Evaluating an Open Domain GRE algorithm on closed domains System IDs: CAM-B, CAM-T, CAM-BU and CAM-TU

Advaith Siddharthan & Ann Copestake
University of Cambridge
{as372,aac10}@cl.cam.ac.uk

Abstract

We present four variations of our 2004 incremental algorithm (Siddharthan and Copestake, 2004), and present results on both the Furniture and People datasets.

1 Introduction

In Siddharthan and Copestake (2004), we presented an algorithm for generating referring expressions in open domains. Our algorithm was novel in that it was intended for open domains where attribute classification in infeasible, and in that it provided the first incremental algorithm that could handle relations as well as attributes. In that paper we evaluated our algorithm by trying to reproduce referrring expressions in the Penn WSJ Treebank. Here, we describe four variations of the general method described there and evaluate it on both the furniture and People datasets.

2 Overview of 2004 algorithm

GRE algorithms make the following assumptions, for example:

1. A semantic representation exists
2. A classification scheme for attributes exists
3. The values that attributes take are mutually exclusive
4. The linguistic realisations are unambiguous

These assumptions are violated when we move from generation in a very restricted domain to regeneration in an open domain. Our 2004 paper was aimed at designing an incremental approach that works when these assumptions are relaxed. Our alternative algorithm measured the relatedness of adjectives, rather than deciding if two of them are the same or not. It worked at the level of words, not their semantic labels. Further, it treated discriminating power as only one criteria for selecting attributes and allowed for the easy incorporation of other considerations such as reference modification.

2.1 Quantifying Discriminating Power

For each attribute of the referent, we define the following three quotients.

1. SQ: Similarity Quotient
2. CQ: Contrastive Quotient
3. DQ: Discriminating Quotient

These are meant to measure how similar or dissimilar an attribute of the referent is to attributes of distractors. Our open domain algorithm measured these quotients using synonym and antonym links in WordNet (Miller et al., 1993). For this close domain task (that makes all the assumptions above), these quotients are calculated more easily as follows.

For each attribute a of the referent, $SQ(a)$ is defined as the number of distractors that have the same value for a as the referent. $CQ(a)$ is the number of distractors that do not share the same value for a as the referent. $DQ(a)$, the discriminating power of a in this context, is then defined as $DQ(a) = CQ(a) - SQ(a)$.

Example

Consider three chairs: `e1`(a big black chair), `e2`(a small black chair) and `e3`(a small white chair).

Consider using the original Reiter and Dale (1992) incremental algorithm to refer to `e1` with **preferred**={`colour`, `size`}. The `colour` attribute *black* rules out `e3`. We then we have to select the `size` attribute *big* as well to rule out `e2`. We thus generate the sub-optimal expression *the big black dog*.

In our approach, for each of `e1`'s attributes, we calculate the three quotients with respect to `e2` and `e3`:

attribute	CQ	SQ	DQ
big	2	0	2
black	1	1	0

2.2 System CAM-B

Our incremental algorithm incorporates attributes in decreasing order of DQ. For this evaluation, this algorithm has the id CAM-B. In this algorithm, *big* has a higher discriminating power (2) than *black* (0) and rules out both `e2` and `e3`. We therefore generate *the big chair*. Our incremental approach thus manages to select the attribute that stands out in context because we construct the **preferred** list *after* observing context. This algorithm does not need any training (for open or closed domains) because it derives its notion of discriminating power entirely from the context.

2.3 System CAM-T

Unfortunately, for an evaluation such as this, CAM-B is suboptimal. The problem is with the assumption of mutual exclusivity of values. In reality, because of the setup of the experiment, some values have more descriminating power to human users than others. For example, *red* might be more distinguishable from *green* than *blue* is. Also, the difference between *small* and *large* images in the experiment was not very distinguishable, which might have caused human subjects to favour other attributes. To factor this in, we adjust $DQ(a)$ using a measure of how discriminating this value really is to humans. The adjustment we make is a linear function of the original: $DQ(a) \longrightarrow w_1(a)+w_2(a)\times DQ(a)$. The weights $w_1(a)$ and $w_2(a)$ are obtained by training on the combined development and training datasets. This submission has the id CAM-T.

2.4 Systems CAM-BU and CAM-TU

We also try a third algorithm, where DQs are updated at each incremental step (so that at each step, the attribute that is most discriminating relative to the the remaining distractors is selected). For this evaluation, these algorithm have the ids CAM-BU and CAM-TU. The former uses the DQ from CAM-B, while the latter uses the adjusted DQ from CAM-T.

System	Av. Dice / Av. Length	
	Training	Development
CAM-B	0.588/2.21	0.563/2.18
CAM-T	0.784/3.05	0.780/3.01
CAM-BU	0.606/2.15	0.585/2.14
CAM-TU	0.774/2.88	0.782/2.89

Table 1: Results for Furniture task

System	Av. Dice / Av. Length	
	Training	Development
CAM-B	0.681/2.32	0.688/2.34
CAM-T	0.681/2.32	0.688/2.34
CAM-BU	0.681/2.32	0.688/2.34
CAM-TU	0.681/2.32	0.688/2.34

Table 2: Results for People task

2.5 Notes

1. In each algorithm we force inclusion of the *type* attribute by setting $DQ(\text{type}) = \infty$. This causes it to be selected first in the incremental process.

2. In the People domain, some attributes are dependent. For example, selecting *hairColour* implies either *hasHair* or *hasBeard*. If *hairColour* is selected, we also include *hasHair* and *hasBeard* provided their value is 1.

3. In the People domain, all the referents can be distinguished using exactly one attribute. All our algorithms generate optimal referring expressions of length 1, if the two rules above are not included.

3 Results

We present our results in Tables 1 and 2.

On the furniture data, SC04-BASIC and SC04-BASIC-UPDATE perform comparable to full brevity (our implementation of full brevity gave 0.606/2.15 and 0.579/2.13 on the two sets). The trainable versions gave much higher Dice scores, and longer expressions (though still shorter than the human gold standard).

On the People data, all our algorithms would achieve full brevity if we did not forcibly include *type* and (when *hairColour* is selected) *hasHair* and/or *hasBeard*. The data is such that every referent is distinguished by exactly one attribute. All our algorithms find this attribute first.

4 Conclusions

We have described an algorithm for generating referring expressions that can be used in any domain. Our algorithm selects attributes that are distinctive in context.

References

George A. Miller, Richard Beckwith, Christiane D. Fellbaum, Derek Gross, and Katherine Miller. 1993. Five Papers on WordNet. Technical report, Princeton University, Princeton, N.J.

Ehud Reiter and Robert Dale. 1992. A fast algorithm for the generation of referring expressions. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, pages 232–238, Nantes, France.

Advaith Siddharthan and Ann Copestake. 2004. Generating referring expressions in open domains. In *Proceedings of the 42th Meeting of the Association for Computational Linguistics Annual Conference (ACL 2004)*, pages 407–414, Barcelona, Spain.