Efficient Algorithm for Context Sensitive Aggregation in Natural Language Generation

Hemanth Sagar Bayyarapu
Language Technologies Research Center
International Institute of Information Technology
Hyderabad, AP, India - 500032
hemanth.sagar@research.iiit.ac.in

Abstract

Aggregation is a sub-task of Natural Language Generation (NLG) that improves the conciseness and readability of the text outputted by NLG systems. Till date, approaches towards the aggregation task have been predominantly manual (manual analysis of domain specific corpus and development of rules). In this paper, a new algorithm for aggregation in NLG is proposed, that learns context sensitive aggregation rules from a parallel corpus of multi-sentential texts and their underlying semantic representations. Additionally, the algorithm accepts external constraints and interacts with the surface realizer to generate the best output. Experiments show that the proposed context sensitive probablistic aggregation algorithm performs better than the deterministic hand crafted aggregation rules.

1 Introduction

Aggregation is the process in which two or more linguistic structures are merged to form a single sentence. It helps in generating concise and fluent text and hence is an essential component in any NLG system (Reiter and Dale 2000). Figure 1(a) presents an example of de-aggregated text while Figure 1(b) shows its aggregated counterpart. Clearly, the aggregated text is fluent while the de-aggregated text is artificial with lot of redundancy.

Reiter (1994) proposed a consensus pipeline architecture for NLG systems with three stages:

- Content-Determination: Selects the information (propositions) to be conveyed and organizes the information in a rhetorically coherent manner.
- Sentence-Planning: Generates referring expressions, combines multiple propositions, selects appropriate lexical items and syntactic structures for each (aggregated) proposition and adds cohesion devices (eg, discourse markers) to make the text flow smoothly.
- Surface-Realizer: Converts the lexicalized linguistic structure into a linearized string while ensuring grammaticality, proper punctuation, correct morphology.

Bacteria are prokaryotic.
Bacteria are unicellular.
Bacteria have a cell wall.
Bacteria have DNA.
The shape of the DNA is circular.
The DNA is inside cytoplasm.

(a) De-aggregated text

Bacteria are prokaryotic and unicellular.
They have a cell wall, a plasma membrane and a circular DNA within cytoplasm.

(b) Aggregated text

Figure 1: Example showing de-aggregated text and its equivalent aggregated text.

The input to the process of aggregation, a submodule of Sentence-Planning in the consensus architecture described above, is a set of propositions selected by Content-Determination module which are organized using rhetorical relations between the propositions. Typical NLG systems use a two-stage aggregation process (Wilkinson, 1995). In the first stage, i.e., *semantic grouping*, the input set of propositions are partitioned into multiple sets, each of which is realized as a sentence. In the second stage, decisions related to actual realization of each set partition are taken.

Proceedings of Recent Advances in Natural Language Processing, pages 84–89,
Hissar, Bulgaria, 12-14 September 2011.

The essential idea behind semantic grouping is that the propositions that form a set and get realized as a meaningful sentence are related somehow. For example in Figure 1, the first two propositions (*Bacteria are unicellular. Bacteria are prokaryotic.*) are two assertive sentences about *Bacteria* and hence are aggregated. But it is not true that these two propositions will always be aggregated into a single sentence as shown in Figure 2.

Bacteria are unicellular while fungi can be either unicellular or multicellular.
Bacteria are prokaryotic and hence lack a cell nucleus.
On the other hand, fungi are eukaryotic and have a true cell nucleus bounded by a membrane.

Figure 2: Example answer from a corpus of QAs in Biology domain.

This shows that semantic grouping depends not only on the similarity between propositions, but also on the context (communicative goal of the text). The issue of context in semantic grouping gains importance especially in systems that present the same information in different views (Example: QA systems). For example, the two propositions (*Bacteria are unicellular. Bacteria are prokaryotic*) occur in examples shown in Figures 1 & 2. In the example in Figure 1, these propositions are aggregated while in the example in Figure 2 they are not. If we look at the context of these texts, the text in Figure 1 is *a short description about Bacteria*. On the other hand, the text in Figure 2 talks about *the fundamental difference between Bacteria and Fungi*.

The problem that is considered in this paper is as follows: Given a parallel corpus of multi-sentential texts and their underlying semantic representations along with the communicative goal of the text, can we learn semantic grouping rules automatically? The semantic representation assumed in this paper is a conceptual graph (Figure 3 shows an example of a conceptual graph), but the applicability of the approach is generic and can be customised to accomodate any semantic representation. A context-dependent discriminative model is learned which, given a proposition set and the context, estimates the probability of aggregation of the propositions. The problem of semantic grouping is modelled as a hypergraph partitioning problem that uses the probabilities outputted by the context-dependent discriminative model. To address the problem of hypergraph partioning, Multi-level Fiduccia-Mattheyses Framework (MLFM) is used (Karypis and Kumar, 1999).

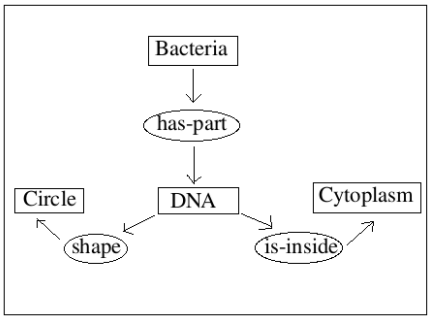

Figure 3: Example of a conceptual graph.

The approach is evaluated in the biology domain against two alternatives, namely handcrafted rules (HC) and a greedy clustering approach (GC) using the probabilities outputted by the context-dependent discriminative model. Additionally, we also test the impact of context by ignoring context while learning the discriminative model (Context-independent discriminative model).

An overview of related work is presented in Section 2. The corpus used in the experiments is discussed in Section 3. Then, in Section 4, the approach is discussed followed by Section 5 which presents the experiments done and their results. Finally, Section 6 concludes the paper with discussions and future work.

2 Related Work

Aggregation has been employed since the early NLG systems. In PROTEUS, a computer program that generates commentaries on a tic-tac-toe game, Davey (1979) used conjunctions to express SEQUENCE and CONTRASTIVE relations. Derr and McKeown (1984) showed how focus of attention helps in taking decisions related to choice between a sequence of simple sentences and a complex one. ANA (Kukich, 1983), used financial domain specific aggregation rules to generate complex sentences upto 34 words. Logical derivations were used to combine clauses and to remove easily inferrable clauses in (Mann and Moore, 1980).

Hand-crafted aggregation rules developed as a result of corpus analysis are employed by (Scott and de Souza, 1990; Hovy, 1990; Dalianis, 1999; Shaw, 1998). Walker et al. (2001) proposed a overgenerate-and-select approach in which the over-generate stage lists out large number of potential sentence plans while the ranking stage selects the top ranked sentence plan using rules that are learned automatically from the training data. Cheng and Mellish (2000) propose a genetic algorithm coupled with a preference function. Barzilay and Lapata (2006) view the problem of semantic grouping as a set partioning problem. They employ a local classifier that learns similarity between the propositions and then use ILP (Branch-and-bound algorithm) to infer a globally optimal partition.

This work is different from the earlier work in two aspects. We use contextual information to obtain better grouping that is applicable across different systems (even QA systems) while their work does not use the contextual information. Also, we assume a more generic hypergraph representation and use MLFM technique which works well even with large number of propositions.

What is the nucleolus?
The nucleolus is a small and optically dense region in the interior of the cell nucleus.
It is made of ribosomic RNA (rRNA) and proteins.

Context C = { Nucleolus }

Proposition set S = {

(Nucleolus is-region-of Cell-Nucleus)
(Nucleolus size small)
(Nucleolus density optically-dense)
(Nucleolus has-part RRNA)
(Nucleolus has-part Protein)

}

Figure 4: Example of a QA pair and its triple representation.

3 Corpus

A total of 717 QA pairs are collected from various sources in the biology domain. Concepts are extracted from the question which acts as contextual information. For example, when the question is *What is a binary fission?*, the concept *Binary-Fission* becomes the context. The answer is converted into sets of triples, each set corresponding to a sentence. Each triple consists of two concepts (or instances of concepts) connected by a relation. For example, the triple *(Mitosis next-event Cytokinesis)* contains two concepts namely *Mitosis* and *Cytokinesis* connected by the relation *next-event*. Figure 4 shows a QA pair and its triple representation. The context and sets of triples are extracted from each QA pair manually. The manual annotation process uses the component library described in (Barker et al., 2001).[1]

A total of 6337 triples are collected corresponding to 717 answers with each answer having 8.839 triples on an average. The highest number of triples for an answer is 46 while the lowest is 1. The total number of sentences in the answers is 1862, i.e., 2.596 sentences per an answer.

4 Approach

4.1 Hypergraphs

A hypergraph (H) is a generic graph wherein edges can connect any number of vertices and are called hyperedges. In other words, each edge is a set of vertices. It is formally represented by a pair (V,E) where V is the set of vertices and E is the set of hyperedges. Each edge $e_i \epsilon$ E has associated weight w_i. An edge with zero weight means that the the edge does not exist.

4.2 k-way Hypergraph Partitioning problem

Let P be a k-tuple (p_0 , p_1, p_2...) where each p_i is a set of vertices from V such that $\cap_{i=0}^{i=k-1} p_i = \phi$ and $\cup_{i=0}^{i=k-1} p_i = V$. The k-way Hypergraph partitioning problem can be formulated as follows:

> Given a hypergraph H = (V,E), find a k-way partitionment $\delta : V \rightarrow P$ that maps each of the vertices of H to one of the k disjoint partitions such that some cost function $\gamma : P \rightarrow \mathbb{R}$ is minimized.

4.3 Modelling Aggregation as hypergraph partitioning problem

Relationships among the propositions are often complex than pairwise. Assuming this complex relationship as pairwise ones reduces the fluency

[1]The component library is available online at http://www.cs.utexas.edu/ mfkb/RKF/tree/

of the verbalized text in some cases. To deal with this complex relationship, it is better to directly use hypergraphs instead of pair-wise approximation.

We view the problem of aggregation as a hypergraph partioning problem guided by a data-driven context sensitive discriminative model. The input to the algorithm is a *conceptual graph* which can be alternatively represented as a set of propositions. The goal is to find optimal partitions of the set of propositions given context, where each partition represents an aggregated sentence. The set of propositions is viewed as a graph where each proposition represents a vertex as shown in Figure 5. Hyperedges are constructed on the graph obtained from propositions. Each hyperedge of this hypergraph connects one or more propositions. The weight w_i of each hyperedge is given by the context sensitive discriminative model discussed in section 4.4. The hypergraph along with edge weights is the input to the multi-level k-partitioning algorithm.

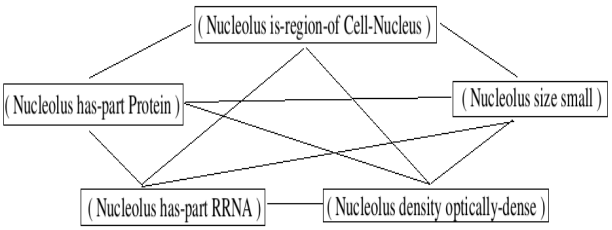

Figure 5: Example of a proposition set and its view as a graph

4.4 Context Sensitive Discriminative Model

The weight w_{iA} of a hyperedge (*A*) in the hypergraph formed from the inputs (*S*) is the probability of aggregation of propositions in *A* given contextual information (*C*) and *S*.

$$w_{iA} = p_A = P(A|C, S) \quad (1)$$

The contextual information include the communicative goal (the concepts in the question) The features that are used to predict the probability of aggregation of a proposition set are based on:

- *Cohesion of the proposition set* is the average score of similarities between each pair of propositions in A:

$$Coh_A = \frac{\sum_{i=1,j=1,i\neq j}^{i=|A|,j=|A|} sim(A_i, A_j)}{|A|} \quad (2)$$

  The similarity between each pair is the number of matches in the components of triples. For example, since the triples *(Mitosis subevent Prophase), (Mitosis subevent Anaphase)* match in two slots, the similarity score is 2/3.

- *Complexity of the realization* is a cumulative weighted score of number of words, number of relative clauses, number of connectives, etc. and this score depicts how difficult it is to interpret the sentence corresponding to the proposition set A (if it is generated using the surface realizer). The score value is ∞ if the propositions cannot be realized as a single sentence because the surface realizer cannot find suitable structure that accomodates all the propositions.

- *Dissimilarity with rest of the propositions* calculates how dissimilar the proposition set A is with the rest of the propositions (S-A). The maximum distance (or minimum similarity) of each proposition in S-A from A is calculated and averaged.

- *Similarity with context C* is the score of the extent of the cover of context by the triples. It is the ratio of number of concepts in the context C that occur in any of the triples in A to the total number of concepts in C.

A number of boolean features and their conjunctive features are generated using the above scores with score bounds. Such feature structures are generated for each hyperedge in the hypergraph formed from S. All the subsets of S which are in Z (the correct partitioning of S) are positive instances and rest are negative instances. A maximum entropy model is employed to predict the probabilities of aggregation of a set of propositions.

While using the maximum entropy model to predict the aggregation probability, we can also utilize pattern matching rules to group propositions as a pre-processing step. The pattern matching rules can include domain specific rules, inference rules, etc. The motivations for this grouping are: (1) propositions are a mere representation of complex texts, (2) when the number of propositons is very high, optimization on the level of propositions becomes intractable.

Any constraint on the output can be expressed as features in the discriminative model. Transitivity constraint on set of propositions is automatically captured in the usage of hyperedges. External constraints like complexity of sentence is expressed in the features of the discriminative model (Complexity of the realization).

5 Experiments

We use a n-fold cross validation on the corpus described in section II. We use two baselines for comparison: (1) Hand-Crafted rules (HC) and (2) Greedy clustering of hypergraph (GC). Hand-crafted rules are pattern matching rules on sets of propositions. An example rule is to aggregate two triples if they share atleast two slots. In the second baseline, i.e., the greedy clustering of hypergraph, the graph is clustered using the probability scores of hyperedges based on the context sensitive model. The top scoring hyperedges that are non-overlapping and cover the entire input set are outputted. Also, in order to test the impact of context, we build a context independent discriminative model but follow the same hypergraph partitioning approach (HGP).

5.1 Evaluation metrics

Let Y be the output partition of our approach and Z be the correct partitioning which is annotated manually. We use the following evaluation metrics:

- Precision: the ratio of correct pair-wise aggregations in Y and total pair-wise aggregations in Y
- Recall : the ratio of correct pair-wise aggregations in Y and total pair-wise aggregations in Z
- F-score: the harmonic mean of Precision and Recall

5.2 Results

The results are shown in Table 1. All the scores are average scores on a 5-fold cross validation. Hand-Crafted rules performed very poor because there are very few rules covering aggregation of more than five propositions while the corpus consisted of many such proposition sets. The effect of context is clear as the context dependent (HGPC) model outperforms context independent model by 7.15%. This proves that the usage of context is very important if the model has to be generic and adaptable to any kind of NLG system.

Model	Recall	Precision	F-Score
HC	32.5	21.6	25.9
GC	41.7	47.5	44.4
HGP	40.02	58.8	47.6
HGPC	49.6	61.1	54.75

Table 1: Results on pairwise aggregations; Comparison between Hand-Crafted rules (HC), Greedy clustering (GC), Hyper-graph partitioning model with context (HGPC) and without context (HGP)

6 Conclusions

The number of propositions in an answer in our corpus varied from 1 to as large as 46. We used an empirically proven scalable partitioning framework that works well when the number of propositions is huge. We presented a novel context sensitive aggregation algorithm for NLG systems. Also we presented a much natural hypergraph approach to semantic grouping than other methods that approximate the complicated relationships (among the entities that are checked for aggregation) with pair-wise approximations. The approach is adaptable to any domain and any representation. With a small corpus of 717 QA pairs, good results are obtained over the hand-crafted approaches.

In our future work, we would like to test the described approach for scalability. The MLFM technique used in this work is proven to be the best technique for partitioning a set of more than 200 propositions. Also, the evaluations in this paper have been conducted in partial isolation from the actual output of the surface realizer. In our future work, we would also like to consider the impact of aggregation on the final textual outputs.

References

Barker, Ken , Bruce Porter, and Peter Clark. 2001 A library of generic concepts for composing knowledge bases. In Proceedings of First International Conference on Knowledge Capture, 2001.

Barzilay, Regina and Mirella Lapata. 2006. Aggregation via Set Partitioning for Natural Language Generation. In Proc. of NAACL/HLT, 2006.

Cheng, Hua and Chris Mellish. 2000. Capturing the interaction between aggregation and text planning in

two generation systems. In Proceedings of INLG-2000, Israel.

Dalianis, Hercules. 1999. Aggregation in Natural Language Generation. Journal of Computational Intelligence, Volume 15, Number 4, pp 384-414, November 1999. Abstract

Davey, Anthony C. 1979. Discourse Production. Edinburgh University Press, Edinburgh.

Derr, Marcia A. and Kathleen R. McKeown. 1984. Using focus to generate complex and simple sentences. In Proceedings of the Tenth International Conference on Computational Linguistics (COLING-84) and the 22nd Annual Meeting of the ACL, pages 319326, Stanford University, Stanford, CA.

Donia R. Scott and Clarisse S. de Souza. 1990. Getting the message across in RST-based text generation. In R. Dale, C. Mellish, and M. Zock, editors, Current Research in Natural Language Generation

Hovy, Eduard H. 1990. Unresolved issues in paragraph planning. In R. Dale, C. Mellish, M. Zock, eds., Current Research in Natural Language Generation, 1741. Academic Press, New York.

Karypis, G. and V. Kumar. 1999. Multilevel k-way hypergraph partitioning. In Proceedings of the Design and Automation Conference, 1999.

Kukich, Karen. 1983. Design of a knowledge-based report generator. In Proceedings of the 21st Annual Meeting of the ACL, pages 145150, Cambridge, MA, June 15-17,.

Mann, William C. and James A. Moore. 1980. Computer as author results and prospects. Technical Report RR-79-82, USC Information Science Institute, Marina del Rey, CA.

Reiter, Ehud and Robert Dale. 2000 Building Natural Language Generation Systems. Cambridge University Press, Cambridge.

Reiter, Ehud. 1994 Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? In Proceedings of the Seventh International Workshop on Natural Language Generation, pages 163170, Nonantum Inn, Kennebunkport, Maine.

Shaw, James. 1998. Clause aggregation using linguistic knowledge. In Proceedings of the 9th International Workshop on Natural Language Generation., pages 138147.

Wilkinson, John. 1995. Aggregation in natural language generation: Another look. Co-op work term report, Department of Computer Science, University of Waterloo, Septembe

Walker, Marilyn, Owen Rambow and Monica Rogati. 2001. Spot: A trainable sentence planner. In Proceedings of the second annual meeting of North American Chapter of Association for Computational Linguistics, 17-24, Pittsburgh, PA