained over three-fold cross validation (scale 0.4-0.7).



(a) Multiplicative scheme

(b) Additive scheme

Figure 4: Average *FRecall*@K obtained over three-fold cross validation (scale 0.35-0.8).

the weights learned for each fold. Both algorithms assigned a weight of 1 to the ASR stage under the Additive attribute-combination scheme, and a higher weight under the Multiplicative scheme; the Syntax stage mostly received a weight of 1; and the Semantics and Pragmatics stages were assigned higher weights. GA tended to assign higher weights than SA to descriptive attributes, while SA consistently ascribed a weight of 1 to size and location. Despite these differences, both algorithms outperformed the Unity baseline on the training set, with GA achieving the best results, and the Multiplicative scheme outperforming the Additive scheme.

Results

Figures 3 and 4 display the average of *NDCG*@K and *FRecall*@K respectively for the three validation folds for $K \in \{1, 3, 10, 20, \text{all}\}$ under the Multiplicative and the Additive attribute-combination schemes (the grey shadow in Figures 3b and 4b represents the best performance obtained under the Multiplicative scheme for ease of comparison). Note that owing to the small number of folds, statistical significance cannot be calculated.

Performance across attribute-combination schemes – the Multiplicative scheme outperforms the Additive scheme in terms of *NDCG*@K for all values of K, and performs slightly better than the Additive scheme in terms of *FRecall*@K for $K \in \{1, 3, \text{all}\}$ (the schemes perform similarly for $K \in \{10, 20\}$).

Comparison with Unity – in terms of *NDCG*@K, SA outperforms Unity for all values of K under both attribute-combination schemes; and GA outperforms Unity for all values of K under the Additive scheme, and for $K \geq 3$ under the Multiplicative scheme. In terms of *FRecall*@K, SA outperforms Unity for all values of K under the Multiplicative scheme; GA performs better than Unity under the Multiplicative scheme for $K \geq 3$; and both SA and GA outperform Unity for $K \leq 10$ under the Additive scheme (all the schemes perform similarly for $K \in \{20, \text{all}\}$). It is worth noting the influence of the Additive scheme on Unity's performance in terms of *NDCG*@K, suggesting that Unity fails to find the correct interpretation more often than the other schemes or finds it at worse (higher) ranks.

SA versus GA – GA outperforms SA under the Additive scheme in terms of both performance metrics for $K = 1$, while SA outperforms GA in terms of *FRecall*@10. Under the Multiplicative scheme, SA performs better than GA in terms of *NDCG*@K for all values of K, and in terms of *FRecall*@K for $K \leq 3$. The algorithms perform similarly for the other values of K under both attribute-combination schemes.

Summary – SA's superior performance in terms of *NDCG*@K indicates that it finds the correct interpretations at lower (better) ranks than Unity and GA. GA's good performance on the training data, together with its slightly worse performance on the test data, suggests that GA over-fits the training data.

5.2 Experiment 2 – Development Set

The results of our first experiment show that the proposed weight-learning scheme for descriptive attributes and interpretation stages improves *Scusi?*'s performance. However, as seen in Table 1, speakers in this dataset used positional attributes in the vast majority of the descriptions, rarely using size. This influences the weights that were learned, in particular W_{siz}, as size had little effect on performance.

To address this issue, we conducted an experiment where we learned the weights on a handcrafted development dataset, and tested the performance of the three versions of our SLU system on the entire 341-dataset. The development dataset, denoted *62-dataset*, was designed to facilitate learning the influence of descriptive attributes on an SLU system's performance (assuming consistent ASR performance, the influence of the interpretation stages should be largely invariant across corpora). Thus, the descriptive attributes and combinations thereof are more evenly distributed in the development corpus than in the 341-dataset, but positional attributes still have a relatively high frequency. Table 2 displays the details of the descriptions in the 62-dataset and their attributes.

Despite the wider range of values considered for this experiment ($[1, 20]$), the weights learned by SA from the 62-dataset were only slightly different from those learned from the three training folds in the 341-dataset. In contrast, several of the weights learned by GA were in the high end of the range. This is partly explained by the fact that the genes in GA are randomly initialized from the entire range.

Table 2: Distribution of descriptive attributes over the 62-dataset.

Attribute	**Number (%)**
Lexicon, Colour	5 (8.06%)
Lexicon, Colour, Position	8 (12.90%)
Lexicon, Colour, Size	7 (11.30%)
Lexicon, Colour, Position, Size	8 (12.90%)
Lexicon, Position	20 (32.26%)
Lexicon, Position, Size	10 (16.13%)
Lexicon, Size	4 (6.45%)
Total	62 (100%)

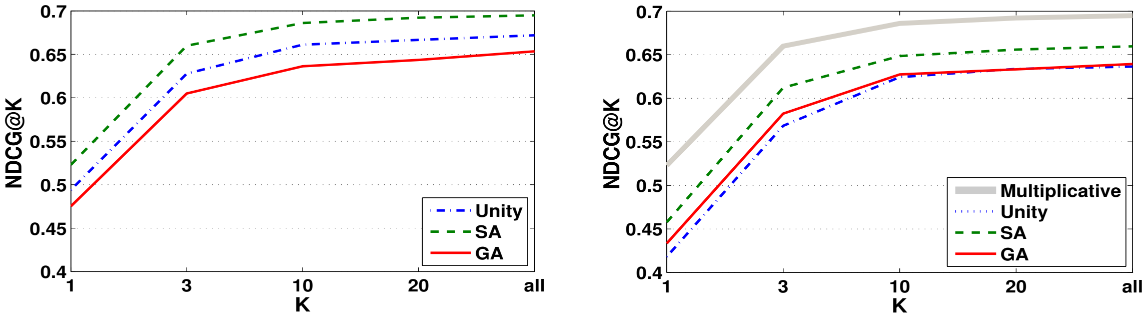
Results

Figures 5 and 6 respectively display the average of *NDCG*@K and *FRecall*@K for $K \in \{1, 3, 10, 20, \text{all}\}$ under the Multiplicative and the Additive attribute-combination schemes. Statistical significance was calculated using the two-tailed Wilcoxon signed rank test, and reported for *p-value* ≤ 0.05.

Performance across attribute-combination schemes – the Multiplicative scheme outperforms the Additive scheme in terms of *NDCG*@K for all values of K (statistically significant with *p-value* $\ll 0.01$ for SA and Unity, and only for $K = 1$ for GA). In terms of *FRecall*@K, the Multiplicative scheme outperforms the Additive scheme for all values of K for SA (statistically significant for $K \leq 10$), for $K \leq 3$ for Unity (statistically significant), and for $K \in \{1, 3, \text{all}\}$ for GA (statistically significant for $K = 1$).

Comparison with Unity – SA outperforms Unity under the Multiplicative attribute-combination scheme (statistically significant for *NDCG*@K for $K \geq 3$ and for *FRecall*@3). Under the Additive scheme, SA outperforms Unity in terms of *NDCG*@K (statistically significant for $K \in \{1, 3, 10, \text{all}\}$), but in terms of *FRecall*@K, SA outperforms Unity only for $K \leq 3$ (statistically significant for $K = 1$). In contrast to SA, GA's performance was rather disappointing, with Unity outperforming GA under the Multiplicative scheme (statistically significant for *NDCG*@20), and GA slightly outperforming Unity under the Additive scheme only for $K \leq 3$.

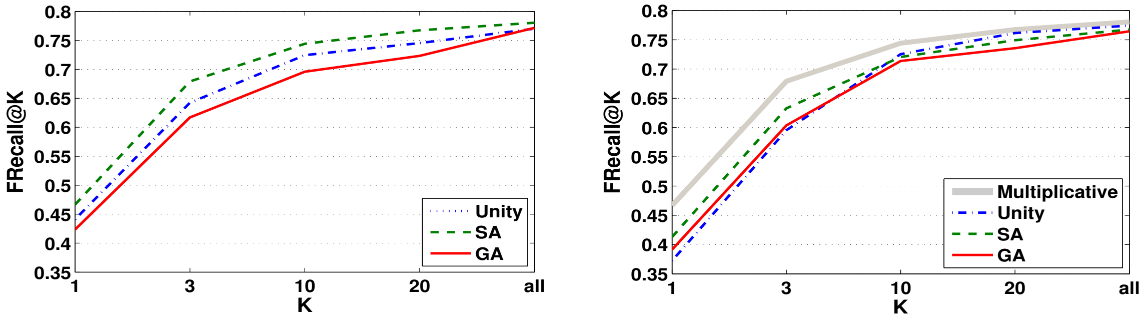
SA versus GA – SA consistently outperforms GA under both attribute-combination schemes for both performance metrics (statistically significant for all values of K in terms of *NDCG*@K and for $K \leq 20$ in terms of *FRecall*@K under the Multi-



(a) Multiplicative scheme

(b) Additive scheme

Figure 5: Average *NDCG*@K obtained from training on the 62-dataset (scale 0.4-0.7).



(a) Multiplicative scheme

(b) Additive scheme

Figure 6: Average *FRecall*@K obtained from training on the 62-dataset (scale 0.35-0.8).

plicative scheme, and in terms of *NDCG*@K for $K \in \{3, 10, 20\}$ under the Additive scheme).

Summary – the results of this experiment are consistent with those of the cross-validation experiment in the superior performance of the Multiplicative attribute-combination scheme, and of SA under this scheme. However, in this experiment, SA consistently outperforms GA under the Additive scheme, Unity outperforms GA under the Multiplicative scheme, and GA and Unity perform similarly under the Additive scheme.

6 Conclusion

We have offered an approach for learning the weights associated with descriptive attributes and the stages of an interpretation for an N-best, probabilistic SLU system that understands referring expressions in a household context. In addition, we have compared two schemes for combining descriptive attributes: Multiplicative and Additive.

Our results show that in the context of our application, interpretation performance can be improved by assigning different weights to different interpretation stages and descriptive attributes. Specifically, the best performance was obtained using weights learned with SA under the Multiplicative attribute-combination scheme. However, the fact that different weights were obtained for each validation fold and for the development dataset indicates that the weights are sensitive to the training corpus, and a larger training corpus is required. Nonetheless, despite the differences in the learned weights, SA performed similarly across both datasets/training-regimes, as did Unity. In contrast, GA exhibited larger differences between the weights and results obtained for the two datasets/training-regimes, in particular for the Additive attribute-combination scheme. This, together with GA's excellent performance on the training data, especially in the cross-validation experiment, compared to its performance on the test data, suggests that GA may be over-fitting the training data.

We also found that performance was sensitive to the values of tunable system parameters, such as the number of interpretations generated per description (*Scusi?* was set to generate only 150 interpretations to reduce the run time of the learning algorithms). The effect of these values on performance requires further investigation, e.g., learning the values of the system's parameters together with the weights of the descriptive attributes and the interpretation stages, which in turn would pose additional challenges for the learning process.

Acknowledgments

This research was supported in part by grant DP110100500 from the Australian Research Council.

References

R. Dale and E. Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 18(2):233–263.

A. Gatt, I. van der Sluis, and K. van Deemter. 2007. Evaluating algorithms for the generation of referring expressions using a balanced corpus. In *ENLG07 – Proceedings of the 11th European Workshop on Natural Language Generation*, pages 49–56, Saarbrücken, Germany.

T. Herrmann and W. Deutsch. 1976. *Psychologie der Objektbenennung*. Hans Huber.

K. Järvelin and J. Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Th. Kleinbauer, I. Zukerman, and S.N. Kim. 2013. Evaluation of the *Scusi?* spoken language interpretation system – A case study. In *IJCNLP2013 – Proceedings of the 6th International Joint Conference on Natural Language Processing*, pages 225–233, Nagoya, Japan.

E. Krahmer and K. van Deemter. 2012. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218.

G.-J. Kruijff, P. Lison, T. Benjamin, H. Jacobsson, and N. Hawes. 2007. Incremental, multi-level processing for comprehending situated dialogue in human-robot interaction. In *LangRo'2007 – Proceedings from the Symposium on Language and Robots*, pages 509–514, Aveiro, Portugal.

W.J.M. Levelt. 1989. *Speaking: from Intention to Articulation*. MIT Press.

M. Mitchell, K. van Deemter, and E. Reiter. 2011. Two approaches for generating size modifiers. In *ENLG2011 – Proceedings of the 13th European Workshop on Natural Language Generation*, pages 63–70, Nancy, France.

J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, San Mateo, California.

T. Pechmann. 1989. Incremental speech production and referential overspecification. *Linguistics*, 27:89–110.

J.F. Sowa. 1984. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, Reading, MA.

K. van Deemter. 2006. Generating referring expressions that involve gradable properties. *Computational Linguistics*, 32(2):195–222.

I. Zukerman, E. Makalic, M. Niemann, and S. George. 2008. A probabilistic approach to the interpretation of spoken utterances. In *PRICAI 2008 – Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence*, pages 581–592, Hanoi, Vietnam.