Predicting sentential semantic compatibility for aggregation in text-to-text generation

Victor Chenal
School of Computer Science
McGill University
victor.chenal@mail.mcgill.ca

Jackie Chi Kit Cheung
School of Computer Science
McGill University
jcheung@cs.mcgill.ca

Abstract

We examine the task of aggregation in the context of text-to-text generation. We introduce a new aggregation task which frames the process as grouping input sentence fragments into clusters that are to be expressed as a single output sentence. We extract datasets for this task from a corpus using an automatic extraction process. Based on the results of a user study, we develop two gold-standard clusterings and corresponding evaluation methods for each dataset. We present a hierarchical clustering framework for predicting aggregation decisions on this task, which outperforms several baselines and can serve as a reference in future work.

1 Introduction

In text-to-text generation, existing sentence compression and sentence fusion systems assume that the input takes the form of one or more sentences, and the output is one suitably modified sentence (Barzilay et al., 1999; Knight and Marcu, 2000; Marsi and Krahmer, 2005; Filippova, 2010; Thadani and McKeown, 2013, for example). This line of work has led to new, knowledge-lean methods for sentence generation for applications such as text simplification and automatic summarization.

The next step in expanding the scope of text-to-text generation is to relax assumptions about the forms of the input and output, with the eventual goal of generating entire passages or documents in an abstractive manner. We focus in this paper on aggregation, the task of determining what input units belong in the same output sentence. Aggregation is an important step in micro-planning in the traditional data-to-text NLG pipeline (Reiter et al., 2000). However, it has been little examined within the context of text-to-text generation outside of restricted syntactic contexts, such as entity-driven noun phrase rewriting (Nenkova, 2008).

In this paper, we propose a new aggregation task suited for text-to-text generation. The goal of the task is to aggregate content into sentence-sized units by taking sentence fragments (i.e., meaningful bits of text) as input, and outputting clusters of these fragments. Figure 1 shows an example of such a clustering. It can be viewed as a preceding step for sentence fusion or other text-to-text generation methods.

Inspired by surface realization tasks (Belz et al., 2011), we use pre-existing sentences as a source of data in order to cheaply and automatically generate datasets with different granularities. We conduct a user study to confirm the validity of these datasets and design two gold-standard clusterings. We use these gold standards to define two methods of evaluation, and introduce two baselines for the task.

Then, we propose a simple clustering model for this task based on logistic regression and hierarchical clustering. All variants of the model are shown to outperform both baselines on our datasets, but an analysis of the results identifies shortcomings in the clustering that could be overcome in future models.

2 Related work

Microplanning, or sentence planning, encompasses the tasks of natural language generation (NLG) that occur after the general organization of arguments and before the syntactic realization of phrases (Hovy

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1061–1070, Osaka, Japan, December 11-17 2016.

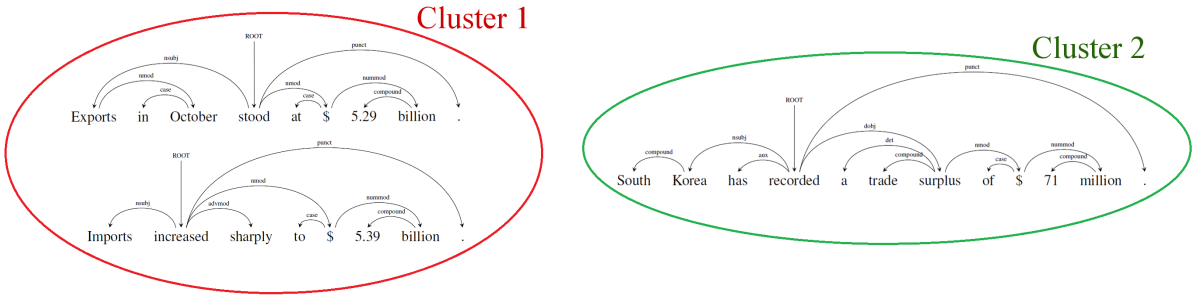


Figure 1: An example of clustering 3 sentence fragments into 2 clusters

and Wanner, 1996). Traditionally, a microplanner uses abstract domain-specific representations to produce linguistic representations. These representations are often used in domain-specific or user-specific generation systems (Mellish et al., 1998; Langkilde, 2000; Walker et al., 2002; Stent et al., 2002) and need to be adapted when changing domains (Stent et al., 2004; Walker et al., 2007). Previous work considered sentence planning to be a rigid succession of three distinct tasks: lexical choice, aggregation and referring expression generation (Reiter et al., 2000). However, more recent approaches have integrated them into a joint system (Angeli et al., 2010; Konstas and Lapata, 2013; Kondadadi et al., 2013).

Microplanning has not been considered relevant as an independent step in most recent work on text-to-text generation, because many microplanning decisions can be directly derived from the input text. For example, sentence fusion and sentence compression systems rely directly on surface text and make assumptions about the similarity of input sentences with limited rewording. Sentence compression, which consists of taking one sentence and removing redundant parts has been heavily studied (Knight and Marcu, 2000; McDonald, 2006; Galley and McKeown, 2007; Cohn and Lapata, 2008; Clarke and Lapata, 2008). Similarly, a large body of work exists in sentence fusion (Barzilay et al., 1999; Barzilay and McKeown, 2005; Marsi and Krahmer, 2005; Filippova, 2010; Thadani and McKeown, 2013).

However, relying so heavily on the input text has limited the scope of text-to-text systems. In terms of aggregation, most current systems in sentence fusion focus on fusing very similar input sentences, as determined by lexical overlap, though several recent approaches expand fusion to more disparate ones (Elsner and Santhanam, 2011; Cheung and Penn, 2014). Our work in this paper can be seen as a systematic investigation to determine what content is semantically compatible at a sentence-level, in order to generate inputs to such systems.

We focus on predicting in this paper on *what* content should be expressed in the same sentence. Other work has examined the issue of *how* the content should be expressed syntactically, such as by applying hand-crafted rules (Pan and Shaw, 2004). White and Howcroft (2015), for example, show that rules for aggregation for clause combination can automatically be learned.

3 Clustering task

3.1 Task definition

The task consists of clustering sentence fragments (see next subsection) into clusters of fragments that can be merged into a single sentence. Figure 1 shows an example of clustering. In this instance, cluster 1 fragments can be combined into the sentence "*Exports in October stood at $5.29 billion while imports increased sharply to $5.39 billion.*". This particular resulting sentence is an example and others can be generated from the same cluster.

The structure of our task can be defined as follows. Given a set of N sentence fragments $\mathbf{x} = \{x_1, x_2, ..., x_N\}$, we compute $\mathbf{y} = \{y_1, y_2, ..., y_k\}$ with $y_i \subseteq \mathbf{x}$, a partitioning of $\mathbf{x}$ into k clusters. The performance of the clustering $\mathbf{y}$ can then be measured by comparing it to a partitioning $\mathbf{c} = \{c_1, c_2, ..., c_m\}$ of $\mathbf{x}$ into m gold-standard clusters (see Section 5.2).

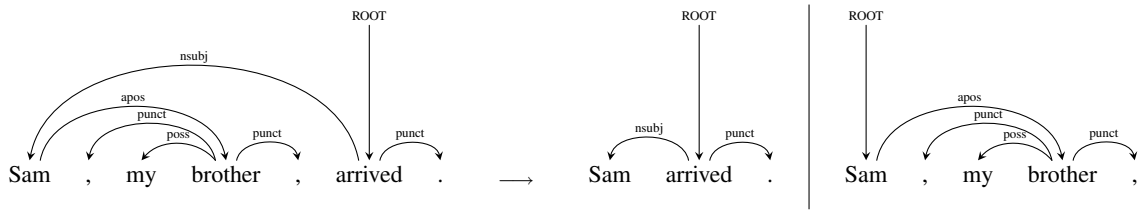


Figure 2: Example of two fragments (right) being extracted from one sentence (left)

3.2 Sentence fragment definition

Sentence fragments are defined as parts of sentences that are semantically consistent and that could be considered as grammatical sentences on their own provided small adjustments such as modifying determiners or punctuation. Sentence fragments are therefore very close to clauses, and a sentence in a text can contain one or multiple fragments. For example, the sentence *"Barack Obama, president of the United States, was born in Hawaii"* contains two fragments: *"Barack Obama (is) president of the United States"* and *"Barack Obama was born in Hawaii"*.

Fragments, as opposed to clauses, can have different levels of granularity while retaining their defining properties. For instance, the sentence *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products"* can be split into two fragments: *"Bell (is) based in Los Angeles"* and *Bell makes and distributes electronic, computer and building products*. The latter fragment could then be further divided into *"Bell makes electronic, computer and building products"* and *"Bell distributes electronic, computer and building products"*. Similarly, these two fragments could then be divided into three subfragments each by removing the conjunction *"and"*.

4 Dataset generation

4.1 Splitting sentences

Sentence fragments are extracted by splitting corpus sentences on specific dependencies. The subtree resulting from such a dependency is extracted and the head word (or compound words) are copied to the new fragment. Figure 2 shows an example of extraction[1].

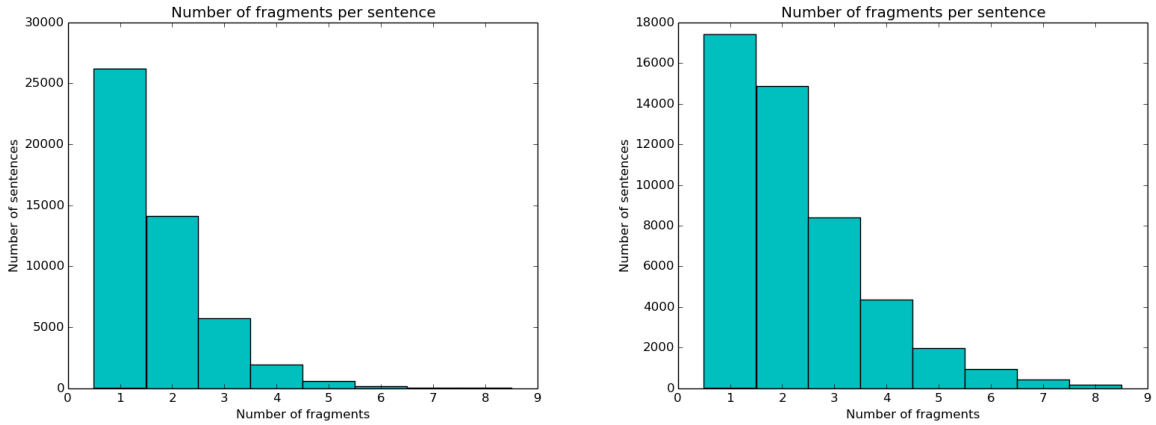
Several types of dependencies can be used to extract such fragments: appositional modifiers (e.g. *"Sam, my brother, arrived"*), adjectival and verbal modifiers (e.g. *"Pierre Vinken, 61 years old,..."*) and relative clauses (e.g. *"South Korea's economic boom, which began in 1986,..."*). To ensure that adjectival and verbal modifiers could be considered as a sentence on their own once extracted, only the subtrees with size at least four are extracted. These rules were designed to make fragments similar to propositions and ensure that they can be interpreted as sentences.

Previous examples show that conjunctions can also be used to extract fragments (see Section3.2). However, extracting conjunctions involves a different splitting mechanism: a fragment is not removed from the original sentence but the sentence is rather copied in two separate instances corresponding to each element from the conjunction (e.g. *"they either ski or snowboard"* is split into *they ski* and *they snowboard*). This different mechanism and the imperfections in the dependency trees make splitting on conjunctions a complex task and can lead to fragments that are not consistent. As a result, we generate two datasets of different granularities from the original corpus: the first one does not involve any action on conjunctions while the second one contains fragments extracted from conjunctions. These datasets will be referred to as *KeepConj* and *SplitConj*, respectively.

4.2 Creation of the datasets

We used the Penn Treebank Wall Street Journal corpus (Marcus et al., 1993). Manually-annotated constituent trees were converted into dependency trees using the the Stanford CoreNLP framework (Manning

[1]Technically, in this example, the first extracted fragment should be post-processed from "*Sam, my brother,*" to "*Sam is my brother*" (and the corresponding dependency tree) to represent a new individual sentence. However, all features used in our models (see Section 6.1) are indifferent to these modifications so the extracted fragments are not modified.



(a) KeepConj

(b) SplitConj

Figure 3: Distribution of the number of fragments per sentence

et al., 2014). Two datasets were created by extracting fragments from all sentences in all documents following the process described in the previous subsection. Each dataset was divided into a training set containing 2000 documents, a development set containing 254 documents and a test set containing 200 documents.

From the 48810 original sentences in the corpus, 84051 fragments were extracted into the *KeepConj* dataset and 111593 fragments were extracted into the *SplitConj* dataset.

Figure 3 shows that the generated datasets contain a lot of singletons, i.e. sentences from which one fragment has been extracted. The less fine-grained the dataset is, the more pronounced the imbalance. In our later experiments, we will define a singleton-only baseline which will form a non-trivial baseline to beat due to this distribution (see Section 7.1).

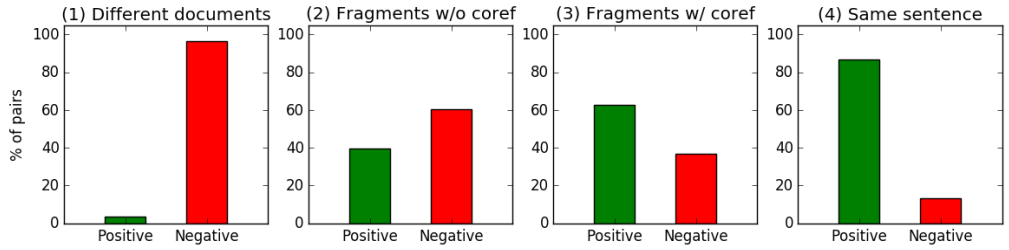
5 Establishing the Gold Standard

5.1 Validation by user study

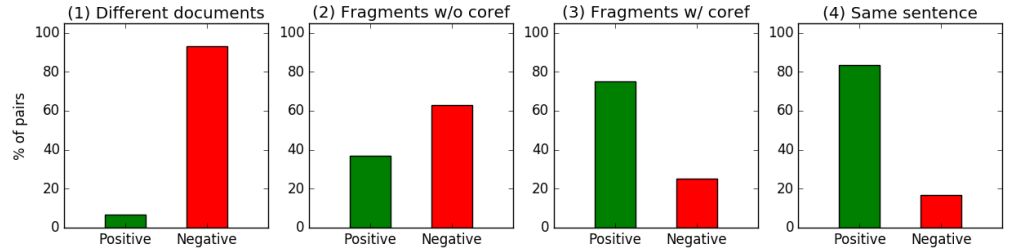
In order to validate our automatic dataset extraction process (i.e. ensuring that fragments extracted from a given sentence could be combined back) and establish a gold-standard clustering to compare the results of the task with, we conducted a user study. Participants, all native English speakers, were presented with pairs of sentence fragments and were asked to decide whether the fragments were mergeable (positive pair) or not (negative pair). Mergeable fragments are defined as fragments that can be combined into a single consistent and grammatical sentence without having a significant impact on their structure. Participants were shown detailed instructions describing the task, the allowed mechanisms for merging fragments as well as examples of positive and negative cases. Each participant was presented with 60 pairs of fragments: 10 pairs originated from the same original sentence, 10 pairs originated from different documents, 20 pairs originated from different sentences within a same paragraph and 20 pairs originated from different paragraphs within a same document. Three separate annotations were recorded for each form of 60 pairs of fragments. A total of 18 participants annotated 360 pairs, 180 pairs per dataset.

The results of the study are shown in Figure 4. The study confirms the validity of the automatic dataset generation process: fragments coming from the same original sentence are mostly considered mergeable by human annotators. A qualitative analysis of the cases when such pairs were labelled as negative show that they either come from an error in the fragment extraction process (that can come from the conversion from constituent trees to dependency trees) or a situation when three or more fragments were extracted from a sentence.

Inter-agreement between participants was measured using Fleiss' kappa (Fleiss, 1971). Agreement is high when fragments come from the same sentence or when they come from different documents. These categories correspond to "easy" cases, respectively positive and negative most of the time. On the other



(a) KeepConj



(b) SplitConj

Figure 4: Results of the user study. Pairs are divided into four categories by source: (1) fragments that come from different documents; (2) fragments that come from different sentences within a same document and that do not contain coreferent entity mentions; (3) fragments that come from different sentences within a same document and that contain coreferent entity mentions; (4) fragments that come from the same sentence.

Pairs considered	Fleiss' κ
All pairs	0.401
(1) Different documents	0.641
(2) Fragments w/o coref	0.261
(3) Fragments w/ coref	0.252
(4) Same sentence	0.731

(a) KeepConj

Pairs considered	Fleiss' κ
All pairs	0.461
(1) Different documents	0.636
(2) Fragments w/o coref	0.358
(3) Fragments w/ coref	0.317
(4) Same sentence	0.727

(b) SplitConj

Table 1: Inter-agreement among human annotators. Pairs are divided into four categories by source, as seen in Figure 4.

hand, agreement in other instances is lower, suggesting that deciding if two fragments are mergeable is a difficult task. However, results show that fragments that contain coreferent entity mentions are more likely to be mergeable (see Figure 4). This result is consistent across datasets.

The study shows that although fragments extracted from the same sentence are indeed mergeable, a significant number of positive cases exist where fragments originate from different sentences within a given document. Most of these cases correspond to fragments that contain coreferent entity mentions, yet not all of these fragments are mergeable. A chi-squared test showed that for both datasets, the proportion of positive pairs among fragments containing coreferent entity mentions (see category (3) in Figure 4) is significantly greater ($p < 0.05$) than the proportion of positive pairs among fragments that originated from the same document but do not contain such mentions (see category (2) in Figure 4). Based on these observations, we will define two different gold-standard clusterings: one optimistic clustering and one pessimistic clustering.

5.2 Evaluation measures

Based on the results from the user study, we define two gold-standards that act as lower and upper bounds on the clustering performance. The first one consists of using the original sentences from the corpus as gold-standard. This means that fragments should be clustered together if and only if they were extracted

from the same sentence. The second gold-standard consists of using fragments that contain coreferent entity mentions in addition to the original sentences. This means that fragments should be clustered together if and only if they were extracted from the same sentence or if they that contain coreferent entity mentions. The first gold-standard evaluation acts as a lower bound on the performance: it restricts the gold-standard to one specific clustering corresponding to the original corpus. On the other hand, the second gold-standard evaluation acts as an upper-bound on the performance: clustering of fragments that contain coreferent entity mentions is always considered to be positive while the user study showed that this is an oversimplification.

To evaluate the performance of a system clustering against one of the gold-standards, the measures of purity and collocation are used. Consider items $x_1, ..., x_N$ to be clustered, $y_1, ..., y_k$ the k system clusters outputted by the algorithm and $c_1, ..., c_m$ the m gold-standard clusters. Purity measures the proportion of items that belong to the same gold-standard cluster in a system cluster, while collocation measures the proportion of items that belong to the same system cluster in a gold-standard cluster:

$$\textit{Purity} = \frac{1}{N}\sum_{j=1}^{k}\max_{i\in[\![1,m]\!]}|y_j \cap c_i| \qquad \textit{Collocation} = \frac{1}{N}\sum_{j=1}^{m}\max_{i\in[\![1,k]\!]}|c_j \cap y_i|$$

When all items x_i are clustered by the system as singletons we have $\textit{Purity} = 1$, while we have $\textit{Collocation} = 1$ when all items x_i are clustered by the system as one cluster. To evaluate the trade-off between purity (that favours a large number of clusters) and collocation (that favours a small number of clusters), we use the $F1$ measure: $\textit{F1} = \frac{2\cdot\textit{Purity}\cdot\textit{Collocation}}{\textit{Purity}+\textit{Collocation}}$.

The mergeability relation in the second gold-standard is not considered to be transitive (i.e. if fragments A and B contain coreferent entity mentions and fragment C originates from the same sentence as fragment B, then fragments A and C should not be clustered together without fragment B); therefore, the second gold-standard only has an impact on purity, and not collocation. Specifically, when computing purity, a gold-standard cluster is the union of the fragments originating from a corpus sentence and all fragments that contain coreferent entity mentions with said sentence. However, when computing collocation, a gold-standard cluster is only the union of the fragments originating from a corpus sentence. This ensures that clustering the fragments back into the original corpus has a collocation score of 1.

We will refer to the metrics associated with the first gold-standard as "$\textit{Metric}^{\textit{LOW}}$" (e.g. $\textit{Purity}^{\textit{LOW}}$) and the metrics associated with the second gold-standard as "$\textit{Metric}^{\textit{UP}}$" (e.g. $\textit{Purity}^{\textit{UP}}$).

6 Hierarchical clustering model

We design a model that uses a logistic regression classifier to learn a similarity measure between two fragments. Based on this similarity function, we perform different methods of hierarchical clustering.

6.1 Learning of the similarity measure

Given a pair of fragments, we train a logistic regression classifier that returns the probability that the pair is positive, which we interpret as the similarity ϕ between the two fragments.

Given two items, x_i and x_j, we compute their joint feature representation $\Psi(x_i, x_j)$. We used a total of 27 pairwise features, detailed in Table 2.

For each document in the training set, each possible positive pair of fragments, i.e. its joint feature representation, is fed to the classifier with a positive label and a number of negative pairs less or equal than the number of positive pairs is fed to the classifier with a negative label. The number of negative cases considered per document is limited to account for the high number of singletons in the corpus and reduce training time.

After training, the similarity function ϕ is defined as $\phi(x_i, x_j) = \sigma_w(\Psi(x_i, x_j))$ where σ_w is the logit function learned by the classifier.

6.2 Variants of the model

Given a set of items $x_1, ..., x_N$ to be clustered, we compute the $N \times N$ similarity matrix K such that $K_{ij} = \phi(x_i, x_j)$. Each line (or column) of K corresponds to a cluster. We then perform bottom-up

Category	Example of feature	Value in example (see Figure 2)
Words in common	number of words in common	1 ("*Sam*")
	weight (*tf-idf*) of words in common	0
	weight (*tf-idf*) of nouns in common	0
	share of common words in longer fragment	0.33
Verb presence	share of fragments containing a verb other than "be"	0.5
Length	difference in length between fragments	1
Semantic roots	share of roots that are a verb other than "be"	0.5
	same root	0

Table 2: Examples of pairwise features used to learn the similarity function ϕ between the fragments "Sam arrived" and "Sam, my brother"

hierarchical clustering on K: we start with each item as a singleton cluster and successively merge clusters that have the highest similarity (clusters corresponding to the line and column of the highest value in K). We stop when a threshold similarity value has been reached. Values in K are updated after each step using one of these methods:

- **Single** linkage: the similarity between two clusters is the maximum similarity between two elements of these clusters
- **Complete** linkage: the similarity between two clusters is the minimum similarity between two elements of these clusters
- **Average** linkage: the similarity between two clusters is the average similarity between two elements of these clusters

In addition to these static methods, we also implement a dynamic **iterative** clustering method: after each step, we recompute new similarity scores by concatenating the fragments in each cluster.

7 Experiments and analysis

7.1 Baselines

We implement two baselines to compare the results of our model with. These baselines rely on specific aspects of the corpus and the fragment extraction process.
SINGLETON: Due to the large number of sentences containing only one fragment in the original corpus, the first implemented baseline consists of clustering each fragment into a separate cluster. This singleton baseline performs particularly well on coarse-grained dataset *KeepConj*.
SAMEROOT: We implemented a second baseline which consists of clustering together fragments that have the same semantic root (node(s) in the dependency tree that have no governor). This "same root" baseline performs well on both datasets.

7.2 Results

Table 3 shows the comparisons between the performances of the baselines and four variants of the hierarchical clustering model. The stopping threshold for each model has been tuned on the development set.

Results show that all variants significantly[2] outperform the baselines on both datasets ($p < 0.01$). The hierarchical clustering method has an inconsistent impact across datasets. The dynamic iterative aggregation performs best on dataset *KeepConj* ($p < 0.05$) while the average-link and complete-link aggregations perform better than other methods on dataset *SplitConj* ($p < 0.05$).

[2]All significance tests were performed using a two-tailed paired sample t-test on F1-scores at the document level.

Model	$Purity^{LOW}$	$Collocation^{LOW}$	$F1^{LOW}$	$Purity^{UP}$	$Collocation^{UP}$	$F1^{UP}$
Singleton baseline	**1.00**	0.588	0.741	**1.00**	0.588	0.741
"Same root" baseline	0.829	0.651	0.729	0.876	0.651	0.747
Single-link	0.900	0.736	0.801	0.961	0.736	0.833
Complete-link	0.895	0.747	0.814	0.953	0.747	0.837
Average-link	0.893	0.754	0.818	0.956	0.754	0.843
Iterative	0.881	**0.778**	**0.827**	0.948	**0.778**	**0.855**

(a) KeepConj

Model	$Purity^{LOW}$	$Collocation^{LOW}$	$F1^{LOW}$	$Purity^{UP}$	$Collocation^{UP}$	$F1^{UP}$
Singleton baseline	**1.00**	0.446	0.617	**1.00**	0.446	0.617
"Same root" baseline	0.841	0.683	0.754	0.884	0.683	0.770
Single-link	0.868	**0.745**	0.802	0.940	**0.745**	0.831
Complete-link	0.920	0.723	0.810	0.970	0.723	0.829
Average-link	0.909	0.733	**0.811**	0.966	0.733	**0.833**
Iterative	0.894	0.731	0.804	0.961	0.731	0.830

(b) SplitConj

Table 3: Evaluation results of the baselines and variants of the model against both gold-standard clusterings

7.3 Analysis

Results show that our model consistently favours purity over collocation, generating a higher number of clusters compared to the gold-standards. Examples in this section were generated by the *iterative* hierarchical clustering model on dataset *KeepConj* but are representative of the behaviour of all variants of the model.

This higher number is mainly due to the fact that our model fails to cluster together some long fragments that share a small number of common words. For example, the sentence "*Influential members of the House Ways and Means Committee introduced legislation that would restrict how the new savings-and-loan bailout agency can raise capital, creating another potential obstacle to the government's sale of sick thrifts.*" is divided during extraction into the fragments "*Influential members of the House Ways and Means Committee introduced legislation*" and "*legislation would restrict how the new savings-and-loan bailout agency can raise capital, creating another potential obstacle to the government's sale of sick thrifts*". These fragments are then rendered as singletons by the model.

On the other hand, shorter fragments are more easily clustered. For instance, fragments "*Another $20 billion would be raised through Treasury bonds*" and "*Treasury bonds pay lower interest rates*" are clustered back into the sentence "*Another $20 billion would be raised through Treasury bonds, which pay lower interest rates.*".

Our model clusters also show instances of coreferent fragments clustering, as evidenced by the significant increase in purity when the second gold-standard is used ($Purity^{UP}$). Fragments "*thrifts are sold, until the assets can be sold separately*" and "*the assets the RTC holds*" are clustered together. While they originate from different sentences, the fragments contain coreferent entity mentions ("*the assets*") and can be merged into the sentence "*Thrifts are sold, until the assets the RTC holds can be sold separately.*".

8 Conclusion

This paper introduced an aggregation task suited for text-to-text generation based on clustering sentence fragments. We presented evaluation metrics that were designed using human-annotated results. These results show that most mergeable pairs of fragments either originate from the same original sentence or contain coreferent entity mentions. We designed a hierarchical clustering model that uses a logistic regression classifier to learn a similarity measure between fragments. Its results were shown to outper-

form simple baselines and can be used as a reference for the task. Given that the task relies on shallow semantic analysis and the evaluation metrics only require coreference resolution, it is highly adaptable. Generating datasets from multi-document corpora can be the object of future work. Other clustering models, using neural networks to learn a similarity measure for instance, can also be considered. Integration of other aspects that were considered outside of the scope of our task, for example discourse relations analysis or information aggregation is another possible future avenue.

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512. Association for Computational Linguistics.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Regina Barzilay, Kathleen R McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 550–557. Association for Computational Linguistics.

Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European workshop on natural language generation*, pages 217–226. Association for Computational Linguistics.

Jackie Chi Kit Cheung and Gerald Penn. 2014. Unsupervised sentence enhancement for automatic summarization. In *EMNLP*, pages 775–786.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 137–144. Association for Computational Linguistics.

Micha Elsner and Deepak Santhanam. 2011. Learning to fuse disparate sentences. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 54–63. Association for Computational Linguistics.

Katja Filippova. 2010. Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378–382.

Michel Galley and Kathleen McKeown. 2007. Lexicalized markov grammars for sentence compression. In *HLT-NAACL*, pages 180–187.

Eduard H Hovy and Leo Wanner. 1996. Managing sentence planning requirements. In *Proceedings, ECAI-96 Workshop on Gaps and Bridges: New Directions in Planning and Natural Language Generation*, pages 53–58.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI*, 2000:703–710.

Ravi Kondadadi, Blake Howald, and Frank Schilder. 2013. A statistical nlg framework for aggregated planning and realization. In *Proceedings of the 51st annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 1406–1415. Association for Computational Linguistics.

Ioannis Konstas and Mirella Lapata. 2013. A global model for concept-to-text generation. *J. Artif. Intell. Res.(JAIR)*, 48:305–346.

Irene Langkilde. 2000. Forest-based statistical sentence generation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 170–177. Association for Computational Linguistics.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Erwin Marsi and Emiel Krahmer. 2005. Explorations in sentence fusion. In *Proceedings of the European Workshop on Natural Language Generation*, pages 109–117. Citeseer.

Ryan T McDonald. 2006. Discriminative sentence compression with soft syntactic evidence. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Chris Mellish, Alistair Knott, Jon Oberlander, and Mick ODonnell. 1998. Experiments using stochastic search for text planning. In *Proceedings of International Conference on Natural Language Generation*, pages 97–108.

Ani Nenkova. 2008. Entity-driven rewrite for multi-document summarization. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pages 118–125.

Shimei Pan and James Shaw. 2004. Segue: A hybrid case-based surface natural language generator. In *Natural Language Generation*, pages 130–140. Springer.

Ehud Reiter, Robert Dale, and Zhiwei Feng. 2000. *Building natural language generation systems*, volume 33. MIT Press.

Amanda Stent, Marilyn A Walker, Steve Whittaker, and Preetam Maloor. 2002. User-tailored generation for spoken dialogue: an experiment. In *INTERSPEECH*.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable sentence planning for complex information presentation in spoken dialog systems. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Kapil Thadani and Kathleen McKeown. 2013. Supervised sentence fusion with single-stage inference. In *IJCNLP*, pages 1410–1418.

Marilyn A Walker, Owen C Rambow, and Monica Rogati. 2002. Training a sentence planner for spoken dialogue using boosting. *Computer Speech & Language*, 16(3):409–433.

Marilyn A Walker, Amanda Stent, François Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research*, 30:413–456.

Michael White and David M Howcroft. 2015. Inducing clause-combining rules: A case study with the sparky restaurant corpus. *ENLG 2015*, pages 28–37.