The Fingerprint of Human Referring Expressions and their Surface Realization with Graph Transducers

Bernd Bohnet
University of Stuttgart
Visualization and Interactive Systems Group
Universitätstr. 58, 70569 Stuttgart, Germany
bohnet@informatik.uni-stuttgart.de

Abstract

The algorithm IS-FP takes up the idea from the IS-FBN algorithm developed for the shared task 2007. Both algorithms learn the individual attribute selection style for each human that provided referring expressions to the corpus. The IS-FP algorithm was developed with two additional goals (1) to improve the indentification time that was poor for the FBN algorithm and (2) to push the dice score even higher. In order to generate a word string for the selected attributes, we build based on individual preferences a surface syntactic dependency tree as input. We derive the individual preferences from the training set. Finally, a graph transducer maps the input strucutre to a deep morphologic structure.

1 IS-FP: Generating Referring Expression with a Human Imprint

A review of the referring expressions shows that humans prefer frequently distinct attributes and attribute combination such as in the following examples.

> grey desk (30t1), a red chair (30t2), red sofa (30t3), blue chair (30t5), a small green desk (30t6)

> the one in the top left corner (31t1), the one to the left in the middle row (31t2), the bottom right most one (31t3), the blue chair at the bottom center (31t5), etc.

The first individual (#30) seems to prefer colour and size while the second one (#31) seems to prefer the relative position (*to the left*) and places (*the top left corner*). Because of the review, we checked, if the Incremental Algorithm (Dale and Reiter, 1995) using the order for the attributes due to the frequency calculated for each individual can outperform the algorithm using the order for the attributes due to the frequency of the complete training set. This was the case. Table 1 shows the results. Using the individual attribute order, the IA performed as good as the FBN algorithm, cf. Table 1.

Algorithm	Furniture	People	Avg.
IA (complete)	0.796	0.710	0.753
IA (individual)	0.835	0.736	0.7855
FBN	0.810	0.762	0.786

Table 1: Incremental algorithm and FBN

However, the FBN algorithm generates all possible referring expressions and selects based on the dice metric the most similar expressions of the same human. Since there is usually a set of equal good referring expressions, it is possible to select a referring expression among these results due to another metric. That this would improve the results shows the experiment to selected among these results the referring expression that is closest to the correct result. The outcome was that the FBN algorithm has still about 9% room for improvements. The following sections investigates possibilities to use this chance.

1.1 Identification Time

An important metric is the identification time that is the time which is used by a human to identify an entity due to a given referring expression. The identification time is very loosely related with the number of minimal referring expressions and therefore likely with the length of a referring expression. The best identification times had a system with 74% minimal referring expressions and the second and third best

systems had about 41%. Good identification times had nearly all systems with only a maximum difference of 0.38 seconds except FBN and FBS which are about 1.05 and 1.49 seconds behind the best one. This is a huge difference compared to all other systems. What could be the reason for that? We know of two differences of FBN to all other systems: **(1)** the lowest portion of minimal referring expressions of all systems and **(2)** the nearest neighbour learning technique. The number of minimal referring expressions is also different to the number of expressions found in the training set. Table 2 shows in the columns *human* the average length and portion of minimal human referring expressions. Because of the different length of the human and the generated expressions, we conducted the experiment to chose always the shortest. Table 2 shows the change between the random selection (FBN) and the selection of the shortest (FP). The experiment leads to a result that have a length and percentage of minimal referring expressions in average similar to the humans ones, cf. columns of *human* and *shortest*.

selection	human		random (FBM)		shortest (FP)	
RE	Len.	Min.	Len.	Min.	Len.	Min.
Furniture	3.1	26.3	3.5	9.4	3.1	15.9
People	3.0	30.9	2.8	28.8	2.8	30.8

Table 2: Length and portion of min. RE

The second difference is the use of the nearest neighbour technique. Could the poor identification time be caused by the nearest neighbour technique? How does it influence referring expressions? – The referring expressions are generated in all the different styles like the human expressions of the corpus. Do humans learn the style of referring expressions and expect then the next expression in the same style? And are we confused when we don't get what we expact? Or does FBN look too much on the expressions of the humans and too less on the domain? We hope to get answers for these questions from the shared task evaluation of IS-FP.

1.2 The IS-FP Algorithm

The basis for the IS-FP algorithm is an extended full brevity implementation in terms of problem solving by search which computes all referring expression, cf. (Bohnet and Dale, 2005). IS-FP uses also the nearest neighbour technique like the IS-FBN algorithm that was introduced by Bohnet (2007). With the nearest neighbour technique, IS-FP selects the expressions which are most similar to the referring expressions of the same human and a human that builds referring expressions similar. The similarity is computed as the average of all dice values between all combinations of the available trails for two humans. From the result of the nearest neighbour evaluation, FP selects the shortest and if still more than one expressions remain then it computes the similarity among them and chooses the most typical and finally, if still alternatives remain, it selects one with the attributes having the highest frequency. Table 3 shows the results for IS-FP trained on the training set and applied to the development set.

Set	Dice	MASI	Accuracy	Uniq.	Min.
Furniture	0.881	0.691	51.25%	100%	1.25%
People	0.790	0.558	36.8%	100%	0%
Total	0.836	0.625	44%	100%	0.62%

Table 3: Results for the IS-FP algorithm

2 IS-GT: Realization with Graph Transducers

We build the input depedency tree for the text generator due to the statistical information that we collect from the training data for each person. This procedure is consistent with our referring expression generator IS-FP that reproduces the individual imprint in a referring expression for the target person. We start with the realization of the referring expressions from a surface syntactic dependency tree, cf. (Mel'čuk, 1988). For the realization of the text, we use the Text Generator and Linguistic Environment MATE, cf. (Bohnet, 2006). We reportet the first time about MATE on the first International Natural Language Generation Conference, cf. (Bohnet et al., 2000). It was since then continuously enhanced and in the last years, large grammars for several languages such as Catalan, English, Finnish, French, German, Polish, Portougees have been developed within the European Project MARQUIS and PatExpert, cf. (Wanner et al., 2007), (Lareau and Wanner, 2007) and (Mille and Wanner, 2008).

2.1 The Referring Expression Models

A learning program builds a Referring Expression Model for each person that contributed referring expression to the corpus. The model contains the following information: (1) The lexicalization for the

values of a attribute such as couch for the value sofa, man for value person, etc. (2) The prefered usage of determiners for the type that can be definite (*the*), indefinite (*a*), no article. (3) The syntactic preferences such as *the top left chair*, *the chair at the bottom to the left*, etc.

The information about the determiner and the lexicalization is collected from the annotated word string and the word string itself. We collect the most frequent usage for each person in the coprpus. In order to collect the prefered syntax, we annotated the word strings with syntactic dependency trees. Each of the dependency tress contains additional attributes, which describe the information content of a branch outgoing from the root as well as the possible value of the attriube at the nodes which carry the information. The learning program cuts the syntactic tree at edges starting at the root node and stores the branches in the referring expression model for the person. For instance, the complete referring expression model of a person would contain due to the training data the following information:

article: definite
lexicalization: person → man, light → white
syntax:
t21a: wearing glasses {t:hasGlasses a1:1 v1:glasses}
t21b: with $compl$ → beard {t:hasBeard a1:1 v1:beard} det → a
beard $compl$ → white {t:hairColour a1:light v1:white a2:dark v2:dark}
t22: with $compl$ → beard {t:hasBeard a1:1 v1:beard}
beard det → a
t23: wearing obj →glasses {t:hasGlasses a1:1 v1:glasses}
t26: with $compl$ → glasses {t:hasGlasses a1:1 v1:glasses}
glasses $coord$ → and $compl$ → heair mod →
dark{t:hairColour a1:dark v1:dark}

2.2 Setting up the Input for the Generator

One of the input attribute sets of the people domain looks like the following one:

```
<TRIAL CONDITION="-LOC" ID="s81t25">
  ...
    <ATTRIBUTE-SET>
    <ATTRIBUTE ID="a4" NAME="hasBeard" VALUE="1"/ >
    <ATTRIBUTE ID="a3" NAME="hairColour" VALUE="light"/ >
    <ATTRIBUTE ID="a2" NAME="hasGlasses" VALUE="1"/ >
    <ATTRIBUTE ID="a1" NAME="type" VALUE="person"/ >
  < /ATTRIBUTE-SET>
< /TRIAL>
```

We start to set up the input structure with the top node which is labeled with the lexicalization of the type or in seldom cases with *elision*, when the type is not in the attribute set. Then we look up in the referring expression model which determiner the person prefers. If she prefers any then a node is build, labeled with the determiner and connected with an edge to the type node. After that we add the lexicalized values of that attributes which are nearly always directly attached to the type node such as age in the people domain or colour and size in the furniture domain. Then the program searches in the model the syntactic annotations of attribute combinations. If IS-FP has build the referring expression then it starts to search in the trail selected by the nearest neighbour algorithm otherwise it calculates the closest due to the dice metric. In our example IS-FP might have build as well the given combination since it is equal to the attribute set of trail *s81t21*. Then the program would select the syntactic part t21b first and adapt the value of the node label *white* to *dark*. After that the the syntactic part t21a would be selected since the attribute *hasGlasses* is still not covered in the structure. This part does not need any adaption. Figure 1 shows the result of the process.

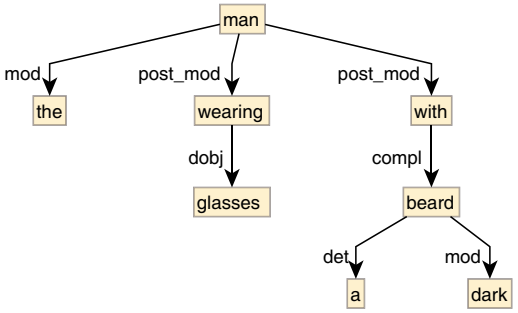

Figure 1: The input to the graph transducer

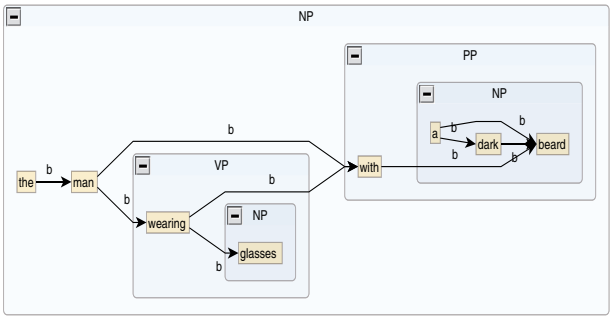

Figure 2: The output of the graph transducer

2.3 Realization of the Word String

For the realization, we use a handcrafted grammar that generates out of the dependency trees roughly

deep morphologic structure / topologic graph. The main task of the grammar is to determine the word order. The grammar contains four main rule groups. The vertical rules order the parent in relation to one of its dependent. The horizontal rules oder two childs. The constituent creation rules build constituents and the constituent adjoin rules adjoins constituents with constituents. Special consideration needed the order of prepositional constituents after the type and the adjective before the type. The prepostional constituents are order because of the order of the prepostions in the corpus. In order to be able to derive the order of the adjective, we used the functional class of the adjectives. Halliday (1994) proposes for English, the classes deictic (this, those, ...), numerative (many, second, , ...), epithet (old, blue, ...), and classifier (vegetarian, Spanish, ...). The order of the adjectives in a noun phrase is in the given order of the classes. In the lexicon entry of the adjectives, we store only a number between one and four which refers to the adjective class.

Table 5 shows the result for the TUNA-R task. The system was developed only by looking on the training data without any consideration of the development data as well without any annotation of the syntax of the development data. We used as guide for the optimization cross validation of training data.

Set	Accuracy	String-Edit Distance
Furniture	35 %	3,163
People	22,06 %	3,647
Total	28,53	3,405

Table 4: Results for the TUNA-R Task

3 IS-FP-GT: The Combination of Attribute Selection and Realization

The only change, we made in compare to IS-FP is that we switched off the feature to add the most similar referring expressions of another human from the training set for the nearest neighbour evaluation since the results have been lower. The reason for this is that other human preferes similar attributes but the individual preferences such as the chosen words and syntax of the other human is different. Table 5 shows the results.

4 Conclusion

The IS-FP algorithm reproduces the imprint of human referring expressions. It generates combinations such as the x-dimension and y-dimension. Elements of a combination have not to occur always together, however they tent to occur together. This is an advantage over incremental algorithms which might have to include other attributes ordered between elements of a combination. FP has the advantage over its predecessor FBN to generate expressions which are additionally mostly equal in respect to the length to human referring expressions, it enlarges automatically the training set for an individual human and it takes into account properties of the domain like the frequency of the attributes.

Set	Accuracy	String-Edit Distance
Furniture	15 %	3,8625
People	8,82 %	4,764
Total	11,91	4,313

Table 5: Results for the TUNA-REG Task

References

B. Bohnet and R. Dale. 2005. Viewing referring expression generation as search. In *IJCAI*, pages 1004–1009.

B. Bohnet, A. Langjahr, and L. Wanner. 2000. A Development Environment for an MTT-Based Sentence Generator. In *Proceedings of the First International Natural Language Generation Conference*.

B. Bohnet. 2006. *Textgenerierung durch Transduktion linguistischer Strukturen.* Ph.D. thesis, University Stuttgart.

B. Bohnet. 2007. IS-FBN, IS-FBS, IS-IAC: The Adaptation of Two Classic Algorithms for the Generation of Referring Expressions in order to Produce Expressions like Humans Do. In *MT Summit XI, UCNLG+MT*, pages 84–86.

R. Dale and E. Reiter. 1995. Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19:233–263.

M. A. K. Halliday. 1994. *An Introduction to Functional Grammar*. Edward Arnold, London.

F. Lareau and L. Wanner. 2007. Towards a Generic Multilingual Dependency Grammar for Text Generation. In *GEAF-07*, Palo Alto.

I.A. Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press, Albany.

S. Mille and L. Wanner. 2008. Making Text Resources Available to the Reader: The Case of Patent Claims. In *LREC*, Morocco, Marrakech.

L. Wanner, B. Bohnet, N. Bouayad-Agha, F. Lareau, A. Lohmeyer, and D. Nickla. 2007. On the Challenge of Creating and Communicating Air Quality Information. In *In: Swayne A., Hrebicek J.(Eds.): Environmental Software Systems Dimensions of Environmental Informatics. Vol.7.*