Unsupervised Sentence Enhancement for Automatic Summarization

Jackie Chi Kit Cheung
University of Toronto
10 King's College Rd., Room 3302
Toronto, ON, Canada M5S 3G4
jcheung@cs.toronto.edu

Gerald Penn
University of Toronto
10 King's College Rd., Room 3302
Toronto, ON, Canada M5S 3G4
gpenn@cs.toronto.edu

Abstract

We present *sentence enhancement* as a novel technique for text-to-text generation in abstractive summarization. Compared to extraction or previous approaches to sentence fusion, sentence enhancement increases the range of possible summary sentences by allowing the combination of dependency subtrees from any sentence from the source text. Our experiments indicate that our approach yields summary sentences that are competitive with a sentence fusion baseline in terms of content quality, but better in terms of grammaticality, and that the benefit of sentence enhancement relies crucially on an event coreference resolution algorithm using distributional semantics. We also consider how text-to-text generation approaches to summarization can be extended beyond the source text by examining how human summary writers incorporate source-text-external elements into their summary sentences.

1 Introduction

Sentence fusion is the technique of merging several input sentences into one output sentence while retaining the important content (Barzilay and McKeown, 2005; Filippova and Strube, 2008; Thadani and McKeown, 2013). For example, the input sentences in Figure 1 may be fused into one output sentence.

As a text-to-text generation technique, sentence fusion is attractive because it provides an avenue for moving beyond sentence extraction in automatic summarization, while not requiring deep semantic analysis beyond, say, a dependency parser and lexical semantic resources.

Input: *Bil Mar Foods Co., a meat processor owned by Sara Lee, announced a recall of certain* ***lots of hot dogs and packaged meat****.*

Input: ***The outbreak led to the recall on Tuesday of*** *15 million pounds of hot dogs and cold cuts* ***produced at the Bil Mar Foods plant.***

Output: *The outbreak led to the recall on Tuesday of lots of hot dogs and packaged meats produced at the Bil Mar Foods plant.*

Figure 1: An example of fusing two input sentences into an output sentence. The sections of the input sentences that are retained in the output are shown in bold.

The overall trajectory pursued in the field can be characterized as a move away from local contexts relying heavily on the original source text towards more global contexts involving reformulation of the text. Whereas sentence extraction and sentence compression (Knight and Marcu, 2000, for example) involve taking one sentence and perhaps removing parts of it, traditional sentence fusion involves reformulating a small number of relatively similar sentences in order to take the union or intersection of the information present therein.

In this paper, we move further along this path in the following ways. First, we present **sentence enhancement** as a novel technique which extends sentence fusion by combining the subtrees of many sentences into the output sentence, rather than just a few. Doing so allows relevant information from sentences that are not similar to the original input sentences to be added during fusion. As

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 775–786,
October 25-29, 2014, Doha, Qatar. ©2014 Association for Computational Linguistics

Source text: *This fact has been underscored in the last few months by two unexpected outbreaks* ***of food-borne illness****.*

Output: *The outbreak* ***of food-borne illness*** *led to the recall on Tuesday of lots of hot dogs and meats produced at the Bil Mar Foods plant.*

Figure 2: An example of sentence enhancement, in which parts of dissimilar sentences are incorporated into the output sentence.

shown in Figure 2, the phrase *of food-borne illness* can be added to the previous output sentence, despite originating in a source text sentence that is quite different overall.

Elsner and Santhanam (2011) proposed a supervised method to fuse disparate sentences, which takes as input a small number of sentences with compatible information that have been manually identified by editors of articles. By contrast, our algorithm is unsupervised, and tackles the problem of identifying compatible event mergers in the entire source text using an event coreference module. Our method outperforms a previous syntax-based sentence fusion baseline on measures of summary content quality and grammaticality.

Second, we analyze how text-to-text generation systems may make use of text that is not in the source text itself, but in articles on a related topic in the same domain. By examining the parts of human-written summaries that are not found in the source text, we find that using in-domain text allows summary writers to more precisely express some target semantic content, but that more sophisticated computational semantic techniques will be required to enable automatic systems to likewise do so.

A more general argument of this paper is that the apparent dichotomy between text-to-text generation and semantics-to-text generation can be resolved by viewing them simply as having different starting points towards the same end goal of precise and wide-coverage NLG. The statistical generation techniques developed by the text-to-text generation community have been successful in many domains. Yet the results of our experiments and studies demonstrate the following: as text-to-text generation techniques move beyond using local contexts towards more dramatic reformulations of the kind that human writers perform, more semantic analysis will be needed in order to ensure that the reformulations preserve the inferences that can be drawn from the input text.

2 Related Work

A relatively large body of work exists in sentence compression (Knight and Marcu, 2000; McDonald, 2006; Galley and McKeown, 2007; Cohn and Lapata, 2008; Clarke and Lapata, 2008, inter alia), and sentence fusion (Barzilay and McKeown, 2005; Marsi and Krahmer, 2005; Filippova and Strube, 2008; Filippova, 2010; Thadani and McKeown, 2013). Unlike this work, our sentence enhancement algorithm considers the entire source text and is not limited to the initial input sentences. Few previous papers focus on combining the content of diverse sentences into one output sentence. Wan et al. (2008) propose sentence augmentation by identifying "seed" words in a single original sentence, then adding information from auxiliary sentences based on word co-occurrence counts. Elsner and Santhanam (2011) investigate the idea of fusing disparate sentences with a supervised algorithm, as discussed above.

Previous studies on cut-and-paste summarization (Jing and McKeown, 2000; Saggion and Lapalme, 2002) investigate the operations that human summarizers perform on the source text in order to produce the summary text. Our previous work argued that current extractive systems rely too heavily on notions of information centrality (Cheung and Penn, 2013). This paper extends this work by identifying specific linguistic factors correlated with the use of source-text-external elements.

3 A Sentence Enhancement Algorithm

The basic steps in our sentence expansion algorithm are as follows: (1) clustering to identify initial input sentences, (2) sentence graph creation, (3) sentence graph expansion, (4) tree generation, and (5) linearization.

At a high level, our method for sentence enhancement is inspired by the syntactic sentence fusion approach of Filippova and Strube (2008) (henceforth, F&S) originally developed for German, in that it operates over the dependency parses of a small number of input sentences to produce an output sentence which fuses parts of the input sentences. We adopt the same assumption as F&S that these initial **core sentences** have a high degree of similarity with each other, and should form the core of a new sentence to be generated (Step 1). While fusion from highly disparate input sentences is possible, Elsner and Santhanam (2011) showed how difficult it is to do so correctly, even where such cases are manually identified. We thus aim for a more targeted type of fusion initially. Next, the dependency trees of the core sentences are fused into an intermediate **sentence graph** (Step 2), a directed acyclic graph from which the final sentence will be generated (Steps 4 and 5). We will compare against our implementation of F&S, adapted to English.

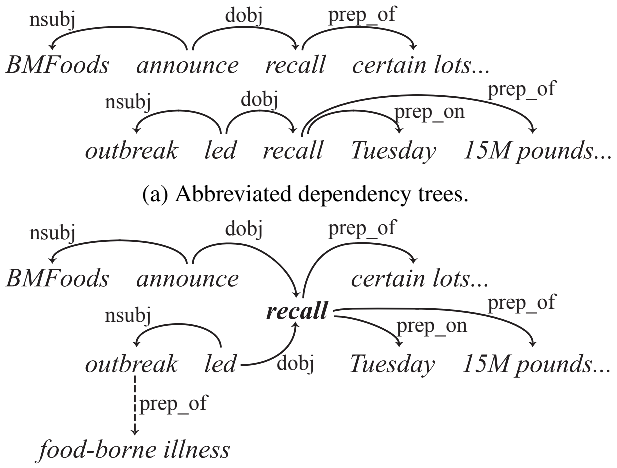
However, unlike F&S or other previous approaches to sentence fusion, the sentence enhancement algorithm may also avail itself of the dependency parses of all of the other sentences in the source text, which expands the range of possible sentences that may be produced. This is accomplished by expanding the sentence graph with parts of these sentences (Step 3). One important issue here is that the expansion must be modulated by an event coreference component to ensure that the merging of information from different points in the source text is valid and does not result in incorrect or nonsensical inferences.

3.1 Core sentence identification

To generate the core sentence clusters, we first identify clusters of similar sentences, then rank the clusters according to their salience. The top cluster in the source text is then selected to be the input to the sentence fusion algorithms.

Sentence alignment is performed by complete-link agglomerative clustering, which requires a measure of similarity between sentences and a stopping criterion. We define the similarity between two sentences to be the standard cosine similarity between the lemmata of the sentences, weighted by IDF and excluding stopwords, and clustering is run until a similarity threshold of 0.5 is reached. Since complete-link clustering prefers small coherent clusters and we select the top-scoring cluster in each document collection, the method is somewhat robust to different choices of the stopping threshold.

The clusters are scored according to the signature term method of Lin and Hovy (2000), which assigns an importance score to each term according to how much more often it appears in the source text compared to some irrelevant background text using a log likelihood ratio. Specifically, the score of a cluster is equal to the sum of the importance scores of the set of lemmata in the cluster.



(a) Abbreviated dependency trees.

(b) Sentence graph after merging the nodes with lemma *recall* (in bold), and expanding the node *outbreak* (dashed outgoing edge).

Figure 3: An example of the input dependency trees for sentence graph creation and expansion, using the input sentences of Figure 1.

3.2 Sentence graph creation

After core sentence identification, the next step is to align the nodes of the dependency trees of the core input sentences in order to create the initial sentence graph. The input to this step is the collapsed dependency tree representations of the core sentences produced by the Stanford parser[1]. In this representation, preposition nodes are collapsed into the label of the dependency edge between the functor of the prepositional phrase and the prepositional object. Chains of conjuncts are also split, and each argument is attached to the parent. In addition, auxiliary verbs, negation particles, and noun-phrase-internal elements[2] are collapsed into their parent nodes. Figure 3a shows the abbreviated dependency representations of the input sentences from Figure 1.

Then, a sentence graph is created by merging nodes that share a common lemma and part-of-

[1] As part of the CoreNLP suite: `http://nlp.stanford.edu/software/corenlp.shtml`

[2] As indicated by the dependency edge label *nn*.

speech tag. In addition, we allow synonyms to be merged, defined as being in the same WordNet synset. Merging is blocked if the word is a stop word, which includes function words as well as a number of very common verbs (e.g., *be*, *have*, *do*). Throughout the sentence graph creation and expansion process, the algorithm disallows the addition of edges that would result in a cycle in the graph.

3.3 Sentence graph expansion

The initial sentence graph is expanded by merging in subtrees from dependency parses of non-core sentences drawn from the source text. First, expansion candidates are identified for each node in the sentence graph by finding all of the dependency edges in the source text from non-core sentences in which the governor of the edge shares the same lemma and POS tag as the node in the sentence graph.

Then, these candidate edges are pruned according to two heuristics. The first is to keep only one candidate edge of each dependency relation type according to the edge that has the highest informativeness score (Section 3.4.1), with ties being broken according to which edge has a subtree with a fewer number of nodes. The second is to perform event coreference in order to prune away those candidate edges which are unlikely to be describing the same event as the core sentences, as explained in the next section. Finally, any remaining candidate edges are fused into the sentence graph, and the subtree rooted at the dependent of the candidate edge is added to the sentence graph as well. See Figure 3b for an example of sentence graph creation and expansion.

3.3.1 Event coreference

One problem of sentence fusion is that the different inputs of the fusion may not refer to the same event, resulting in an incorrect merging of information, as would be the case in the following example:

S1: *Officers pled not guilty but* ***risked*** *25 years to life.*

S2: *Officers recklessly engaged in conduct which seriously* ***risked*** *the lives of others.*

Here, the first usage of *risk* refers to the potential sentence imposed if the officers are convicted in a trial, whereas the second refers to the potential harm caused by the officer.

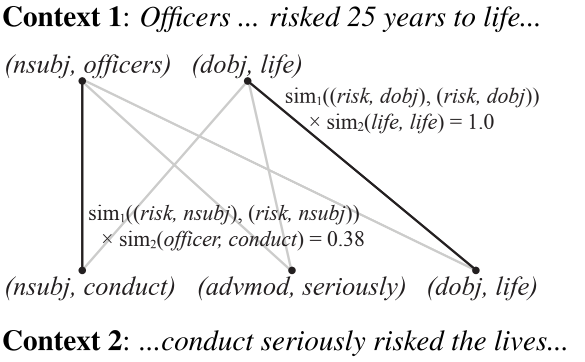

Context 1: *Officers ... risked 25 years to life...*
(nsubj, officers)
(dobj, life)
sim1((risk, dobj), (risk, dobj)) × sim2(life, life) = 1.0
sim1((risk, nsubj), (risk, nsubj)) × sim2(officer, conduct) = 0.38
(nsubj, conduct)
(advmod, seriously)
(dobj, life)
Context 2: *...conduct seriously risked the lives...*

Figure 4: Event coreference resolution as a maximum-weight bipartite graph matching problem. All the nodes share the predicate *risk*.

In order to ensure that sentence enhancement does not lead to the merging of such incompatible events, we designed a simple method to approximate event coreference resolution that does not require event coreference labels. This method is based on the intuition that different mentions of an event should contain many of the same participants. Thus, by measuring the similarity of the arguments and the syntactic contexts between the node in the sentence graph and the candidate edge, we can have a measure of the likelihood that they refer to the same event.

We would be interested in integrating existing event coreference resolution systems into this step in the future, such as the unsupervised method of Bejan and Harabagiu (2010). Existing event coreference systems tend to focus on cases with different heads (e.g., *X kicked Y*, then *Y was injured*), which could increase the possibilities for sentence enhancement, if the event coreference module is sufficiently accurate. However, since our method currently only merges identical heads, we require a more fine-grained method based on distributional measures of similarity.

We measure the similarity of these syntactic contexts by aligning the arguments in the syntactic contexts and computing the similarity of the aligned arguments. These problems can be jointly solved as a maximum-weight bipartite graph matching problem (Figure 4). Formally, let a syntactic context be a list of dependency triples (h, r, a), consisting of a governor or head node h and a dependent argument a in the dependency relation r, where head node h is fixed across each element of the list. Then, each of the two input syntactic contexts forms one of the two disjoint sets in a complete weighted bipartite graph where each node corresponds to one dependency triple. We define the edge weights according to the similarities of the edge's incident nodes; i.e., between two dependency triples (h_1, r_1, a_1) and (h_2, r_2, a_2). We also decompose the similarity into the similarities between the head and relation types ((h_1, r_1) and (h_2, r_2)), and between the arguments (a_1 and a_2). The edge weight function is thus:

$$\begin{aligned}\text{sim}((h_1, r_1, a_1), (h_2, r_2, a_2)) = \\ \text{sim}_1((h_1, r_1), (h_2, r_2)) \times \text{sim}_2(a_1, a_2),\end{aligned} \quad (1)$$

where sim_1 and sim_2 are binary functions that represent the similarities between governor-relation pairs and dependents, respectively. We train models of distributional semantics using a large background corpus; namely, the Annotated Gigaword corpus (Napoles et al., 2012). For sim_1, we create a vector of counts of the arguments that are seen filling each (h, r) pair, and define the similarity between two such pairs to be the cosine similarity between their argument vectors. For sim_2, we create a basic vector-space representation of a word d according to words that are found in the context of word d within a five-word context window, and likewise compute the cosine similarity between the word vectors. These methods of computing distributional similarity are well attested in lexical semantics for measuring the relatedness of words and syntactic structures (Turney and Pantel, 2010), and similar methods have been applied in text-to-text generation by Ganitkevitch et al. (2012), though the focus of that work is to use paraphrase information thus learned to improve sentence compression.

The resulting graph matching problem is solved using the NetworkX package for Python[3]. The final similarity score is an average of the similarity scores from Equation 1 that participate in the selected matching, weighted by the product of the IDF scores of the dependent nodes of each edge. This final score is used as a threshold that candidate contexts from the source text must meet in order to be eligible for being merged into the sentence graph. This threshold was tuned by cross-validation, and can remain constant, although retuning to different domains (a weakly supervised alternative) is likely to be beneficial.

3.4 Tree generation

The next major step of the algorithm is to extract an output dependency tree from the expanded sentence graph. We formulate this as an integer linear program, in which variables correspond to edges of the sentence graph, and a solution to the linear program determines the structure of an output dependency tree. We use ILOG CPLEX to solve all of the integer linear programs in our experiments.

A good dependency tree must at once express the salient or important information present in the input text as well as be grammatically correct and of a manageable length. These desiderata are encoded into the linear program as constraints or as part of the objective function.

3.4.1 Objective function

We designed an objective function that considers the importance of the words and syntactic relations that are selected as well as accounts for redundancy in the output sentence. Let X be the set of variables in the program, and let each variable in X take the form $x_{h,r,a}$, a binary variable that represents whether an edge in the sentence graph from a head node with lemma h to an argument with lemma a in relation r is selected. For a lexicon Σ, our objective function is:

$$\max \sum_{w \in \Sigma} \max_{x_{h,r,a} \in X s.t. a=w} (x_{h,r,w} \cdot P(r|h) \cdot I(w)), \quad (2)$$

where $P(r|h)$ is the probability that head h projects the dependency relation r, and $I(w)$ is the informativeness score for word w as defined by Clarke and Lapata (2008). This formulation encourages the selection of words that are informative according to $I(w)$ and syntactic relations that are probable. The inner $\max$ function for each w in the lexicon encourages non-redundancy, as each word may only contribute once to the objective value. This function can be rewritten into a form compatible with a standard linear program by the addition of auxiliary variables and constraints. For more details of how this and other aspects of the linear program are implemented, see the supplementary document.

3.4.2 Constraints

Well-formedness constraints, taken directly from F&S, ensure that the set of selected edges pro-

[3] http://networkx.github.io/

duces a tree. Another constraint limits the number of content nodes in the tree to 11, which corresponds to the average number of content nodes in human-written summary sentences in the data set. Syntactic constraints aim to ensure grammaticality of the output sentence. In addition to the constraint proposed by F&S regarding subordinating conjunctions, we propose two other ones. The first ensures that a nominal or adjectival predicate must be selected with a copular construction at the top level of a non-finite clause. The second ensures that transitive verbs retain both of their complements in the output[4]. Semantic constraints ensure that only noun phrases of sufficiently high similarity which are not in a hyperonym-hyponym or holonym-meronym relation with each other may be joined by coordination.

3.5 Linearization

The final step of our method is to linearize the dependency tree from the previous step into the final sequence of words. We implemented our own linearization method to take advantage of the ordering information can be inferred from the original source text sentences.

Our linearization algorithm proceeds top-down from the root of the dependency tree to the leaves. At each node of the tree, linearization consists of realizing the previously collapsed elements such as prepositions, determiners and noun compound elements, then ordering the dependent nodes with respect to the root node and each other. Restoring the collapsed elements is accomplished by simple heuristics. For example, prepositions and determiners precede their accompanying noun phrase.

The dependent nodes are ordered by a sorting algorithm, where the order between two syntactic relations and dependent nodes (r_1, a_1) and (r_2, a_2) is determined as follows. First, if a_1 and a_2 originated from the same source text sentence, then they are ordered according to their order of appearance in the source text. Otherwise, we consider the probability $P(r_1 \text{ precedes } r_2)$, and order a_1 before a_2 iff $P(r_1 \text{ precedes } r_2) > 0.5$. This distribution, $P(r_1 \text{ precedes } r_2)$, is estimated by counting and normalizing the order of the relation types in the source text corpus. For the purposes of ordering, the governor node is treated as if it were a dependent node with a special syntactic relation label *self*. This algorithm always produces an output ordering with a projective dependency tree, which is a reasonable assumption for English.

4 Experiments

4.1 Method

Recent approaches to sentence fusion have often been evaluated as isolated components. For example, F&S evaluate the output sentences by asking human judges to rate the sentences' informativeness and grammaticality according to a 1–5 Likert scale rating. Thadani and McKeown (2013) combine grammaticality ratings with an automatic evaluation which compares the system output against gold-standard sentences drawn from summarization data sets. However, this evaluation setting still does not reflect the utility of sentence fusion in summarization, because the input sentences come from human-written summaries rather than the original source text.

We adopt a more realistic setting of using sentence fusion in automatic summarization by drawing the input or core sentences automatically from the source text, then evaluating the output of the fusion and expansion algorithm directly as one-sentence summaries according to standard summarization evaluation measures of content quality.

Data preparation. Our experiments are conducted on the TAC 2010 and TAC 2011 Guided Summarization corpus (Owczarzak and Dang, 2010), on the initial summarization task. Each document cluster is summarized by one sentence, generated from an initial cluster of core sentences as described in Section 3.1.

Evaluation measures. We evaluate summary content quality using the word-overlap measures ROUGE-1 and ROUGE-2, as is standard in the summarization community. We also measure the quality of sentences at a syntactic or shallow semantic level that operates at the level of dependency triples by a measure that we call **Pyramid BE**. Specifically, we extract all of the dependency triples of the form $t = (h, r, a)$ from the sentence under evaluation and the gold-standard summaries, where h and a are the lemmata of the head and the argument, and r is the syntactic relation, normalized for grammatical voice and excluding the collapsed edges which are mostly noun-phrase-internal elements and grammatical

[4] We did not experiment with changing the grammatical voice in the output tree, such as introducing a passive construction if only a direct object is selected, but this is one possible extension of the algorithm.

Method	Pyramid BE	ROUGE-1	ROUGE-2	Log Likelihood	Oracle Pyramid BE
Fusion (F&S)	10.61	**10.07**	**2.15**	-159.31	28.00
Expansion	8.82	9.41	1.82	-157.46	52.97
+Event coref	**11.00**	9.76	1.93	**-156.20**	40.30

Table 1: Results of the sentence enhancement and fusion experiments.

particles. Then, we perform a matching between the set of triples in the sentence under evaluation and in a reference summary following the Transformed BE method of Tratz and Hovy (2008) with the *total* weighting scheme. This matching is performed between the sentence and every gold-standard summary, and the maximum of these scores is taken. This score is then divided by the maximum score that is achievable using the number of triples present in the input sentence, as inspired by the Pyramid method. This denominator is more appropriate than the one used in Transformed BE, which is designed for the case where the evaluated summary and the reference summaries are of comparable length.

For grammaticality, we parse the output sentences using the Stanford parser[5], and use the log likelihood of the most likely parse of the sentence as a coarse estimate of grammaticality. Parse log likelihoods have been shown to be useful in determining grammaticality (Wagner et al., 2009), and many of the problems associated with using it do not apply in our evaluation, because our sentences have a fixed number of content nodes, and contain similar content. While we could have conducted a user study to elicit Likert-scale grammaticality judgements, such results are difficult to interpret and the scores depend heavily on the set of judges and the precise evaluation setting, as is the case for sentence compression (Napoles et al., 2011).

4.2 Results and discussion

As shown in Table 1, sentence enhancement with coreference outperforms the sentence fusion algorithm of F&S in terms of the Pyramid BE measure and the baseline expansion algorithm, though only the latter difference is statistically significant ($p = 0.019$[6]). In terms of the ROUGE word overlap measures, fusion achieves a better performance, but it only outperforms the expansion baseline significantly (ROUGE-1: $p = 0.021$, ROUGE-2: $p = 0.012$). Note that the ROUGE scores are low because they involve comparing a one-sentence summary against a paragraph-long gold standard. The average log likelihood result suggests that sentence enhancement with event coreference produces sentences that are more grammatical than traditional fusion does, and this difference is statistically significant ($p = 0.044$). These results show that sentence enhancement with event coreference is competitive with a strong previous sentence fusion method in terms of content, despite having to combine information from more diverse sentences. This does not come at the expense of grammaticality; in fact, it seems that having a greater possible range of output sentences may even improve the grammaticality of the output sentences.

Oracle score. To examine the potential of sentence enhancement, we computed an oracle score that provides an upper bound to the best possible sentence that may be extracted from the sentence graph. First, we ranked all of dependency triples found in each gold-standard summary by their score (i.e., the number of gold-standard summaries they appear in). Then, we took the highest scoring triples from this ranking that are found in the sentence graph until the length limit was reached, and divided by the Pyramid-based denominator as above[7]. The oracle score is the maximum of these scores over the gold-standard summaries. The resulting oracle scores are shown in the rightmost column of Table 1. While it is no surprise that the oracle score improves after the sentence graph is expanded, the large increase in the oracle score indicates the potential of sentence enhancement for generating high-quality summary sentences.

[5] The likelihoods are obtained by the PCFG model of CoreNLP version 1.3.2. We experimented with the Berkeley parser (Petrov et al., 2006) as well, with similar results that favour the sentence enhancement with event coreference method, but because the parser failed to parse a number of cases, we do not report those results here.

[6] All statistical significance results in this section are for Wilcoxon signed-rank tests.

[7] There is no guarantee that these dependency triples form a tree structure. Hence, this is an upper bound.

Grammaticality. There is still room for improvement in the grammaticality of the generated sentences, which will require modelling contexts larger than individual predicates and their arguments. Consider the following output of the sentence enhancement with event coreference system:

(3) *The government has launched an investigation into Soeharto's wealth by the Attorney General's office on the wealth of former government officials.*

This sentence suffers from coherence problems because two pieces of information are duplicated. The first is the subject of the investigation, which is expressed by two prepositional objects of *investigation* with the prepositions *into* and *on*. The second, more subtle incoherence concerns the body that is responsible for the investigation, which is expressed both by the subject of *launch* (***The government** has launched an investigation*), and the *by*-prepositional object of *investigation* (an investigation ... ***by the Attorney General's office***). Clearly, a model that makes fewer independence assumptions about the relation between different edges in the sentence graph is needed.

5 A Study of Source-External Elements

The sentence enhancement algorithm presented above demonstrates that it is possible to use the entire source text to produce an informative sentence. Yet it is still limited by the particular predicates and dependency relations that are found in the source. The next step towards developing abstractive systems that exhibit human-like behaviour is to try to incorporate elements into the summary that are not found in the source text at all.

Despite its apparent difficulty, there is reason to be hopeful for text-to-text generation techniques even in such a scenario. In particular, we showed in earlier work that almost all of the **caseframes**, or pairs of governors and relations, in human-written summaries can be found in the source text or in a small set of additional related articles that belong to the same domain as the source text (e.g., natural disasters) (Cheung and Penn, 2013). What that study lacks, however, is a detailed analysis of the factors surrounding why human summary writers use non-source-text elements in their summaries, and how these may be automatically identified in the in-domain text. In this section, we supply such an analysis and provide evidence that human summary writers actually do incorporate elements external to the source text for a reason, namely, that these elements are more specific to the semantic content that they wish to convey. We also identify a number of features that may be useful for automatically determining the appropriateness of these in-domain elements in a summary.

5.1 Method

We performed our analysis on the predicates present in text, such as *kill* and *computer*. We also analyzed predicate-relation pairs (**PR pairs**) such as $(kill, nsubj)$ or $(computer, amod)$. This choice is similar to the caseframes used by Cheung and Penn (2013), and we similarly apply transformations to normalize for grammatical voice and other syntactic alternations, but we consider PR pairs of all relation types, unlike caseframes, which only consider verb complements and prepositional objects. PR pairs are extracted from the preprocessed corpus. We use the TAC 2010 Guided Summarization data set for our analyses, which we organize into two sub-studies. In the provenance study, we divide the PR pairs in human-written summaries according to whether they are found in the source text (**source-internal**) or not (**source-external**). In the domain study, we divide in-domain but source-external predicate-relation pairs according to whether they are used in a human-written summary (**gold-standard**) or not (**non-gold-standard**).

5.2 Provenance Study

In the first study, we compare the characteristics of gold-standard predicates and PR pairs according to their provenance; that is, are they found in the source text itself? The question that we try to answer is why human summarizers need to look beyond the source text at all when writing their summaries. We will provide evidence that they do so because they can find predicates that are more appropriate to the content that is being expressed according to two quantitative measures.

Predicate provenance. Source-external PR pairs may be external to the source text for two reasons. Either the predicate (i.e., the actual word) is found in the source text, but the dependency relation (i.e., the semantic predication that holds between the predicate and its arguments) is not found with that particular predicate, or the

	Average freq (millions)
Source-internal	1.77 (1.57, 2.08)
Source-external	1.15 (0.99, 1.50)

(a) The average predicate frequency of source-internal vs. source-external gold-standard predicates in an external corpus.

	Arg entropy
Source-internal	7.94 (7.90, 7.97)
Source-external	7.42 (7.37, 7.48)

(b) The average argument entropy of source-internal vs. source-external PR pairs in bits.

Table 2: Results of the provenance study. 95% confidence intervals are estimated by the bootstrap method and indicated in parentheses.

predicate itself may be external to the source text altogether. If the former is true, then a generalized version of the sentence enhancement algorithm presented in this paper could in principle capture these PR-pairs. We thus compute the proportion of source-external PR pairs where the predicate already exists in the source text.

We find that 2413 of the 4745 source-external PR pairs, or 51% have a predicate that can be found in the source text. This indicates that an extension of the sentence enhancement with event coreference approach presented in this paper could capture a substantial portion of the source-external PR pairs in its hypothesis space already.

Predicate frequency. What factors then can account for the remaining predicates that are not found in the source text at all? The first such factor we identify is the frequency of the predicates. Here, we take frequency to be the number of occurrences of the predicate in an external corpus; namely the Annotated Gigaword, which gives us a proxy for the specificity or informativeness of a word. In this comparison, we take the set of predicates in human-written summaries, divide them according to whether they are found in the source text or not, and then look up their frequency of appearance in the Annotated Gigaword corpus.

As Table 2a shows, the predicates that are not found in the source text consist of significantly less frequent words on average (Wilcoxon rank-sums test, $p < 10^{-17}$). This suggests that human summary writers are motivated to use source-external predicates, because they are able to find a more informative or apposite predicate than the ones that are available in the source text.

Entropy of argument distribution. Another measure of the informativeness or appropriateness of a predicate is to examine the range of arguments that it tends to take. A more generic word would be expected to take a wider range of arguments, whereas a more particular word would take a narrower range of arguments, for example those of a specific entity type. We formalize this notion by measuring the entropy of the distribution of arguments that a predicate-relation pair takes as observed in Annotated Gigaword. Given frequency statistics $f(h, r, a)$ of predicate head h taking an argument word a in relation r, we define the argument distribution of predicate-relation pair (h, r) as:

$$P(a|h, r) = f(h, r, a) / \sum_{a'} f(h, r, a') \quad (4)$$

We then compute the entropy of $P(a|h, r)$ for the gold-standard predicate-relation pairs, and compare the average argument entropies of the source-internal and the source-external subsets.

Table 2b shows the result of this comparison. Source-external PR pairs exhibit a lower average argument entropy, taking a narrower range of possible arguments. Together these two findings indicate that human summary writers look beyond the source text not just for the sake of diversity or to avoid copying the source text; they do so because they can find predicates that more specifically convey some desired semantic content.

5.3 Domain study

The second study examines how to distinguish those source-external predicates and PR pairs in in-domain articles that are used in a summary from those that are not. For this study, we rely on the topic category divisions in the TAC 2010 data set, and define the in-domain text to be the documents that belong to the same topic category as the target document cluster (but not including the target document cluster itself). This study demonstrates the importance of better semantic understanding for developing a text-to-text generation system that uses in-domain text, and identifies potentially useful features for training such a system.

Nearest neighbour similarity. In the event-coreference step of the sentence enhancement algorithm, we relied on distributional semantics to

	N	NN sim
GS	2202	0.493 (0.486, 0.501)
Non-GS	789K	0.443 (0.442, 0.443)

(a) Average similarity of gold-standard (GS) and non-gold-standard (non-GS) PR pairs to the nearest neighbour in the source text.

	N	Freq. (millions)	Fecundity
GS	1568	2.44 (2.05, 2.94)	21.6 (20.8, 22.5)
non-GS	268K	0.85 (0.83, 0.87)	6.43 (6.41, 6.47)

(b) Average frequency and fecundity of GS and non-GS predicates in an external corpus. The differences are statistically significant ($p < 10^{-10}$).

Table 3: Results of the domain study. 95% confidence intervals are given in parentheses.

measure the similarity of arguments. Here, we examine how well distributional similarity determines the appropriateness of a source-external PR pair in a summary. Specifically, we measure its similarity to the nearest PR pair in the source text. To determine the similarity between two PR pairs, we compute the cosine similarity between their vector representations. The vector representation of a PR pair is the concatenation of a context vector for the predicate itself and a selectional preferences vector for the PR pair; that is, the vector of counts with elements $f(h, r, a)$ for fixed h and r. These vectors are trained from the Annotated Gigaword corpus.

The average nearest-neighbour similarities of PR pairs are shown in Table 3a. While the difference between the gold-standard and non-gold-standard PR pairs is indeed statistically significant, the magnitude of the difference is not large. This illustrates the challenge of mining source-external text for abstractive summarization, and demonstrates the need for a more structured or detailed semantic representation in order to determine the PR pairs that would be appropriate. In other words, the kind of simple event coreference method based solely on distributional semantics that we used in Section 3.3.1 is unlikely to be sufficient when moving beyond the source text.

Frequency and fecundity. We also explore several features that would be relevant to identifying predicates in in-domain text that are used in the automatic summary. This is a difficult problem, as less than 0.6% of such predicates are actually used in the source text. As a first step, we consider several simple measures of the frequency and characteristics of the predicates.

The first measure that we compute is the average predicate frequency of the gold-standard and non-gold-standard predicates in an external corpus, as in Section 5.2. A second, related measure is to compute the number of possible relations that may occur with a given predicate. We call this measure the **fecundity** of a predicate. Both of these are computed with respect to the external Annotated Gigaword corpus, as before.

As shown in Table 3b, there is a dramatic difference in both measures between gold-standard and non-gold-standard predicates in in-domain articles. Gold-standard predicates tend to be more common words compared to non-gold-standard ones. This result is not in conflict with the result in the provenance study that source-external predicates are less common words. Rather, it is a reminder that the background frequencies of the predicates matter, and must be considered together with the semantic appropriateness of the candidate word.

6 Conclusions

This paper introduced sentence enhancement as a method to incorporate information from multiple points in the source text into one output sentence in a fashion that is more flexible than previous sentence fusion algorithms. Our results show that sentence enhancement improves the content and grammaticality of summary sentences compared to previous syntax-based sentence fusion approaches. Then, we presented studies on the components of human-written summaries that are external to the source text. Our analyses suggest that human summary writers look beyond the source text to find predicates and relations that more precisely express some target semantic content, and that more sophisticated semantic techniques are needed in order to exploit in-domain articles for text-to-text generation in summarization.

Acknowledgments

We would like to thank the anonymous reviewers for valuable suggestions. The first author was supported by a Facebook PhD Fellowship during the completion of this research.

References

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

Cosmin A. Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1412–1422. Association for Computational Linguistics.

Jackie Chi Kit Cheung and Gerald Penn. 2013. Towards robust abstractive multi-document summarization: A caseframe analysis of centrality and domain. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1233–1242, August.

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

Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 137–144, Manchester, UK, August. Coling 2008 Organizing Committee.

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 and Michael Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185, Honolulu, Hawaii, October. 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 (Coling 2010)*, pages 322–330, Beijing, China, August. Coling 2010 Organizing Committee.

Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 180–187.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2012. Monolingual distributional similarity for text-to-text generation. In *Proceedings of *SEM 2012: The First Joint Conference on Lexical and Computational Semantics*, pages 256–264, Montréal, Canada, June. Association for Computational Linguistics.

Hongyan Jing and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pages 178–185.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization—step one: Sentence compression. In *Proceedings of the National Conference on Artificial Intelligence*, pages 703–710.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*, COLING '00, pages 495–501, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

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

Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. 2011. Evaluating sentence compression: Pitfalls and suggested remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97. Association for Computational Linguistics.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the NAACL-HLT Joint Workshop on Automatic Knowledge Base Construction & Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100.

Karolina Owczarzak and Hoa T. Dang. 2010. TAC 2010 guided summarization task guidelines.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*.

Horacio Saggion and Guy Lapalme. 2002. Generating indicative-informative summaries with SumUM. *Computational Linguistics*, 28(4):497–526.

Kapil Thadani and Kathleen McKeown. 2013. Supervised sentence fusion with single-stage inference. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1410–1418, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Stephen Tratz and Eduard Hovy. 2008. Summarization evaluation using transformed Basic Elements. In *Proceedings of the First Text Analysis Conference (TAC)*.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2009. Judging grammaticality: Experiments in sentence classification. *CALICO Journal*, 26(3):474–490.

Stephen Wan, Robert Dale, Mark Dras, and Cecile Paris. 2008. Seed and grow: Augmenting statistically generated summary sentences using schematic word patterns. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 543–552, Honolulu, Hawaii, October. Association for Computational Linguistics.