1-1-2011

Unsupervised Models of Text Structure

Annie Louis
University of Pennsylvania, lannie@seas.upenn.edu

Unsupervised Models of Text Structure

Abstract
Models of text structure are necessary for applications that generate text. These models provide information about what content fits together and how to organize the content as coherent text. In some domains such as newswire, biographies and stories for children, texts tend to have similar content and structure. Such regularities have allowed the development of unsupervised methods to learn text structure using human-written examples from such domains. We survey some of the recently proposed approaches in this area and review their use in different text generation tasks.

First, we consider approaches with a focus on computational semantics. We review work aiming to discover patterns of related events from news articles and children's stories. We consider one application of such knowledge–an automatic story-telling system.

Next, we move to methods which focus on coherence and organization. We describe these in the context of two generation tasks–sentence ordering and the creation of long articles. In view of the sentence ordering problem, we survey approaches targeted at learning properties of coherent transitions between adjacent sentences in texts. Then, we consider the generation of long biographical descriptions. Here we survey recent work on automatically generating such articles using higher level patterns in text structure such as subtopics and their organization.

Comments
University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-11-16.

Unsupervised Models of Text Structure

Annie Louis

Written Preliminary Exam - II Report
Department of Computer and Information Science
University of Pennsylvania

April 14, 2010

Abstract

Models of text structure are necessary for applications that generate text. These models provide information about what content fits together and how to organize the content as coherent text. In some domains such as newswire, biographies and stories for children, texts tend to have similar content and structure. Such regularities have allowed the development of unsupervised methods to learn text structure using human-written examples from such domains. We survey some of the recently proposed approaches in this area and review their use in different text generation tasks.

First, we consider approaches with a focus on computational semantics. We review work aiming to discover patterns of related events from news articles and children's stories. We consider one application of such knowledge–an automatic story-telling system.

Next, we move to methods which focus on coherence and organization. We describe these in the context of two generation tasks–sentence ordering and the creation of long articles. In view of the sentence ordering problem, we survey approaches targeted at learning properties of coherent transitions between adjacent sentences in texts. Then, we consider the generation of long biographical descriptions. Here we survey recent work on automatically generating such articles using higher level patterns in text structure such as subtopics and their organization.

1 Introduction

Automatic systems to perform text generation, summarization and question answering tasks should produce output text that is both informative as well as coherent and readable. To this end, computers need knowledge about how information should be structured in written texts. Our focus in this survey is to review of some of the most recent computational methods for automatically learning properties of text structure using example texts written by humans and requiring no further annotation.

The development of such unsupervised approaches is possible because human-written texts from the same domain exhibit several regularities (Wray, 2002). Such patterns are clearly evident in domains such as newswire. A news report on an earthquake specifies the location and intensity of the earthquake. It gives details regarding the extent of damage from the disaster and the relief efforts that were undertaken. All this content is also organized in a fairly standard manner. Descriptions of casualties typically precede information about relief efforts. One would expect that most news articles on earthquakes and other disasters have similar content and structure. Similarly, biographies might be organized to describe aspects of early life, education and career, in that order. Such patterns have also been observed in other domains such as scientific articles and children's stories. The content in abstracts for scientific articles frequently follow the order of introduction, methods, results and conclusions (Salanger-Meyer, 1990). Propp (1968) found that there are some frequent patterns of event sequences in Russian fairy tales.

Word level cues in these texts may reflect the similarity in structure. For example, one would expect that words such as "damage", "loss" and "deaths" would often appear in news texts prior to "medics", "support" and "compensation". Authors also focus on a small set of entities at a time (Grosz et al., 1995). As a result, entities get repeated in subsequent sentences as the writer continues to talk about them. Such entity links and word co-occurrence properties can be detected and used by computational methods.

Some aspects of higher level structure are also explicitly marked by authors. They demarcate texts into paragraphs with the intention of presenting the content as a series of subtopics. These subtopics can be expected to appear in similar sequences within a domain. Automatic methods can also aim to learn such patterns. The idea behind unsupervised text structure models is that by examining a large collection of human-written texts from a domain, it is possible to identify common patterns across the articles regarding what content is presented and how it is organized.

In this survey, we review some of the recently proposed automatic methods to learn discourse structure from example texts. The work we discuss uses these methods for a range of generation tasks spanning computational story-telling to automatic generation of Wikipedia articles. Some of these approaches involve word co-occurrence and association metrics. Others also make use of coreference information and machine learning methods such as Hidden Markov Models. These approaches and methods have shown varying degrees of success for the proposed applications.

The rest of this survey is organized as follows. We begin with a brief description of some of the theoretical ideas put forth by linguists and computational linguists on the notion of text structure (Section 2). Such theories have been a motivation for several ideas in the automatic approaches. We then survey data-driven approaches by dividing them into two categories for discussion. Some approaches focus on learning semantic content in the form of related entities and events from texts. Others concentrate on organizing content for coherence and readability.

In Section 3, we review automatic methods to obtain semantic knowledge from text. These approaches aim to create a repository or ontology of information about text structure. Here, we review work by Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) who focus on narrative texts. In Chambers and Jurafsky (2009), the idea is to learn situation-specific patterns of events and their participants from a corpus of news articles. For example, a prosecution scenario would involve events related to trial, judgement, sentence, etc. and these events occur in some temporal order. Chambers and Jurafsky (2009) automatically collect such sets of related events and their participants using the frequency of such patterns in the example articles. McIntyre and Lapata (2009) work with a corpus of children's stories. They use recurring patterns of entity-event and event-event co-occurrences in these stories to automatically construct a knowledge base of entity and event properties. This knowledge base is subsequently used to automatically generate new stories.

We then describe methods for organizing content in text generation applications (Section 4). Apart from learning models of text structure, these approaches also propose efficient algorithms to use the information in the target applications. In the context of Soricut and Marcu (2006), we overview the information ordering task, the problem of organizing a set of sentences into coherent text. They use previously proposed as well as new approaches to learn what properties create coherent transitions between adjacent sentences in a text. They also develop efficient methods to use these coherence metrics for ordering sentences. We survey this work in Section 4.1. Soricut and Marcu (2006) work with collections of earthquake and accident descriptions. We then discuss very recent work by Sauper and Barzilay (2009) who consider the problem of automatically generating Wikipedia articles for a given domain. In Section 4.2, we describe how they identify patterns of subtopics in existing Wikipedia articles and develop methods to use this information to create and organize relevant content for new articles. Sauper and Barzilay (2009) experiment with two domains–disease descriptions and biographies of American film actors.

In Section 5, we conclude with some further discussion and suggestions for future work in this area.

2 Linguistic theories of text structure

In this section, we highlight some of the linguistic ideas about text structure that are most relevant to our discussion in this survey. From the linguistic viewpoint, text structure has been described in terms of semantic relations between text units as well as entity and event-based links in the discourse. Automatic approaches incorporate some of these notions while learning from human-written texts.

2.1 Scripts

Abelson and Schank (1977) put forward the idea that memory is organized as a collection of "scripts". A script describes a sequence of events commonly encountered in a particular situation. One of the most famous examples from Abelson and Schank (1977)'s work is the restaurant script. On a visit to a restaurant, one enters the place, finds a table to sit, moves to that table, then orders food, eats, pays the check and leaves. Abelson and Schank (1977) propose that such a sequence of events is so frequently encountered by us that we grow to incorporate such information in our memory. Such knowledge influences what we perceive as related and coherent in different situations. For example, consider the sentences below.

> John was walking on the street. He thought of cabbages. He picked up a shoe horn.

Abelson and Schank (1977) explain that we would regard the above sentences as not meaningful at all because these events do not conform to situations that we know to be common.

2.2 Centering

The Centering theory (Grosz et al., 1995) uses entity repetition properties to describe "local" coherence: *adjacent sentences* in a discourse often share entities and in this way, focus and topic is maintained in the text. Centering also describes that certain entity sharing patterns are preferred to others in view of coherence. This preference is explained based on notions of salience. For example, an entity in subject position is more salient compared to those assuming object and other grammatical roles. In coherent texts, the more salient or focal entities in a sentence are assumed to be the ones more likely to continue as the focus and be mentioned in subsequent sentences. Centering proposes that such preferred coreference patterns make the text coherent for the reader.

2.3 Discourse relations

Another perspective on text structure is given by theories of rhetorical or discourse relations. In these accounts, clauses or sentences in coherent texts are described as connected by semantic relations such as "cause", "contrast" and "elaboration". The Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) puts forth the hypothesis that the units of a text can be connected by such relations in a *tree structure*. At the lowest level, adjacent clauses in the text are related to each other. The related units then recursively participate in relations with other units creating higher level structures such as paragraphs and ultimately the entire text. More recently, theory-neutral approaches have been developed to describe rhetorical structure such as the Penn Discourse Treebank (Prasad et al., 2008). In the latter case, no assumption is made regarding the final structure of the full text.

With this understanding of the linguistic views on text structure, we move on to review approaches and techniques that automatic methods employ for modeling text. We start with a survey of approaches for mining semantic knowledge from texts (Section 3). In Section 4, we describe automatic methods for organizing content.

3 Semantic knowledge from unannotated text

Manual construction of semantic knowledge is costly in terms of the human effort and time involved. Further, human annotations are based upon intuitions about relatedness and may be restricted to only some regularities. One interesting question is whether we can automatically learn a repository of semantic information from a collection of texts.

The papers which we discuss here (Chambers and Jurafsky, 2009; McIntyre and Lapata, 2009) focus on obtaining semantic information about the structure of narrative texts. Chambers and Jurafsky (2009) construct script-like information automatically by identifying related events and their participants from news articles. In McIntyre and Lapata (2009), the idea is to automatically learn a knowledge base about entities and events in children's stories so that the resource can then be used to automatically generate new stories.

3.1 Learning narrative schemas: Chambers and Jurafsky, 2009

A narrative involves several events during its course and a number of different entities participate in these events. Chambers and Jurafsky (2009) hypothesize that this structure can be best described in the form of *narrative schemas*. A narrative schema, similar to scripts (Section 2.1), is associated with a particular situation. Different situations would be modeled by different scripts. Similarly, the idea of a schema in Chambers and Jurafsky (2009) is targeted to model frequent patterns in news articles of small sets of related events and their participants.

Such semantic information would be useful in a variety of tasks. Consider summarization of news documents, for example. When a system chooses information about some event to include in a summary, it might be beneficial to include content about its related events as well. Suppose we know that someone was *arrested*, we would also like to know if the person *was tried*, *found guilty* or *released*. Using schemas to identify other related events for that scenario, a summarization system could explicitly look for such information in the source documents. Consider that the schemas also include information about the order in which these events occur, then the system can use this preferred ordering in the summary to make the text more coherent.

There is some prior work on identifying verb pairs with semantic relations such as "similarity" and "antonymy" as in *VerbOcean* (Chklovski and Pantel, 2004). However, these pairwise relationships do not provide any information about the participants of those verbs. On the other hand, work on semantic role labeling concentrates on the task of identifying for a given verb, the patterns of how its syntactic arguments (subject and object) map to semantic participants (agent and patient) (Grenager and Manning, 2006). In this latter case, the focus is on participants of an individual event in isolation. The idea of schemas proposed in Chambers and Jurafsky (2009) differs from this body of work in three ways by:

(a) Creating larger sets of related verbs describing a situation

(b) Incorporating information about participants of the events

(c) Focusing on temporal relationships between events

3.1.1 Defining relatedness: verb pairs, chains and schemas

Chambers and Jurafsky (2009) start at a low level, learning pairs of related verbs and then use this information to create larger groups of verbs called chains and schemas.

Related verb pairs. Given a corpus of articles, one approach for computing the relatedness of two verbs would be to use the frequency with which the verbs co-occur in the given documents. In contrast, Chambers and Jurafsky (2009) consider two verbs as highly related only if they frequently occur in the articles such that one of their participants is the *same entity (coreferent)*. The common entity is called the *protagonist* or main actor in the events.

Chambers and Jurafsky (2009)'s motivation for using coreference to signal relatedness comes from ideas of entity coherence. Centering theory (Grosz et al., 1995) states that coreference properties of entities in adjacent sentences are one way in which writers create coherence in a discourse and make it easy to follow. Chambers and Jurafsky (2009) consider the situations in narratives as analogous. There are several events along the course of a narrative, some of them involving the same or a common set of participants. For example, consider that the following sentences refer to the same "car" but appear in different sections of a document.

> Smith drove his car to work.
> ...
> The car zipped into another lane without proper signaling.
> ...
> The police pulled the car over.

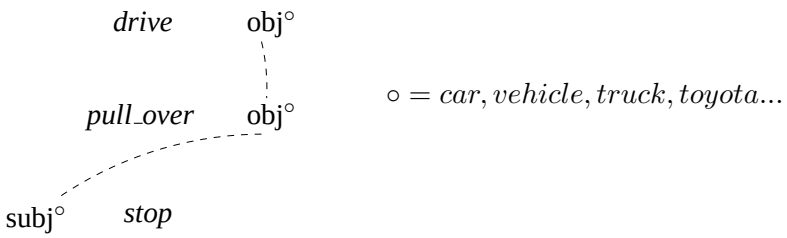

Figure 1: An example narrative chain

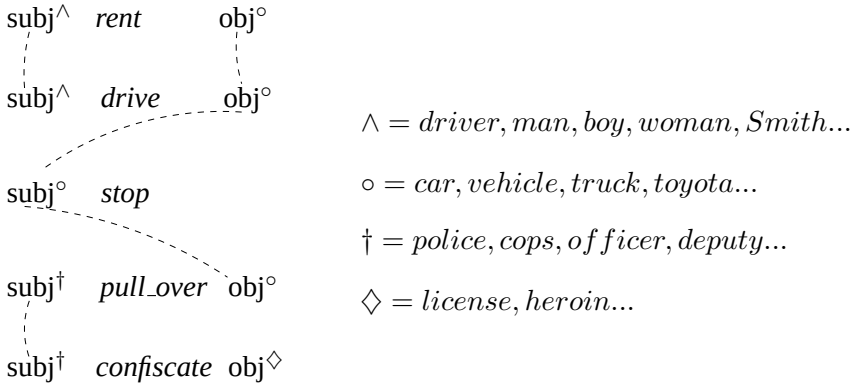

Figure 2: An example narrative schema

The verbs "drive", "zip" and "pull_over" are intuitively related because the same car is involved in all these events. Therefore only verbs with coreferent arguments are considered as semantically related by Chambers and Jurafsky (2009).

In other words, a relationship is defined for a pair of verb-dependency links such that these dependency slots have a coreferring argument. The strength of the association is measured by the frequency with which these dependency slots of the verbs are observed with a coreferent entity in the training corpus. For example, $<drive, obj>$ and $<pull\_over, obj>$ would be considered strongly related if the verbs "drive" and "pull_over" frequently co-occur with a coreferent entity filling their object positions. The actual entity in these positions could be different in the different documents in the corpus, eg. "car", "truck", "bus" etc.

Narrative chains. A narrative chain is an extension of this idea to more than two verbs. A chain consists of *a set of strongly related verb-dependency links*. Entities in the dependency slot associated with a particular verb in the chain often appear in narratives as coreferent with entities in the dependency slots listed for other verbs in the same chain. The construction of chains from pairs of related verbs is discussed in the next section. Only a single dependency link is associated with each verb in a chain, so a chain models a single actor or common entity. Figure 1 shows an example narrative chain comprising a set of verb-dependency links related by entities belonging to the class of vehicles. The set of lexical items that are frequently observed in the protagonist slot of these verbs in the text collection is also recorded. A chain also defines a partial ordering for the verbs to indicate their likely sequence of occurrence.

Narrative schemas. In contrast to chains, schemas (Chambers and Jurafsky, 2009) are *a collection of verbs* such that *all* their participants (in different grammatical roles) are highly related to the situation ie., all the participants frequently corefer with entities involved with other verbs in the schema. In other words, a schema is not restricted to one main protagonist. It consists of a *set of narrative chains*, each chain corresponding to a different protagonist. An example schema is shown in Figure 2. It involves four different protagonist entities. In the next section, we describe how Chambers and Jurafsky (2009) construct these schemas automatically.

3.1.2 Constructing schemas

Computing narrative chains. We first describe the method to compute narrative chains. Chambers and Jurafsky (2008) use point-wise mutual information to compute related verb-dependency pairs. Let $< v, d >$ represent a verb v plus the dependency link d to the protagonist, eg. $< drive, obj >$ when the protagonist "car" is in object position. For two verbs a and b, such that the entities in the chosen dependency links d_x and d_y are coreferent, one can obtain a measure of their co-occurrence as follows:

$$sim(<a, d_x>, <b, d_y>) = log\left(\frac{P(<a, d_x>, <b, d_y> |coref)}{P(<a, d_x> |coref)P(<b, d_y> |coref)}\right) \quad (1)$$

where the entities in the dependency positions of $<a, d_x>$ and $<b, d_y>$ are coreferent and $d_i \in D = \{subject, object, preposition\}$.

$$P(<a, d_x>, <b, d_y> |coref) = \frac{\| <a, d_x>, coref, <b, d_y> \|}{\sum_{s,t} \sum_{d_i, d_j \in D} \| <s, d_i>, coref, <t, d_j> \|}$$

$$P(<a, d_x> |coref) = \frac{\sum_{c} \sum_{d_z \in D} \| <a, d_x>, coref, <c, d_z> \|}{\sum_{s,t} \sum_{d_i, d_j \in D} \| <s, d_i>, coref, <t, d_j> \|}$$

where $\| <s, d_i>, coref, <t, d_j> \|$ represents the number of times the verbs s and t have coreferent arguments filling dependencies d_i and d_j.

A table of verb-dependency pairs and their similarities is created in this way. Starting with a verb, a chain is then created by adding new verbs in decreasing order of their similarity *to the entire chain* at that instant. Similarity of a new verb with a chain is computed as the sum of pairwise similarities with existing verbs in the chain. Suppose that C is a chain with verb-dependency pairs $< e_i, d_{p_i} >$, a new verb-dependency link $<a, d_m>$ is evaluated for relatedness with C as below.

$$chainsim(C, <a, d_m>) = \sum_{<e, d_p> \in C} sim(<e, d_p>, <a, d_m>) \quad (2)$$

Including argument information. The metrics so far are defined entirely based upon the grammatical role of the protagonist entity. They only consider the verb and dependency link but ignore the actual lexical identity of the protagonist while computing similarity (Eqn. 1). But the identity of the protagonist might help improve the estimates of relatedness for verbs.

For example, consider the chain in Figure 1. Suppose that we want to check the similarity of two verb-dependency links $<halt, subject>$ and $<pierce, subj>$ with this chain. $<halt, subj>$ might have coreference with several verb-dependency slots in the chain. On the other hand, $<pierce, subj>$ might have frequent coreference with $<drive, obj>$ (*...drove a nail, the nail pierced...* is a very common construction) but low similarity with other verbs. Suppose that for both candidates, the sum of similarity to all verbs in the chain (*chainsim* as described above) ends up being the same value. Then both candidates are equally good for adding to the chain. Now suppose that we include information that a frequent head word of the entity in subject position of "halt" is "car". Then it is indicative of stronger similarity of $<halt, subj>$ with the chain since "car" is already a frequent lexical realization of the protagonist for that chain. On the other hand, the frequent lexicalization of the protagonist when $<pierce, subj>$ and $<drive, obj>$ are coreferent might be "nail" but this entity is infrequent as a protagonist for existing verbs in the chain. So argument information helps to make a better choice, in this case $<halt, subject>$ for chain construction.

Following this motivation, Chambers and Jurafsky (2009) update the method to compute similarity (Eqn. 1) to also consider the head word p used for the coreferent entity. Now similarity (called *typed* similarity) between two verb-dependency pairs with respect to that entity p is defined as:

$$sim_{typed}(<a, d_x>, <b, d_y>, p) = sim(<a, d_x>, <b, d_y>) + \lambda log(\| <a, d_x>, coref, <b, d_y>, p \|)$$

where $\| <a, d_x>, coref, <b, d_y>, p\|$ is the number of times the dependency positions d_x and d_y of events a and b had a coreferent entity lexicalized as p. λ is a constant weighting factor.

The method to compute suitability of a new verb-dependency pair $< a, d_m >$ for a chain (Eqn. 2) is then modified as follows. $chainsim_{typed}$ considers the argument p which would result in maximum similarity between all pairs of verbs in the chain C with the candidate verb included. Let $<e_i, d_{x_i}>$ be events in the chain, and n be the chain length.

$$chainsim_{typed}(C, <a, d_m>) = \max_p(score(C, p) + \sum_{i=1}^{n} sim_{typed}(<e_i, d_{x_i}>, <a, d_m>, p) \quad (3)$$

$$score(C, p) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} sim_{typed}(<e_i, d_{x_i}>, <f_j, d_{y_j}>, p)$$

In our example, even though $nail$ frequently occurs in the slots $<drive, obj>$ and $<pierce, subj>$, it is a very uncommon entity to be coreferent between other pairs of verbs, so $score(C, \text{“}nail\text{”})$ would turn out very low compared to $score(C, \text{“}car\text{”})$.

Creating schemas. Now we turn to creating schemas which are a collection of chains. Each chain models one of the participants for the set of related events. To add a new verb v to a schema N, we must check for relatedness not only with one chain which accomodates one of v's arguments but also whether its other arguments can be assigned to chains within the same schema.

For example, “search” might be a good verb to add to the schema in Figure 2 because both the frequent subjects of search (police, officer...) and its objects (vehicle, car...) might often be coreferent with the verbs in the schema and can become part of existing chains. On the other hand, consider the verb “grant”, with frequent object “license” and frequent subjects–company, organization, authority. Although its object is already modeled by a chain in the schema, its subject may not be very similar to any of the existing chains. It would be preferrable to add the verb “search” in this case.

Therefore the similarity of a verb with a schema is computed by checking the compatibility of all the dependency slots d_x of the candidate verb with chains C_N belonging to the schema N.

$$schemasim(N, v) = \sum_{d_x \in D} \max(\beta, \max_{c \in C_N} chainsim_{typed}(c, <v, d_x>)) \quad (4)$$

The parameter β is a threshold on similarity with the existing chains. When a verb-dependency pair $<v, d_m>$ does not have enough similarity with any of the existing chains in the schema, it indicates that the entity in d_m cannot be assigned to an existing chain. In this case, Chambers and Jurafsky (2009), check whether it would be beneficial to create a new chain to accomodate the entity in d_m. This new chain would have an initial score of β.

Suppose that the training corpus contains $|V|$ verbs, each candidate verb v_j is checked for similarity with a schema using the above equation and the one scoring maximum similarity is added at each step.

$$v_{to\_add} = \arg\max_{v \in V} \ schemasim(N, v)$$

But Chambers and Jurafsky (2009) do not have a way of determining when to stop adding verbs to a schema and indicate its completion. A schema should have a small set of verbs reflecting a particular situation. But different situations are likely to comprise different number of events. This issue has remained unaddressed in Chambers and Jurafsky (2009). For testing purposes, Chambers and Jurafsky (2009) limit the number of verbs in the schema to six. This limit was chosen to enable a comparison of the schemas with FrameNet (Baker et al., 1998), a manually constructed knowledge base.

3.1.3 Results

Chambers and Jurafsky (2009) collect all verbs which occur more than 3000 but less than 50000 times from a section of the Gigaword Corpus. This filtering removes very frequent verbs which might not be specific to any individual schema. Starting with each of these verbs (approximately 1800 verbs), they create a schema limiting the number of verbs added to six.

Chambers and Jurafsky (2009) do not add temporal relations between the verbs in the schemas. However, this task was addressed for narrative chains in Chambers and Jurafsky (2008). A temporal classifier was trained over the TimeBank corpus (Pustejovsky et al., 2003) to classify for every pair of verbs (a, b) (containing coreferring arguments) whether a 'before' relation was present or otherwise. Then using the frequency with which a is classified to appear before b relative to the frequency with which b is classified as appearing before a, the precedence of a in the pair (a, b) was determined. The orderings produced in Chambers and Jurafsky (2008) were tested in a coherence evaluation task and found to produce reasonable performance.

Comparison with FrameNet FrameNet (Baker et al., 1998) is a manually constructed database of semantic "concepts". Each concept is represented by a *frame* containing a set of lexical items and participants related to the concept. This information is collected using human annotations of large text corpora. For example, the frame for *attack* comprises words such as "onslaught", "raid", "ambush", "assault", etc. and lists the main participants involved in this situation: *an assailant* and *a victim*.

Chambers and Jurafsky (2009) compare the schemas produced starting with the most frequent 20 verbs to frames in FrameNet. Narrative structures, however, are not identical to frame relationships. In FrameNet, the relationship between two frames is defined in terms of whether they share the same participants but in narratives, the similarity also requires them to co-occur in the same narrative. For example, the frames for *complain* and *question* are related as both are types of "statements". However, both need not necessarily appear over the course of a discourse. However, comparison with such a database provides one way of evaluating the automatically learned knowledge.

Each of the 20 schemas was examined for closeness to some frame (the frame covers most of the verbs in the schema) in FrameNet. Only 13 schemas could be mapped in this way. Close to half of the verbs in these mapped schemas occurred either in the same frame or in a related frame spanning one link. However, around 20% of the verbs did not occur in related frames. Chambers and Jurafsky (2009) report that upon examination the vast majority of them were found to be cases where one would still consider them related in a 'narrative' sense.

These results however do not fully bring out the quality of the schemas. Chambers and Jurafsky (2009) only evaluate their top 20 schemas. This is a very small test set to obtain any reasonable conclusions. Even for this data, several schemas do not have mappings to FrameNet and among those which do, only some of the verbs are in related frames. It is true that FrameNet uses a different notion of relatedness and hence the comparison should not be taken as robust evaluation. Then this fact brings us to the question of whether this assessment tells us much about the quality of the produced schemas: probably not. Chambers and Jurafsky (2009) also present an "event prediction task" to evaluate the schemas.

Narrative cloze task Narrative cloze experiments are frequently done in psycholinguistic studies. A word is removed from a sentence, and a person or system is asked to fill it. Depending on the choice made by the person or system, we can say how well they performed on the task. Chambers and Jurafsky (2009) use a similar experiment to test the predictions of relatedness from their similarity metrics.

The chains of events with coreferent arguments in about 70 documents were identified. One of these events was chosen at random and removed from the chain. The *candidate* verbs for the missing slot were generated from the training data. For each verb in the training corpus, its relatedness to the remaining verbs in the chain is measured. The verbs are ranked on the basis of the obtained scores and the rank of the gold standard verb in this list is noted. Ideally, if the relatedness scores could predict the missing verb well, one would end up with a low (better) rank on average for the chains in the test documents.

When relatedness was computed in a schema-like manner (considering all dependencies to compute similarity as in Eqn. 4), the results turned out better compared to using only one protagonist dependency

link as in narrative chains. In addition, the gold standard verb was ranked much lower when argument information was also used while computing the similarity (Eqn. 3). Chambers and Jurafsky (2008) also show that the baseline of computing relatedness based on verb co-occurrence in documents gives significantly lower results as opposed to also requiring that their participants be coreferent.

However, in all the setups, the average rank of the true event is above 1000 which means that the true event is only predicted after nearly 1000 misses. In fact, it is probably the case that several options could fit the missing event, but given how the evaluation can be done, we would only be looking for one specific event which occurred in the text. All improvements suggested by Chambers and Jurafsky (2008) and Chambers and Jurafsky (2009) lower the rank for the gold standard but still the rank is over a 1000. Hence this evaluation again has the same weakness as the comparison to FrameNet. It does not robustly predict the quality of automatically created schemas.

It would have been more useful if Chambers and Jurafsky (2009) performed a small experiment with human judges instead to score some of their schemas on a scale for whether they are likely to represent some situation or whether they are noisy. However, a thorough evaluation of different schemas would be impossible to do with human judges. Alternatively, we could use the automatically generated scripts in some application and evaluate their usefulness in this way. The next paper (McIntyre and Lapata, 2009) which we discuss involves such a task-based setting to evaluate the usefulness of automatically generated semantic information.

3.2 Automatic story generation: McIntyre and Lapata, 2009

While comparison with manually constructed knowledge bases provide some insight into the quality of automatic scripts, testing the use of such information in a real task would be more illuminating. McIntyre and Lapata (2009) present a completely automatic story generation system based upon exactly the kind of co-occurrence information constructed in Chambers and Jurafsky (2009).

Automatic methods of generating stories would be very useful in educational applications. Story writing tasks are a great way of encouraging children to improve their writing skills and creativity. An automatic generation system could help in such situations by tracking the story written so far and providing suggestions regarding turns in the story. The task of generating stories using computers is also investigated for its nature as a good artificial intelligence task, because it involves considerable world knowledge which a system must use (Mateas and Sengers, 1999).

But stories, especially those written for children tend to have a more formulaic structure. So McIntyre and Lapata (2009) ask the following question: Given a large collection of children's stories, can we identify patterns in them automatically and how useful will this information be for generating new stories?

3.2.1 Construction of knowledge base from data

McIntyre and Lapata (2009) use the knowledge base for both the content of their stories as well as information for constructing story sequences. Therefore, McIntyre and Lapata (2009) incorporate two types of information into their knowledge base: a) entities and their associated actions and properties, b) closely related events and in what temporal order they are likely to appear.

Agents and actions This portion of the knowledge base is intended to convey what properties are associated with different entities and actions, and also which actions are likely to be performed by a given entity. For this purpose, McIntyre and Lapata (2009) parse a collection of children's stories and obtain predicate-argument relationships. They identify all verb-subject, verb-obj, verb-adverb, noun-adjective dependencies and use a mutual information metric to identify the dependencies that have high association values. For example, we would like to know if "dog-*subject_of*-bark" is a frequent construction.

We can denote each of these dependencies as a relationship between two words. In "dog-*subject_of*-bark", the words "dog" and "bark" are related by the subject relationship. For any two words w and w' and relation r, the task is to compute how much information is obtained from the combined knowledge of the words and their relationship, $count(w, r, w')$ as opposed to the individual descriptions of the two words and the relation, $count(w)$, $count(w')$ and $count(r)$. Mutual information provides a way to

quantify this property.

$$MI(w, r, w') = log \frac{P(w, r, w')}{P(w)P(r)P(w')} \quad (5)$$

However it is not always the case that all three w, r, and w' be independent. Once we know that $r =$ "$subj\_of$", we can identify that w must be a noun and w' a verb. Hence following Lin (1998), McIntyre and Lapata (2009) modify Eqn. 5 with the assumption that w and w' are conditionally independent given the relation r:[1]

$$MI(w, r, w') = log \frac{P(w, r, w')}{P(r)P(w|r)P(w'|r)}$$

Properties of entities and events and entity-event associations are ranked based upon this score and listed in the knowledge base. McIntyre and Lapata (2009) compute some additional associations so as to avoid certain problematic constructions being generated. For example, we could have a noun phrase such as "the boy" which is likely to occur both as the subject or object of a verb such as "saw". So we might obtain high association values for the dependencies *boy-subj_of-saw* as well as *boy-obj_of-saw*. However, a combination of the two in the *same* sentence is unlikely: *The boy saw the boy*. So one would need some measure of compatibility between the subject and object in a sentence. Similarly, with double-object verb constructions, one would like both objects to be compatible entities. *"I gave Mary a book"* would be a more frequent construction compared to *"I gave the dog a book"*.

As a check on such constructions, McIntyre and Lapata (2009) compute and store a score for ternary relationships as follows:

$$P(a_1, a_2|s, v) = \frac{\|s, v, a_1, a_2\|}{\|s, v, *, *\|}$$

Here a_1, a_2 denote the first and second arguments of the verb v and s stands for the subject. When a_2 is ϵ, the equation encodes the likelihood of a single object construction.

These components of the knowledge base will be involved in content planning for a sentence. But sentences cannot be constructed in isolation without any dependence on previous context of the story. There are two components of context that are necessary to consider for the smooth flow of a newly constructed sentence: a) previously mentioned entities and b) events that have occurred. For the event sequence model, McIntyre and Lapata (2009) learn likely *chains* of events involving a *common protagonist* using a similar technique as Chambers and Jurafsky (2009). McIntyre and Lapata (2009) do not have a model of entity co-occurrence. Rather, they make a simplistic assumption inspired by the Centering theory that either the entity in subject or object positions of the previous sentence will be the subject of the next sentence. It is in fact a very difficult task to pick entities to talk about without going off-topic and losing focus. On the other hand, always picking an entity from the previous sentence would create stories that have a routine structure. We return to a discussion of this aspect later in Section 3.2.4.

Event sequences McIntyre and Lapata (2009) follow the same approach as Chambers and Jurafsky (2009): they extract verbs with coreferent arguments and score pairs of verb-dependency links using mutual information. However, in contrast to Chambers and Jurafsky (2008), they do not use a temporal classifier to get precedence relationships. Rather they simply record the order in which the events occur in the example documents. This is done by defining the relation between two verbs in terms of both temporal order as well as a common protagonist. Mutual information values are computed for both the transitions–$<chase, obj>$ *precedes and coreferent with* $<run, subj>$ as well as $<run, subj>$ *precedes and coreferent with* $< chase, obj >$–the former is probably the more likely one. Here "precedes" is defined by whether the event was mentioned before the other one in the stories during training.

In the work by Chambers and Jurafsky (2008), a temporal classifier is used instead of document order because the latter might not always be indicative of the actual occurrence order of the events. For

[1]This definition is identical to the mutual information criterion used in Chambers and Jurafsky (2009). In Chambers and Jurafsky (2009), $P(w, r, w')$ was written as $P(r)P(w, w'/r)$.

	girl	fairy	wand	wish	giant
S1	X	S	—	—	—
S2	S	—	—	—	X
S3	—	S	O	O	—

Figure 3: Entity Grid representation for the example text in Section 3.2.2

example, a news report about an accident might first mention the accident and later provide a description of prior events leading to the accident. But children's stories have a simple structure and document order might be more or less indicative of the actual order of events. So it is likely that the simplification used by McIntyre and Lapata (2009) would be sufficient in this case.

3.2.2 Models for ranking stories

However, stories cannot be generated based only on the likelihood of entities and event sequences in them. The text must also be coherent and for stories, one would also desire that they be interesting. Therefore McIntyre and Lapata (2009) wish to integrate both aspects–coherence and interest–during story generation. In this section, we describe how scores along these two dimensions were obtained. In the next section, we detail how the different scores from the knowledge base, and coherence and interest models were combined to select the best stories.

Local coherence There have been several work investigating factors which influence coherence of written texts. Several data-driven methods to capture coherence also exist, one of which is the *Entity Grid* model (Barzilay and Lapata, 2005). The goal of these methods is to identify patterns from training documents that are indicative of coherence and proper sentence flow and use these for analyzing new texts. We return to a discussion of the Entity Grid and other data-driven coherence models in Section 4.1.1 when we detail the work of Soricut and Marcu (2006).

Barzilay and Lapata (2005)'s Entity Grid method is inspired by the Centering theory and focuses on coreference and salience properties of entities in text. It may be expected that the entity coreference patterns in coherent texts differ from those in texts which lack coherence. Barzilay and Lapata (2005) aim to capture this difference automatically by observing a large collection of coherent and incoherent articles. Coherent articles are just texts written by people and incoherent articles are generated by randomly permuting the sentences of coherent articles. We briefly describe the learning procedure for the Entity Grid method below.

Consider an example text containing three sentences $S1$ to $S3$.

(S1) The fairy appeared before the girl.
(S2) The girl wished to be freed from the giant.
(S3) The fairy waved her wand and granted the wish.

In this framework, every text is represented by a set of n rows corresponding to the sentences (here 3) and p columns one for each unique entity mentioned in the text. In our case, we would have five columns as shown in Figure 3.

The cell corresponding to the i^{th} sentence (row) and j^{th} entity (column) is filled with the given entity's grammatical role (S-subject, O-object and X-other) in that sentence. The absence of entity j in sentence i is recorded by a '$-$' in cell ij. Figure 3 shows the populated Entity Grid for our example sentences. In this *grid*, a column's entries from top to bottom reflect that entity's transitions in the text. The entity *fairy* is the subject of the first sentence, absent in the second and reappears as the subject of the third. Therefore between any two adjacent sentences, different types of transitions *SO, SX, O–, XX, –X...* can occur for the different entities. A total of $M = 16$ such transitions are possible including the $--$ transition.

Barzilay and Lapata (2005) record the total count of each of these transitions over the entire text. The proportion of each transition type among the total transitions is calculated. Given a collection of coherent and incoherent articles in a particular domain, these proportions are computed for the example texts and are used as features to train a discriminative model to predict the coherence score of the article.

McIntyre and Lapata (2009) train an Entity Grid model using a collection of fairy tales as positive examples; incoherent texts to use as negative examples are obtained by randomly permuting the sentences from the original articles.

Interestingness This is the second aspect that McIntyre and Lapata (2009) consider as important for scoring stories. But there have been no studies previously showing which linguistic properties of a text are indicative of how interesting it is.

McIntyre and Lapata (2009) elicited human judgements for the level of interestingness of 40 stories from a collection of Aesop's fables. McIntyre and Lapata (2009) then obtained several word-based features from the texts and measured their correlations with the human judgements.

Some were basic features such as number of tokens and types of nouns, verbs, etc. and counts of dependency relations such as subject-verb and verb-object. The MRC psycholinguistic database was also used to extract features. It contains several large word lists where words have been assigned scores reflecting different aspects such as familiarity, concreteness, imagery and meaningfulness. Imagery ratings, for example, were obtained from an experiment (Paivio et al., 1968) in which humans were asked to score words on a given scale depending on how quickly they could associate the word with some image. A word such as "apple" would receive a higher rating compared to the word "fact".

McIntyre and Lapata (2009) found that the number of objects was the best predictor of interest level-with a correlation value of about 0.2. The next best features were the number of noun tokens and noun types. The features from MRC database–concreteness, meaningfulness and imagery–were also good correlates of interest level of stories. Surprisingly, the number of adverbs or adjectives were uncorrelated with interest in this analysis. One might expect that such descriptive words add more excitement to the story, however it turns out that this is not the case.

The significantly correlated features were combined in a regression model to predict the interest level.

3.2.3 Generation of story possibilities and search process

With the knowledge base and the two models for local coherence and interest level in place, the task now is the generation of stories with a good combination of these aspects. McIntyre and Lapata (2009) use a beam search method which we describe in this section.

McIntyre and Lapata (2009) require the user to specify a starting sentence or set of entities. Consider that a sentence such as "The lion pounced on the rabbit." was obtained. As described before, the system assumes that the entity in subject or object position of the previous sentence would become the main entity of the next sentence. In this case, *lion* and *rabbit* are possibilities for the "subject" of the next sentence.

Next, a list of most associated verbs with the chosen entities is obtained from the knowledge base. This list is also ordered by considering which verbs are more likely to follow the previous event of "pounced". Therefore both the entities as well as previous event influence the next event in the story. These dependencies are shown in the first level in Figure 4. McIntyre and Lapata (2009) do not provide details about how the association scores for these two factors–what actions are likely for that entity and the likelihood of the event sequence–were combined. However, it appears correct that both factors would influence the choice of the next event. For our example story in Figure 4, let us consider that the verb *escape* is the most highly ranked by the combined criteria. To keep the search space small, only the top five events are kept in the search tree.

Next the system obtains subcategorization information (number and type of syntactic arguments) for each of the verbs in the story tree. For example, the verb *escape* could appear without objects and adverbs or in conjunction with them. This information is provided by a database collected in Korhonen et al. (2006). The most plausible set of five subcategorization templates are retained. The adverb, or object slots for a verb are then populated using the knowledge base and considering which lexical items

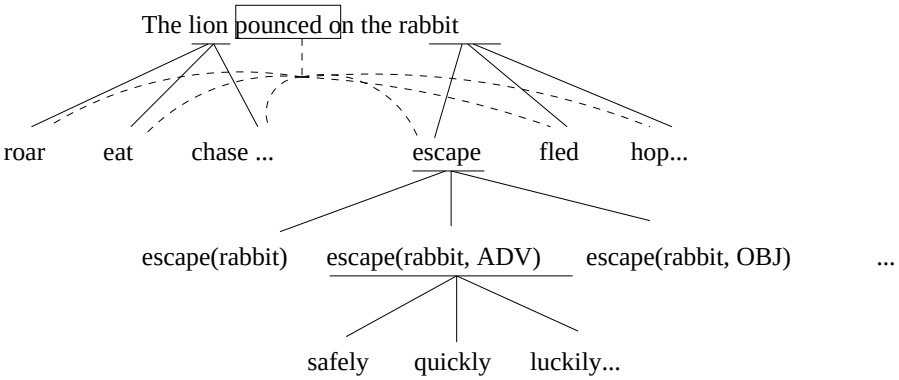

Figure 4: Generation of possible sentences to follow "*The lion pounced on the rabbit.*"

are very likely to fill these syntactic positions for that verb. This choice that McIntyre and Lapata (2009) make is bound to be problematic. We would pick out frequently co-occurring objects for that verb, without consideration of how they fit with the rest of the story, the already existing entities and past actions. Note that in this case, we would ideally like the object of escape to be "lion". But McIntyre and Lapata (2009) ignore prior context and so the stories produced are likely to not be very focused.

At this point we have abstract representations for several possible sentences for the next level of the story. Each of these is realized into a sentence by a language generation component (RealPro (Lavoie and Rambow, 1997)) which handles the other parameters such as choosing an appropriate tense , adding articles, etc. Several combinations are experimented and the sentences are scored using a trigram language model. The best scoring realization for each sentence is returned.

This generation process using the knowledge base produces several sentence choices at each step. Now the task is to find a sequence yielding a good story. McIntyre and Lapata (2009) aim to combine the information from the knowledge base together with the coherence and interest scores to find the best stories. They use a beam search method and keep at each level only the sentences corresponding to the top scoring 500 stories. They perform a study with human judges during development to identify which setup for scoring would be best–only coherence, interest or both. To obtain this information, McIntyre and Lapata (2009) generated a few stories by ranking according to each of these options and asked humans to rate the quality of the stories produced. The combination of both scores proved best from this study and forms the settings for their final model at test time.

Also, note from the previous section, that Entity Grid and interest models were trained on full text stories. During beam search, as the story is being built, we might have considerably fewer sentences (starting with two) for which we would like to obtain a score from these methods. Further, the final stories generated by McIntyre and Lapata (2009) are also only five sentences long. So the predictions from both these methods trained on longer articles may not be robust for considerably smaller *test* articles. McIntyre and Lapata (2009) specifically refrain from training the Entity Grid on short stories because they consider them less appropriate and use stories of more than 100 sentences. However, this could be a problematic choice. The distribution of lexical items as well as coreference features could be different in long versus short stories. It might be much better to define scores at the level of sentence-to-sentence transitions so that these scores can be summed up for the partial text as the search progresses. Recall that the Entity Grid simply used human-written articles as coherent examples. We could get finer level coherence judgements by considering that adjacent sentences in human-written articles are coherent and define features for pairs of sentences. We later describe how this exact modification is done to the Entity Grid model in the work of Soricut and Marcu (2006) for their search algorithm. The interestingness features are all word-based and harder to split to a lower level without labelled data at the same level, but since the training articles in this case were much smaller, four sentences on average, it is not very different from the test examples in McIntyre and Lapata (2009).

3.2.4 Results

McIntyre and Lapata (2009) train their story generation model on a collection of 437 fairy tales (`http://www.mythfolklore.net/andrewlang/`) each approximately 125 sentences in length. McIntyre and Lapata (2009) obtained *five* sentence stories from their search process one each for 10 *starting* or input sentences. Apart from the full model, two baselines were incorporated in the evaluation. One of these chooses the next event and entities at *random*. The other also does not use the coherence and interestingness models and simply chooses the *most plausible* next event and entity[2] from the knowledge base. Both these baselines generate a single story hypothesis. Human judges were asked to rate these stories on a scale from 1 to 5 on three aspects: *fluency*, *coherence* and *interest*. McIntyre and Lapata (2009) found that the stories produced by their full search-based model scored significantly better than the random and most plausible story baselines. This shows that multiple aspects such as coherence, interest and likelihood play a role in generation and a combination of these improves story quality.

However, the scores provided by humans for McIntyre and Lapata (2009)'s generated stories as well as the baselines are around 2 (maximum score is 5). Therefore the overall quality of the automatic stories is not high. The system performs generation starting at a very fine level, realizing sentences from entity and event specifications. This turns out to be very hard, most of the sentences even from the full model are not grammatical.

The choice that McIntyre and Lapata (2009) make to keep an entity from previous sentence in the next helps to maintain some focus in the text. But from the example stories in the paper, it is clear that linking all sentences in this way makes the story appear cliched. It is important to note that generating pronouns or other ways of referring to the same entity is another hard task and hence entities are simply repeated as such. The lack of a better notion of entity co-occurrence also shows up in other situations. For example, the system generates a predicate and argument such as "rescues the son", where there is no mention of "the son" previously. In fact, even without the definite article, the word "son" cannot be used except if a family or other entities provide a context for its use in the discourse.

Coming to the issue of whether event sequence information was useful, from an analysis of example stories in McIntyre and Lapata (2009), it does not appear that the co-occurrence clues are very robust for use in such applications. One of the baselines in McIntyre and Lapata (2009) as we described above used the most probable event associated with an entity and previous event for generating stories. Hence the chain of events from this approach, should give us some insight into the kinds of chains in the knowledge base. For the two examples provided in the paper, the sequence of verbs in the five sentences of the stories are *has* $\rightarrow$ *rounds* $\rightarrow$ *comes* $\rightarrow$ *wonders* $\rightarrow$ *meets* and *guards* $\rightarrow$ *rescues* $\rightarrow$ *beats* $\rightarrow$ *feels* $\rightarrow$ *hears*. One can immediately notice that when the system started with a very common less specific verb such as "has" or "comes", the co-occurrence information from the database would be highly unreliable for predicting the next event. But in applications such as generation, this would be a common case. Even in the other cases, the event co-occurrence information appears to be very noisy.

3.3 Discussion

In this section, we compare some properties of Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) and discuss how the approaches can be improved. We also describe how some of the issues that arise in the context of these two papers are handled by those which we discuss next in this report.

Domain dependence The idea of learning from example texts is based upon the intuition that *similar* texts would have noticable patterns. But, one could define similarity in many ways. Both Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) base their learning on texts from the same "genre", narratives: news articles and stories respectively.

On the other hand, similarity can be defined in terms of the "topic" of the text. For example, within news articles, those about accidents, would all have very similar content as well as organization: describing the location, cause, casualties, etc. Since the notion of same topic in addition to genre is more fine-grained, it is likely that we can learn more accurate information about text structure by examining

[2] Again, the combination of both entity-event and event-event association values is not specified

articles on the same topic. It would be interesting to investigate if the narrative structures learnt in Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) can be improved by learning entity and event co-occurrences on topic-tagged texts. The two papers which we discuss next, Soricut and Marcu (2006) and Sauper and Barzilay (2009), use topical similarity for choosing their training documents.

Combining multiple aspects In McIntyre and Lapata (2009), we came across the issue of how different properties seemed necessary to create good quality stories: plausibility, interestingness and coherence. Their results also show that when these aspects were combined during story search, the generated stories were given higher scores. This situation is likely to arise in any generation task. However, for McIntyre and Lapata (2009), the hypotheses generated from different settings of parameters during development had to be scored by human judges to find the best combination. Therefore, they could not try different settings or weights for the component models.

The two papers which we introduce next also deal with applications to generation and the problem of incorporating multiple aspects of text structure comes up in these. But both Soricut and Marcu (2006) and Sauper and Barzilay (2009) use automatic and cheap methods of evaluating their output during development time. We show how this fact enables them to explore a wider range of possibilities and directly train their models to find a way of combining different constraints.

Level of text structure Both Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) have focused on entities and events in the text. At sentence level, entities and associated events were learnt. Between sentences, event co-occurrence forms the primary link. In contrast, Soricut and Marcu (2006) and Sauper and Barzilay (2009) work with larger granularities of text: sentences or paragraphs. Their aim is to create coherent texts starting with such larger units of content.

In addition, Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) do not consider the fact that texts have a hierarchical structure: paragraphs, subtopics etc. For example, McIntyre and Lapata (2009)'s system has no notion of 'start' and 'end'. Stories are simply generated upto a length of five sentences. It is unlikely that they would have a plot or be meaningful as stories. Moreover, apart from the fact some entities from the previous sentence are repeated in the next, there is no attempt in McIntyre and Lapata (2009) to return to topics or entities mentioned in even earlier discourse. In other words, the component of global structure is missing in McIntyre and Lapata (2009)'s system. In Soricut and Marcu (2006), we introduce some models of local structure as well as those which incorporate some aspects of the global document structure. The focus of Sauper and Barzilay (2009) is to show exactly that having an idea of the overall structure and content of the document greatly aids the generation of articles.

Application needs While Chambers and Jurafsky (2009) do not perform any application-oriented evaluation, we saw how to use the same information in a generation task, McIntyre and Lapata (2009) had to tackle several other challenges: producing grammatical sentences, considering multiple possibilities, search algorithms, optimizing for multiple desired aspects, etc. Therefore, apart from producing good models of text structure, in order to use them successfully in applications, these other problems also need to be addressed. One of the main focus of the papers we discuss next is contributing efficient algorithms for specific generation tasks.

4 Models and algorithms for selecting and ordering content in generation tasks

So far, we have discussed approaches and tasks where the focus was learning and use of semantic knowledge. Next we discuss two papers which concentrate on text-to-text generation. The content is already available at the level of sentences or paragraphs. Consider most of the current day summarization systems. They do not generate new text for their summaries. Rather systems identify sentences from the source texts which contain important information and include these sentences as such for creating the summaries. At this stage, the summary is only a bag of sentences possibly coming from different portions of a source document or even different documents when summarizing a cluster of related articles. These sentences need to be subsequently ordered to make the text more readable for the user. Similarly, in natural language generation, a system might select a set of facts to represent in the output but a coherent ordering of these needs to be created before generating the text. For such applications as well, systems

need information about how humans organize content while writing.

The goal of the two papers we study here Soricut and Marcu (2006) and Sauper and Barzilay (2009) is to acquire knowledge of text structure for use in organizing content units. In Soricut and Marcu (2006), we see how properties of coherence between adjacent sentences can be learnt and using this information one can obtain benefits in the task of ordering a given bag of sentences to create coherent text. On the other hand, Sauper and Barzilay (2009) focus on the coherence of articles at the global level. They discuss how templates about subtopics in articles from a particular domain can be automatically obtained. Using these templates, they automatically generate Wikipedia articles in two domains. When the focus is on usability in a task, properties and challenges associated with the task also become important to solve. Both Soricut and Marcu (2006) and Sauper and Barzilay (2009) concentrate on efficient algorithms and techniques for their respective generation tasks.

4.1 Learning to order sentences coherently: Soricut and Marcu, 2006

One approach to address the task of sentence ordering is to develop metrics which indicate how coherent the flow between a pair of sentences is. This information can be used to decide which sentence pairs may appear adjacent in the output. Several data-driven models of text have been developed for this problem. Soricut and Marcu (2006) propose a new metric based on lexical statistics and also test whether existing approaches can be combined to yield better estimates of coherence.

However, given such *local* metrics to estimate the coherence of pairs of sentences, finding the ordering of a set of sentences such that the overall coherence score of the output is maximized is still hard. Althaus et al. (2004) proved this discourse ordering problem to be NP-complete by a reduction from the Traveling Salesman Problem. Consequently, methods to perform information ordering rely on search methods. Previous approaches address this problem through a greedy approach (Lapata, 2003) or by methods to find a local optimum (Mellish et al., 1998). However, if several errors are made during search, these errors might undermine the predictiveness of the coherence scores. To address this problem, Soricut and Marcu (2006) propose a compact representation of possible orderings of the input sentences using IDL (Interleave-Disjunction-Lock) graphs and present an A* search algorithm on these graphs to reduce complexity as well as the likelihood of search errors.

4.1.1 Coherence models

Soricut and Marcu (2006) combine three different methods to score coherence. Two of them are from prior work (Barzilay and Lee, 2004; Barzilay and Lapata, 2005). One of these (Barzilay and Lapata, 2005) is the Entity Grid method already introduced in Section 3.2.2 which uses coreference information. Barzilay and Lee (2004)'s approach, also called a "content model", is based on word patterns in the text. In this paper, Soricut and Marcu (2006) introduce a new method to compute coherence using word co-occurrence patterns, adapting a technique from statistical machine translation. In this section, we provide a brief description of what aspects of text structure are captured by each model and provide motivation for why we can expect better performance by combining these approaches.

Soricut and Marcu (2006) are interested in obtaining scores which reflect how naturally a given sentence follows after its immediately previous sentence. Sentences further back in the discourse are ignored. The coherence of the full text is defined in terms of the coherence level between its adjacent sentences.

Entity Grid As we described previously, Barzilay and Lapata (2005) focus on entities in the text and their coreference and salience properties.

The entity transition patterns present in the "grids" of documents from a given domain form the features for training this model. In Section 3.2.2, we described the 16 types of transitions that can occur in the grid between adjacent sentences, eg. SS, SX, O–... where S-subject, O-object, X-other noun phrase and - represents absence of the entity. The proportion of each transition type among the total transitions in the document were introduced as features in Barzilay and Lapata (2005) to predict the coherence of unseen articles. McIntyre and Lapata (2009) used the same setup of features for search during story generation. In Section 3.2.3, we discussed that such features defined over entire articles may not be suitable

during search stages when we want to compute the coherence of shorter articles.

Soricut and Marcu (2006) use the same Entity Grid features but define them at the level of adjacent sentences. For two sentences s_i and s_j, the features for the transition from s_i to s_j, $\phi(s_j|s_i)$ are the counts of different types of entity transitions for that ordering of the two sentences. Soricut and Marcu (2006) take pairs of sentences from the original document as positive examples (coherent sentence transitions) and non-adjacent sentence pairs in the article as negative examples. During training, they learn a weight vector w of length Q $(= 16)$, to predict the probability[3] that sentence s_{i+1} follows s_i under the Entity Grid coherence model.

$$p_E(s_{i+1}|s_i) = w \cdot \phi(s_{i+1}|s_i)$$

The overall probability of a text T with n sentences is then computed as follows:

$$P_E(T) = \prod_{i=1}^{n-1} p_E(s_{i+1}|s_i)$$

The Entity Grid approach is based entirely on noun phrase coreference patterns in the text. Other words in the training documents are ignored and the identities of the words themselves are never used. But the distribution of words are also important clues to how content is organized in documents. The two papers which we have already discussed, Chambers and Jurafsky (2009) and McIntyre and Lapata (2009), were strongly based upon word co-occurrences. Coreference and lexical information could be complementary clues and one would like to use them simultaneously for predicting coherence. So Soricut and Marcu (2006) combine the Entity Grid method with some lexical approaches. We describe these next.

Word co-occurrence Soricut and Marcu (2006) introduce two ways to compute the coherence between adjacent sentences using the word co-occurrence patterns in them. They apply an idea from machine translation to quantify the co-occurrence likelihood.

In machine translation, large amounts of source language sentences and their equivalent target language sentences (translations) are used to train "alignment" models to obtain the probability of observing a target language word as the translation of a source language word. Methods such as IBM Model 1 (Brown et al., 1993) based upon the Expectation Maximization (EM) algorithm are typically used for this purpose. Soricut and Marcu (2006) apply this idea to compute coherence of two adjacent sentences, ie., for measuring the likelihood of observing the words present in sentence s_{i+1} after the words in sentence s_i. For this task, they train the IBM alignment model using pairs of adjacent sentences rather than sentences from two languages. In this way, they obtain a table of probabilities, $p_t(s^x_{i+1}|s^y_i)$ where s^x_{i+1} is a word in sentence s_{i+1} and s^y_i is a word in the previous sentence s_i.

In the *forward* model, sentence s_{i+1} is assumed to be generated from words in sentence s_i. The probabilities of adjacent sentence transitions and full texts under this model are defined as follows.

$$p_{W_f}(s_{i+1}|s_i) = \prod_{j=1}^{|s_{i+1}|} \frac{\epsilon}{|s_i|+1} \sum_{k=0}^{|s_i|} p_t(s^j_{i+1}|s^k_i)$$

For each word in s_{i+1}, the probability of observing it after each of the words in s_i is computed and the average value is taken. The words in s_i also include a special *NULL* word. (In translation, words in target language sentence which do not align to any source sentence words are assumed to be aligned to *NULL*.) ϵ is a small constant probability value for observing a sentence of length $|s_{i+1}|$ after a sentence of length $|s_i|$.

$$P_{W_f}(T) = \prod_{i=1}^{n-1} p_{W_f}(s_{i+1}|s_i)$$

[3]Soricut and Marcu (2006) do not report which classifier they used for training the Entity Grid. I assume that they predict probability of membership in the class of coherent sentence pairs.

The probability of a text can also be measured under a *backward* model which assumes that words in sentence s_i are generated by words in sentence s_{i+1}.

$$p_{W_b}(s_i|s_{i+1}) = \prod_{k=1}^{|s_i|} \frac{\epsilon}{|s_{i+1}|+1} \sum_{j=0}^{|s_{i+1}|} p_t(s_i^k|s_{i+1}^j)$$

$$P_{W_b}(T) = \prod_{i=1}^{n-1} p_{W_b}(s_i|s_{i+1})$$

The Entity Grid and Soricut and Marcu (2006)'s model of word co-occurrences can be expected to combine to produce a better method. However, both of these methods are defined for pairs of adjacent sentences, and all pairs in a document are treated the same. They lack any notion of global structure. For example, suppose that we moved the paragraphs around in the document. The entity transition and word co-occurrence patterns are still preserved within paragraphs, the only changes occurring at the beginning and end of paragraphs. In this case, it is unlikely that Entity Grid or word co-occurrence model would be able to predict that this change has made the document much less coherent.

Content models Barzilay and Lee (2004) introduced a Hidden Markov Model (HMM) method to model text structure as a sequence of topics and transitions between them. This method called the "content model" can also capture some global aspects of text structure such as the beginning and end of documents. So Soricut and Marcu (2006) suppose that by combining this global information with the local models it might be possible to obtain an even better predictor of coherence scores.

This method is also based on word co-occurrence patterns. Barzilay and Lee (2004) cluster similar sentences from articles in their training corpus and each cluster represents a "topic". These topics are the states of the HMM for modeling that domain's articles.

The sentences clustered under each topic (hidden state) are used to compute a bigram language model for estimating the emission probabilities $p_e(s_i|h_i)$ for sentences s_i from that hidden state h_i.

The transition probability p_t between two states, h_i and h_{i+1} is computed as follows. Let c and c' be the clusters of sentences corresponding to these states respectively. Consider that the sentences in cluster c came from $D(c)$ unique documents in the training corpus. Let $D(c, c')$ be the number of training documents where some sentence from cluster c appears immediately before a sentence in cluster c'. Using this information about precedence relationships, the transition probability is computed as:

$$p_t(h_{i+1}|h_i) = \frac{D(c, c')}{D(c)}$$

The probability of a sequence of n sentences is given both by the likelihood of transitioning between topics p_t as well as the probability of a sentence being generated in particular topic state p_e. The following equations represent this computation for adjacent sentences and full text.

$$p_C(<s_{i+1}, h_{i+1}> \mid <s_i, h_i>) = p_t(h_{i+1}|h_i) \cdot p_e(s_{i+1}|h_{i+1})$$

$$P_C(T) = \max_{h_1 \ldots h_n} \prod_{i=0}^{n-1} p_C(<s_{i+1}, h_{i+1}> \mid <s_i, h_i>)$$

The probability of the text is computed as the maximum probability from the sequences of hidden states.

A loglinear representation for combining coherence models Soricut and Marcu (2006) combine the three coherence models described above in a loglinear framework. Overall, there are $M = 4$ feature functions, one each corresponding to the forward and inverse variations of the word co-occurrence model, the Entity Grid and the content model. Each feature function is the probability of the text T under a model $P_m(T)$ and is associated with a weight parameter λ_m. The probability $P(T)$ for the text under all the models taken together is written as follows:

$$P(T) = \frac{e^{\sum_{m=1}^{M} \lambda_m P_m(T)}}{\sum_{T'} e^{\sum_{m=1}^{M} \lambda_m P_m(T')}}$$

During search, we would like to find the ordering of sentences which maximizes this probability and hence the coherence. For all orderings, the denominator of the above equation would be the same. Hence the objective can be written as follows ignoring the normalization term.

$$\arg\max_{T} P(T) = \arg\min_{T}(-\sum_{m=1}^{M} \lambda_m \log P_m(T)) \tag{6}$$

These weights λ_m are tuned such that they improve the *quality* of the orderings. In Section 4.1.5, we describe the training procedure used by Soricut and Marcu (2006) to learn these weights and the metrics defining the quality of orderings.

All the coherence models in Soricut and Marcu (2006) are defined at the *level of adjacent sentences*. However, optimizing for the best order according to these *local* coherence metrics is hard because of the huge search space. Soricut and Marcu (2006) seek to tackle this problem by using IDL graphs, a framework which can represent the ordering possibilities in a very compact manner. They develop an A* search algorithm to use on these graphs to find good quality orderings. We first provide a brief description of IDL graphs and discuss how Soricut and Marcu (2006) represent the information ordering problem with these graphs. In Section 4.1.4, we detail their search algorithm.

4.1.2 Introduction to IDL (Interleave-Disjunction-Lock) graphs

In generation tasks such as sentence ordering, the set of possible hypotheses for the system to consider is very large. For n sentences, there are $n!$ possible orderings. It becomes difficult to represent all these possibilities and compare them. IDL (Interleave-Disjunction-Lock) graphs proposed by Nederhof and Satta (2004) provide a solution to this problem. IDL graphs enable:

(a) a very compact representation of the set of possible constructions

(b) efficient computation on this representation

IDL graphs can generate finite languages only and have equivalent finite state acceptor (FSA) representations. However, compared to FSAs, IDL graphs involve additional operations which enable IDL representations to be more compact than FSA. IDL graphs have a concatenation and a disjunction operation: the semantics of these operations are identical to FSAs. Two additional operations–*interleave* and *lock*–are also available. The *interleave* operator (represented as $\|$) is the one relevant for the ordering task and we illustrate its use through an example.

The *interleave* of *happily* and *play.piano* (the concatenation operator on *play* and *piano* keeps the precedence constraint that 'play' should appear before 'piano') is represented by the following expression (every IDL graph also has an equivalent IDL expression).

$$\|(happily, (play.piano))$$

The above expression generates strings where 'happily' can appear in any position in the sequence 'play.piano', ie., *happily play piano, play happily piano, play piano happily*. The expression is much more compact compared to the possible enumerations.

4.1.3 IDL graph representation for information ordering

Consider that we have three sentences α, β and γ in our document. Let the symbols $<d>$ and $</d>$ represent the beginning and end of the document. All possible permutations of α, β and γ within the document boundaries can be written using an interleave operation as follows:

$$E = <\text{d}> \cdot \|(\alpha, \beta, \gamma) \cdot <\text{/d}> \tag{7}$$

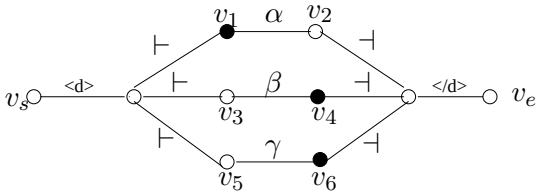

Figure 5: An IDL graph for ordering three sentences

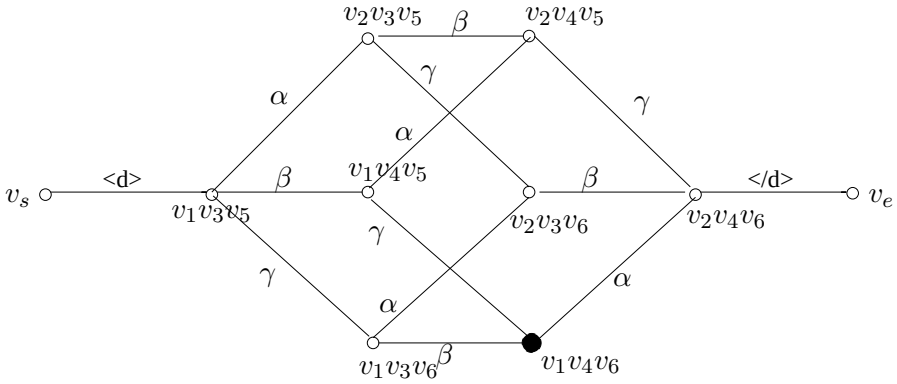

Figure 6: The unfolded IDL graph corresponding to Figure 5

The IDL graph for the expression in 7 is shown in Figure 5. It is considerably compact compared to a FSA representation of all the orderings. v_s and v_e represent the start and end states. The start of an k-argument interleave operation at a state is indicated by k outgoing $\vdash$-labelled arcs. The end of interleave is signaled by arcs labelled with $\dashv$ corresponding to each of the arguments all leading to a new state.

A procedure *unfold* is defined which expands an IDL graph into the finer level transitions as in the corresponding FSA. Formal details are available in Nederhof and Satta (2004) and Soricut and Marcu (2005). For illustration purposes, we provide a brief description. At the beginning of an interleave operation, all $\vdash$ edges must be followed simultaneously, reaching the set of states (v_1, v_3, v_5) for our example. After that, one move on the IDL graph can correspond to traversing any one of edges within the interleave operation. From state (v_1, v_3, v_5), the next possible state is one of (v_2, v_3, v_5), (v_1, v_4, v_5) or (v_1, v_3, v_6). These three transitions correspond to having added one of α, β and γ respectively to the ordering. In the state (v_1, v_4, v_6) boldfaced in Figure 5, β and γ have already been added to the ordering and α has still not been added.

The *unfold* operation handles this task of enumerating the set of next states from any given state. The fully unfolded graph for our example is shown in Figure 6. This uncompressed representation explicitly shows the path for each ordering of the sentences.

For the information ordering task, Soricut and Marcu (2006) need to keep track of the previous sentence included before the current state to compute the coherence score for transition to next sentence. Their approach also involves hidden variables corresponding to the content model. So, Soricut and Marcu (2006) split every state into more states. These new states are more fine-grained, also containing information about which input symbol (sentence) was last added to the ordering and corresponding to which hidden variable. For example, the state $v_1v_4v_6$ can be split into $(v_1v_4v_6, \gamma, h_i)$ and $(v_1v_4v_6, \beta, h_j)$ corresponding to which sentence γ or β was added to the ordering in the previous step (see Figure 6) and the associated hidden variables h_i and h_j. The transitions leaving these new states are the same as those leaving the unsplit state.

The complexity of searching the unfolded graph for the correct ordering is what one would like to avoid. Therefore the aim of Soricut and Marcu (2006) is to use the compressed IDL representations and during search, only selectively *unfold* the graph. The reduction in complexity and correctness are both dependent on the *heuristics* used to select the nodes to expand next. The design of a heuristic for an A* search algorithm is the focus of Soricut and Marcu (2006) and we describe this design next.

4.1.4 An A* search algorithm

A* search is a heuristic based search method and when an "admissibility" property is maintained, the search is guaranteed to lead to the optimal solution even if the search space is infinite. Consider that at a certain point during search, we have r possible nodes which we could include in the path and continue along. In the case of IDL graphs, these options correspond to nodes we could *unfold* to search further. In the A* algorithm, for each of these nodes r_i, a cost value $c(r_i)$ is computed. This total cost $c(r_i)$ is the sum of two components, $f(r_i)$ and $g(r_i)$. $f(r_i)$ is the cost up to the current node r_i and $g(r_i)$ is a heuristic future cost–an estimate of the cost from r_i to the goal state. A* search returns the optimal path to the goal state as long as $g(r_i)$ is *not* an overestimate of the future cost (Russell and Norvig, 1995). This property of $g(r_i)$ is called "admissibility".

For Soricut and Marcu (2006)'s task, an optimal path through the IDL graph corresponds to the ordering of sentences which has the maximum probability under a chosen coherence metric. Soricut and Marcu (2006) design an admissible heuristic for this search. Their idea is as follows. Given a particular state, they identify all the *future* sentences, ie., sentences that have not been added to the ordering as yet. In our example, at state $(v_1v_4v_6, \gamma, h_i)$, the future sentences are the symbols on the arcs along all paths from that node to the goal state. This set is called as F. From Figure 5, $F = \{\alpha, </d>\}$.

Soricut and Marcu (2006) also define a set of *conditioning* sentences C for the future sentence set F. This set C contains sentences that are likely to be immediately previous to the sentences in F during search. Therefore C includes all but one of the elements of F–the last sentence that might be encountered ($</d>$ in our example). C would also include the the most recent sentence added to the ordering (γ for our case when current state is $(v_1v_4v_6, \gamma, h_i)$). Therefore $C = \{\alpha, \gamma\}$ for our example.

For $g(r_i)$ to be admissible, it must not overestimate the cost to goal state from r_i. Therefore, for each of the future sentences $s_f \in F$, Soricut and Marcu (2006) assume that the conditioning or previous sentence $s_p \in C$ is the one giving the least transition cost for moving to s_f. In this way, for all the future sentences, an optimistic cost from previous sentence is estimated. By summing these up for the set F, Soricut and Marcu (2006) obtain a cost measure from current state r_i to the goal state which is guaranteed to *not* be an overestimate.

Since there are $M = 4$ coherence models in Soricut and Marcu (2006), all costs incorporate these models using the loglinear framework from Eqn. 6. Since the content model also involves hidden variables (set H), each of the sentences in F and C can be associated with a variable from H: an event is represented by $<p, h_k>$ where $p \in F$ or $p \in C$ and $h_k \in H$. The Entity Grid and word co-occurrence models would ignore the hidden variables. The future cost from r_i is estimated as follows.

$$g(r_i) = \sum_{s_f \in F} \sum_{m=1}^{M} \lambda_m \min_{\substack{h_i \in H \\ <s_p, h_j> \in C \times H}} -\log\ p_m(<s_f, h_i> \mid <s_p, h_j>) \tag{8}$$

The cost of a transition from event $<s_p, h_j>$ to $<s_f, h_i>$ is defined as the negative value of probability of coherent transition predicted from each of the M coherence models $p_m(<s_f, h_i> \mid <s_p, h_j>)$. If coherence is less, the cost is high.

A* search has its disadvantages. It guarantees that the optimal path will be found with an admissible heuristic, however, this performance comes at a significant memory cost. Multiple hypotheses and partial paths need to be maintained so that they can be explored if the current path turns out suboptimal. For ordering large bags of sentences, A* search becomes impossible to do. At this stage, some approximation needs to be done to reduce the time complexity. Soricut and Marcu (2006) introduce a beam search version of the algorithm (*IDL-100*, beam width of 100) to use for large input sizes. In this case, at each step during *unfolding* of the IDL graph, the states are ordered by total cost $c(r_i)$ and only the top 100 low cost options are kept for further search. However, Soricut and Marcu (2006) report that this beam search version also takes more than a minute to order sets containing 11 sentences on average (on a 3 GHz CPU). This time complexity turns out to be a significant disadvantage of Soricut and Marcu (2006)'s approach, however, it appears that Soricut and Marcu (2006) concentrate more on quality of the resulting orderings. We discuss this aspect further when we analyze their results in Section 4.1.6.

4.1.5 Learning parameters of the loglinear model

In this section, we describe how Soricut and Marcu (2006) learn the parameters λ_m for their loglinear model defined in Eqn. 6. The λ_m's should maximize the coherence of the ordering obtained. However, it would be difficult to enumerate all orderings in order to learn the weights. For such argmax operations defined over a compressed representation, Och (2003) introduced a *minimum error rate training* procedure, which has been widely used in machine translation where different knowledge sources are often combined in loglinear frameworks.

In this method, the actual distribution of all possible hypotheses (all orderings in our case) is approximated using a k-best list produced by a search process. Further, the training is targeted at reducing the error rate according to a metric which corresponds to the final evaluation criteria rather than maximizing the likelihood of the training data. The idea is that by basing the training on the final evaluation metric the resulting model would be tuned to generate candidates that rank better according to the specified metric. This metric must be automatically computable in order to evaluate the candidates during training. The training begins with manually set weights for the parameters. The k-best candidates are obtained using search with these model parameters. The obtained orderings are evaluated according to the evaluation metric provided and these scores are used to train and update the parameters. A new n-best list is obtained using the updated model and merged with the existing n-best list and used for training. The process continues until the n-best lists do not change between iterations.

In Soricut and Marcu (2006)'s evaluation setup, the test set is a collection of documents. The sentences in each document form one input for testing. The original ordering in the test document is taken as the reference or gold standard ordering. Therefore, Soricut and Marcu (2006) are able to use two automatic metrics to define the utility of a produced ordering by comparing it to the original ordering. One is *Kendall's Tau* which measures the correlation between two sets of rankings. It is defined as:

$$\tau = \frac{n_c - n_d}{\frac{1}{2}n(n-1)} \tag{9}$$

n_c is the number of *concordant* sentence pairs in the system-produced ordering where the relative ranking of the two sentences in the pair agree with their gold standard ordering. n_d or *disconcordant* pairs represent the cases when the rankings of the two sentences in a pair in the proposed ordering are different from the gold standard. $\frac{1}{2}n(n-1)$ is the number of unique pairs of sentences. With fewer concordant pairs, τ would be small. $\tau = -1$ indicates completely opposite rankings with respect to the original. $\tau = 1$ represents a perfect match with the actual ordering.

The other utility metric used in Soricut and Marcu (2006) is *BLEU*. BLEU is standardly used for Machine Translation (MT) evaluation for comparing a sentence produced by a translation system with that produced by a human. In MT, BLEU is computed as the overlap in ngrams between the two sentences. Soricut and Marcu (2006) use BLEU differently, to compare the sentence sequence proposed by the system with that in the gold standard. It is computed as follows. Suppose that the sequence $1...n$ represents the sentence numbers in the gold standard ordering. Let σ represent the ordering produced by the system, therefore σ would be some permutation of $1...n$. This evaluation metric measures the number of unigrams, bigrams, trigrams, four-grams of sentence numbers from the actual ordering that are preserved in the proposed one. A geometric mean of these four measures is the metric that Soricut and Marcu (2006) use.

4.1.6 Results

Soricut and Marcu (2006) evaluate their approach on two corpora of articles. One is a collection of news articles on earthquakes from the Associated Press. The other set contains reports on airplane crashes from the National Transportation Safety Board. These documents contain 11 sentences on average. Training and test sets consist of 100 documents each. Given the set of sentences from a document, the task is to generate a coherent ordering of these sentences.

Search efficiency To estimate search errors, Soricut and Marcu (2006) measure the proportion of times the generated ordering from their system had a lower probability under a coherence model compared

to the probability of the actual ordering. For all the coherence metrics that were tested: Entity Grid, content models and IBM word co-occurrence models, even the beam search version of A* with a width of 100 (IDL-B100) gave no search errors on the test documents. Soricut and Marcu (2006) find that if the previously proposed greedy approach for ordering (Lapata, 2003) is used, one is likely to make around 46% to over 90% search errors for the same coherence models.

But Soricut and Marcu (2006) note that even the beam search version takes more than a minute. Then full A* search without pruning could take much longer. Therefore, it appears that Soricut and Marcu (2006)'s approach does not scale well even with inputs of about 11 sentences. But for several generation tasks the time to produce output should be low regardless of input size. For example, in a system to summarize search results long time delays might be inacceptable. Hence Soricut and Marcu (2006)'s search method has the disadvantage of being costly in terms of runtime.

Usefulness of model combination Soricut and Marcu (2006) obtain orderings using beam search with different coherence metrics and measure the Tau and BLEU scores of generated orderings with the original document order. Individually, the content models, and the new word co-occurrence approach introduced in Soricut and Marcu (2006) obtain better results compared to the Entity Grid approach.[4] Coreference information by itself appears to be not as robust as word co-occurrence patterns. Both word co-cooccurrence and content models turn out to have similar performance: Kendall's correlation of about 0.4 with the original ordering. Combining the all three with the loglinear model was found to produce even better results compared to each of the individual models: the τ score for the combination is around 0.5. Further Soricut and Marcu (2006) show that these best scores were obtained when the utility metric for training loglinear parameters matched the final evaluation metric. For example, when τ metric was used for training, better τ scores were obtained compared to no error rate training as well as training using a different metric than the final evaluation, in this case, BLEU.

Soricut and Marcu (2006)'s results show clearly that coherence models can be combined to produce better results as opposed to using them individually. This finding is extremely useful and can be applied to small-sized inputs without consideration of search quality. For example, sometimes only very short summaries are required from systems. A 100 word summary might contain five sentences on average. In such cases, an approach of enumerating and ranking all permutations might just be sufficient and an optimal solution is also guaranteed. Using the right coherence metric becomes more important in such cases, and according to Soricut and Marcu (2006)'s finding, a combination of different metrics might be better to use compared to a single coherence score. Also, if the generation method is offline and not time critical, one might be assured of good results using Soricut and Marcu (2006)'s approach.

But coming to the issue of generating long documents, we need to rethink if such measures of local coherence are sufficient for good performance on this task. Long documents tend to be arranged in paragraphs each with a particular subtopic. Perhaps this idea could be captured in some way. The next paper that we discuss, Sauper and Barzilay (2009) address this task of generating long articles.

4.2 Mining subtopic structure from texts: Sauper and Barzilay, 2009

Consider that we want to generate a biography for a person who does not already have an entry in an encyclopedia. One way to do so, would be to pose a query containing the person's name to a search engine and summarize the most relevant documents to create the biography. Prior work in Biadsy et al. (2008) have adopted this approach. A classifier is trained to identify sentences from these retrieved documents that are most likely to contain biographical information (facts about birth, career, life). The top scoring sentences are chosen and ordered using some coherence metric to create the biography. However, such sentence selection and ordering approaches can only aim to create short coherent articles. Biadsy et

[4]These results are different from those reported in Barzilay and Lapata (2008) where the Entity Grid appears to outperform content models on the same corpus. However the evaluation metrics are different for the two studies. Soricut and Marcu (2006) measure the Kendall's correlation between the original and system produced order. In contrast, Barzilay and Lapata (2008) generate about 20 random permutations of the original article and each permutation is paired with the original ordering to form a test example. Barzilay and Lapata (2008) measure as error rate, the proportion of times, a permuted ordering was classified as more coherent than the actual ordering. Soricut and Marcu (2006)'s experiments show that Barzilay and Lapata (2008) made a simplistic approximation of possible orderings.

al. (2008) generate summaries of about 100 words. Long articles have a clear structure consisting of a sequence of subtopics. For example, long biographies would have a paragraph about birth and early life, followed by a section on occupation and so on. For automatically generated long biographies, it would be best to have a similar structure. Presenting a large collection of relevant sentences without any notion of subtopics might turn out less informative and incoherent in such a situation.

Sauper and Barzilay (2009) aim to tackle this problem by automatically learning patterns of subtopics that appear in Wikipedia articles from two domains–diseases and American film actors. The central hypothesis in this work is that in the case of generating long articles, a template outlining the subtopic structure of existing articles should be useful for creating new articles. The content for creating such articles is already available on the World Wide Web and excerpts from these documents can be used to fill out the templates for new articles. The aspect that Sauper and Barzilay (2009) focus on is the right selection of *paragraphs* for each subtopic such that the articles are both informative as well as coherent. This setup is considerably different from the work we have discussed so far. In McIntyre and Lapata (2009)'s story generation system, even entities and their actions at subsentence level were generated using their automatically constructed knowledge base. Soricut and Marcu (2006) focused on the arrangement of sentences.

While each subtopic can be filled with appropriate content individually, it may arise that the chosen excerpts for the different subtopics might not comprise a globally coherent article. There might be information repeated in the paragraphs chosen for different subtopics. Therefore one would like to take an approach which selects good content for all subtopics, while at the same time maintains the globally desired properties. To address this issue, Sauper and Barzilay (2009) propose a structured prediction approach and show how consideration of the global fit results in better quality articles.

4.2.1 Template creation

Wikipedia articles are a very good starting point for learning subtopics. These articles are demarcated explicitly with section headings. Sauper and Barzilay (2009) use the prevalence and arrangement of these topic headers in existing articles for creating templates. We detail their template creation method below.

Even though similar sections might appear in the articles in a domain, they may not have exactly the same headings. Overviews of diseases might have a section listing its different types but these sections could be named differently in the articles eg. "Types of arthritis" and "Types of cataracts". Some articles may use a longer heading such as "signs and symptoms", others using only the word "symptoms". So Sauper and Barzilay (2009) create clusters of similar section headings to represent the subtopics. They use a partitioning approach for clustering. All the headings start out in the same cluster. At each step, a cluster is chosen and bisected. Each heading is represented by a vector of tf*idf weights of its constituent words during this procedure and cosine similarity metric is used to make clustering decisions. If there are k sections on average in the articles for a domain, k clusters (subtopics) are chosen in this process. The most frequent heading in each cluster is added to the template as a subtopic title. For a domain such as diseases, the subtopics obtained could be "symptoms", "causes", "diagnosis" and "treatment".

Next ordering information is added for the subtopics in the template. From the original article, we know the preferred or most frequent ordering for a pair of headings. Using this information, we can find pairwise preferences for ordering clusters of headings that were obtained. Sauper and Barzilay (2009) then use a majority ordering algorithm (Cohen et al., 1998) to obtain a global ordering for the clusters. The method proposed in Cohen et al. (1998) is a greedy approach which aims to maximize the pairwise preferences satisfied in the global ordering.

4.2.2 Extraction of candidate excerpts and features for ranking

Given the template for disease overview articles, for example, an article on a new topic, "Cataract", can be generated as follows. We have the article name, ie., the disease name as well as the section headings from the template. Sauper and Barzilay (2009) use a search engine to obtain the relevant documents for each subtopic using the article title and subtopic name as queries. For example, the query "Cataract causes" is used to obtain content for the "causes" subtopic. All paragraphs from the documents corresponding to the top 10 pages of search results are chosen as candidate excerpts for that subtopic.

However, not all paragraphs obtained from these documents would be relevant content for the subtopic. Sauper and Barzilay (2009) define a number of content selection features to identify those which are most relevant and informative. These features include counts of each unigram and bigram in the paragraph, the first word in the paragraph, and the counts of its sentences, pronouns, etc. They also include a feature to reflect the centrality of an excerpt. (An excerpt highly similar to other excerpts is probably very informative and contains important content.)

A naive approach to create an article at this stage is to rank the excerpts for each subtopic for relevance and choose the best paragraph in each case. Sauper and Barzilay (2009) however, note that doing so could lead to articles where the excerpts for different subtopics have overlapping content thereby leading to less coherence as well as lower informativeness. Therefore one would like to keep informative paragraphs for each subtopic, at the same time, minimize redundancy over the entire article. Sauper and Barzilay (2009) aim to achieve this by encoding these constraints in an Integer Linear Programming (ILP) framework.

4.2.3 ILP constraints for article creation

We first discuss how Sauper and Barzilay (2009) set up the problem as an ILP task. In the next section, we detail the perceptron training algorithm used to learn the weights for ranking the excerpts for a subtopic.

Consider that there are k subtopics in a particular template, we represent these as t_1, t_2, ..., t_k. Let the candidate excerpts for subtopic t_j be e_{j1}, ...e_{jr}. Each of these excerpts, e_{jp} is associated with a vector of features $\phi(e_{jp})$ as described in the previous section. Suppose that w_j is the weight vector for scoring the excerpts belonging to subtopic t_j. A ranking of excerpts can be obtained by ordering them by the scores values, $\phi(e_{jp}) \cdot w_j$. Let us consider that a lower rank indicates a more informative excerpt. Now, the goal is to bring in redundancy considerations at a global level, while at the same time selecting minimally ranked excerpts for each topic.

Sauper and Barzilay (2009) incorporate both these factors using the following ILP constraints. Let us add a superscript to each excerpt indicating its rank for the subtopic it belongs to. For example, e_j^2 indicates the 2^{nd} most highly ranked excerpt for subtopic j. The objective is to minimize the sum of the ranks for excerpts chosen for the subtopics in the final article. Let $x_{jl} = 1$ be an indicator that the l-ranked excerpt gets chosen for the j^{th} topic (excerpt e_j^l), otherwise the value of the variable is zero. This objective can be written as follows for a domain with k subtopics each with r excerpts.

$$\min \sum_{j=1}^{k} \sum_{l=1}^{r} l \cdot x_{jl}$$

Since only one excerpt can be chosen for each subtopic, the following constraint is added.

$$\sum_{l=1}^{r} x_{jl} = 1 \quad \forall\, j \in \{1...k\}$$

Now for reducing global redundancy, Sauper and Barzilay (2009) add a constraint that the cosine similarity between any pair of excerpts in the generated article should not exceed a value of 0.5.

$$(x_{jl} + x_{j'l'}) \cdot sim(e_j^l, e_{j'}^{l'}) \leq 1 \qquad \forall\, j, j' \in \{1...k\} \qquad \forall\, l, l' \in \{1...r\}$$

When the similarity value $sim(e_j^l, e_{j'}^{l'})$ is greater than 0.5, then this constraint would allow only one of the excerpts to be selected in the output, ie, x_{jl} and $x_{j'l'}$ cannot both take a value of one in this case. However, if the similarity value is less than or equal to 0.5, then both can be included.

Sauper and Barzilay (2009) use a branch and bound approximation algorithm to solve the ILP.

Solving this ILP provides the desired properties of informative excerpts as well as a global fit in the generated articles. During test time, one could first obtain a ranking of the excerpts for each subtopic using a weight vector trained for that subtopic. The ILP could then be applied to make the final selection for all excerpts. Sauper and Barzilay (2009) however, use the ILP procedure during training as well with an aim to jointly learn the weight vectors for different subtopics. We describe this training procedure in the next section. The output of training is a set of weight vectors, one for each subtopic for ranking

its excerpts. However, these weights are tuned such that they provide the best performance during ILP decoding. During test, Sauper and Barzilay (2009) apply the weight vector for each topic for producing a ranking of its excerpts. Then ILP is used to finalize the best excerpts for all the subtopics.

4.2.4 Training a structured prediction model

The training is done in an online fashion using a perceptron ranking algorithm (Collins, 2002). Initially the weight vectors for all subtopics w_j are set to zero. The algorithm iterates over each document in turn until convergence or a fixed number of iterations. At a particular time step, the perceptron update procedure is as follows.

Consider that a particular training document is used at this step.

(i) First, the ranking of excerpts for every subtopic of that document are predicted using the current weight vectors.

(ii) Then ILP is used to optimize the selection across all excerpts for low global redundancy. This produces a prediction of the best excerpts for all subtopics considering both informativeness and coherence.

(iii) The selected excerpt for each subtopic top_j is now evaluated by measuring its similarity with the gold standard content for that subtopic gs_j from the original article. If the cosine similarity value between these texts is below a threshold value of 0.8, then the weight vector for that subtopic w_j is updated. Note that the update is performed based upon the predictions obtained jointly, ie., after ILP decoding rather than the rankings based only upon excerpt scores for relevance to individual subtopics. The update is done as follows. Consider that there are k subtopics. Let $\phi(top_j)$ and $\phi(gs_j)$ represent the feature vectors of top_j and gs_j respectively.

for $j = 1...k$ **do**
  if $sim(top_j, gs_j) < 0.8$) **then**
    $w_j = w_j + \phi(gs_j) - \phi(top_j)$
  end if
end for

After convergence, the final set of weight vectors are returned. These vectors are specific to each subtopic and are representative of content importance for that subtopic. At the same time, these parameters were updated by accounting for global coherence as well. Sauper and Barzilay (2009) hypothesize that such a method would produce weight vectors well-suited for decoding using ILP at test time.

4.2.5 Results

Sauper and Barzilay (2009) apply their method for the generation of articles in two domains–*American Film Actors* and *Diseases*. The articles on these topics in Wikipedia (around 500 for 'diseases' and 2000 for 'film actors') were used for training and testing purposes. The average number of topics in these articles is around four.

Sauper and Barzilay (2009) use as gold standard the actual article in Wikipedia. The articles from a system whose content match with the actual articles to a greater extent are more informative. Sauper and Barzilay (2009) compare the system-produced articles with the gold standard using ROUGE (Lin and Hovy, 2003), a tool to measure the n-gram coverage of a given text with respect to a reference text. ROUGE is the standard automatic evaluation metric used in summarization for comparing system-produced summaries to human-written ones. The fscore from unigram overlaps was used by Sauper and Barzilay (2009) for evaluation.

Usefulness of structure information One of the aspects that Sauper and Barzilay (2009) want to demonstrate is the usefulness of knowing the structure of the article, in this case, the usefulness of the templates. For this purpose, they generate articles using two baselines that do not use structure information. One of these simply uses the article's title to search for relevant documents. The document corresponding to

the top-ranked search result is then output as the final article after truncating for a required number of subtopics (equal to the average subtopics for that domain).

The other baseline uses a classifier to distinguish domain versus non-domain content. This approach is the same as the previous work in biography production which we descibed earlier (Biadsy et al., 2008). However, in contrast to Biadsy et al. (2008), Sauper and Barzilay (2009) use paragraphs as the basic units for article construction. For building a classifier for predicting whether an excerpt is relevant to domain of "disease" articles, the positive examples are paragraphs from Wikipedia articles on diseases. Some articles from the web on diseases are chosen and the paragraphs in these which have very little similarity to the Wikipedia articles on diseases comprise the set of negative examples. The trained model is used to predict scores for the excerpts mined during test time. The top ranking paragraphs (very similar paragraphs are filtered to keep only one) up to the required number are selected as article content.

Sauper and Barzilay (2009) compare these baselines with a template-based approach. In this setup, the ILP constraints were not used. Rather a classifier was trained for each subtopic to rank its excerpts. The best excerpt was chosen for each subtopic. Since this approach using the template produced articles with significantly higher scores compared to the baselines, Sauper and Barzilay (2009) confirm that structure information was greatly helpful for article generation.

Impact of redundancy removal Sauper and Barzilay (2009) also compare the *disjoint* template based approach from the previous section with their full model which also uses ILP constraints for lowering redundancy. ILP was used for this model both during training and test time. Sauper and Barzilay (2009) find that the articles produced using their full model was significantly better than disjoint selection of excerpts for each subtopic. By lowering the chance of redundant information, more useful content could be added to the articles.

Sauper and Barzilay (2009) do not provide a comparison with an approach where ILP is not used for joint update during training. That is, training could only involve learning the weight vectors for each subtopic individually. These can be used for a first step prediction during test, followed by the ILP optimization. It is therefore unclear whether the joint training algorithm has been beneficial. However, in prior work Snyder and Barzilay (2007) carried out such an experiment incorporating a global decoding procedure during training and found that results were better compared to using only individual rankers during training and applying the global decoding only during test time.

Human evaluation Sauper and Barzilay (2009) also do not present an evaluation of the readability of their articles. However, Sauper and Barzilay (2009) report a summary of human edits made to some of their articles which were posted to Wikipedia. The articles produced by the full model only were used and had been on Wikipedia for at least six months. During this time, two-thirds of these articles were promoted to full-article status. A number of edits were made to these articles by humans. Most frequently, these edits involved formatting and addition of links to other articles. Some were focused on correcting grammatical errors. Overall the pattern of edits is indicative that most content was very relevant to the topic and that these articles were of considerably good quality for an automatic approach.

The fact that there were some grammatical errors suggests that one approach to improve Sauper and Barzilay (2009)'s approach would be to also consider the linguistic quality aspects of the candidate paragraphs while ranking them. Since the excerpts are taken from articles on the web, they are likely to have embedded links and other errors. Even if the content of the excerpt is good, it may not be well-written or readable. To tackle this problem, one could use scores from language models to reflect the quality of sentences as well as metrics such as those we discussed in Soricut and Marcu (2006) to score the local coherence of the excerpts. These scores can be easily incorporated in Sauper and Barzilay (2009)'s model as features during ranking. In this way, even better quality articles can be generated.

4.3 Discussion

Coherence at different levels Both Soricut and Marcu (2006) and Sauper and Barzilay (2009) aim to generate coherent articles but focus on coherence at different levels. The methods in Soricut and Marcu (2006) captured properties of flow between adjacent sentences. Here we discussed that entity

coreference, co-occurring words and topic continuity were useful predictors of local coherence. When adjacent sentences had frequently co-occurring words or topics, the texts tended to be better organized.

On the other hand, in Sauper and Barzilay (2009), subtopics or paragraph level coherence is considered. They use the ordering of subtopics in existing wikipedia articles to find the most likely sequence for that domain. While generating new articles, a coherent flow of subtopics is ensured using this information. Also, Sauper and Barzilay (2009) are not ordering paragraphs from the same article as in the evaluation setup of Soricut and Marcu (2006). They extract content from different documents from the web and use these for creating new articles. If the paragraphs selected for an article have redundant content across them, this aspect would also make the article less coherent. Hence, another central focus in Sauper and Barzilay (2009) is on lowering redundancy in the generated articles.

Properties of coherence models At first look, the Entity Grid approach appears to be domain independent. Centering and other entity coherence theories are not specific to certain types of texts. One would expect that coreference patterns in coherent articles would be similar at least in the same domain. Barzilay and Lapata (2008) tested the Entity Grid method on the earthquakes and airplane accidents corpus. Barzilay and Lapata (2008) note that the models when trained on the same topic texts provided the best performance for that domain. But, by training on both texts together, Barzilay and Lapata (2008) show that good performance can be obtained for both texts. So the Entity Grid method can be used in a topic-independent way.

The word co-occurrence model would also have some topic independence properties. Soricut and Marcu (2006) do not specify the training corpus for their experiments. But it is likely that with a large collection of texts in the same domain, word statistics would work well for most of the topics in that domain. Moreover, it is also the case that the word co-occurrence estimates might turn out robust only with very large amounts of data. Such large training sets also restricted to the same topic might be difficult to obtain in practice. Hence using large generic text collections might be the most suitable training approach in this case.

On the other hand, we have the HMM model which models subtopics and their transitions and is dependent on topic of the articles. As expected, content models work very poorly on out-of-topic texts (Barzilay and Lapata, 2008). It is not possible to use them in a domain independent way.

Combining coherence models In contrast to Soricut and Marcu (2006)'s method, Elsner et al. (2007) define a generative model for combining content models with the Entity Grid approach. Recall that in the HMM approach of Barzilay and Lee (2004) the emission probabilities from a hidden state were estimated using a bigram language model trained on the cluster of sentences corresponding to that state. Elsner et al. (2007) define these emission probabilities using the Entity Grid. Their model has the advantage of a direct integration of the two coherence methods. On the other hand, Elsner et al. (2007)'s assumption of content models as the basic structure might be problematic.

Content models directly capture the underlying structure of the training documents, so it is possible that they can only be used for very formulaic texts. Barzilay and Lee (2004) report varying performance of their content model approach for different domains. For example, content models could be trained and used to predict the coherence of earthquake reports with very good results. All these documents would comment on the location, casualties and relief efforts. However, consider articles dealing with drug-related offenses. These articles have a less rigid structure: the offenses might have come up under different circumstances and the nature of the offenses themselves could be quite different. For such articles, content models turned out less useful. Local coherence statistics would probably provide more flexibility in such cases. While it is possible that Soricut and Marcu (2006) can learn a good balance between local and global models for each domain in terms of weight parameters in their loglinear model, this might be harder to do in Elsner et al. (2007)'s approach. Elsner et al. (2007) only evaluate their model on a single domain and here their results are identical to Soricut and Marcu (2006). The performance on more diverse domains is yet to be investigated for Elsner et al. (2007)'s approach.

Data for ordering tasks In fact, the data used for ordering experiments certainly needs more attention. Soricut and Marcu (2006) use the earthquake and airplane accidents corpus which have been widely used

in other approaches as well. But these documents come from the same organization, so the structure might not vary much across the articles. With experiments on such specific data, it is difficult to get an idea of how well these methods would perform on a different collection, even of earthquake reports. Elsner et al. (2007) note that in the airplane accident reports, nearly 50% of the training documents start with the same two sentences–"This is preliminary information, subject to change, and may contain errors. Any errors in this report will be corrected when the final report has been completed." Most methods might simply observe this pattern and use it for predictions which gives an unrealistic estimate of their performance. Therefore it is important to test systems on a variety of domains.

It is important to also remember that in most situations, an ordering may be needed for sentences that are not the entire set of sentences from an article. For example, multidocument summaries are created using sentences from multiple source articles. These sentences need to be properly ordered to maximize the readability of the summary. Such inputs could be more difficult for systems. Lapata (2003) note that one of their best features, noun co-occurrence performs well on the document ordering task but was only as good as a random baseline when tested on summaries. Hence using more realistic datasets or system output should be a goal for future studies.

5 Conclusions and future work

We have presented a survey of recent attempts to model text structure in an unsupervised manner. We briefly review some aspects of these models and suggest some areas for future work.

Importance of entity coherence The use of entity coherence as an indicator of text structure is a common idea in most of the work we have discussed.

In Chambers and Jurafsky (2009) and McIntyre and Lapata (2009), coreference between participants in events is used to define event relatedness. Chambers and Jurafsky (2008) also show that if coreference was not considered and we simply scored verbs by their frequency of co-occurrence in documents, the performance on the narrative cloze task was much worse. Both these approaches have taken a global view of entity coherence in that they consider events in different portions of a document as related when they have coreferent participants.

In the context of Soricut and Marcu (2006)'s work, we reviewed the Entity Grid method which is a shallow approach to capture the predictions of the Centering theory. Here entity coreference patterns were used to learn how texts are organized locally at the level of adjacent sentences. McIntyre and Lapata (2009) also use the idea of entity coherence and repeat entities from the current sentence in the next sentence to maintain focus in the story. These ideas conform to a more local view of entity coherence as in Centering theory.

Only in Sauper and Barzilay (2009), do we not see the use of a notion of entity coherence. Sauper and Barzilay (2009) concentrate entirely on global ordering of subtopics. They assume that they start with locally coherent paragraphs of text. So for global ordering they do not use entity coherence.

Incorporating rhetorical relations We discussed in Section 2 that discourse relations are also hypothesized by linguistic theories to be an important component of coherence. So one direction for future work is investigating how the unsupervised text structure models can be combined with discourse relations. Automatic discourse parsers could be used for this purpose (Marcu, 2000; Blair-Goldensohn et al., 2007; Sporleder and Lascarides, 2008; Pitler et al., 2009).

For example, the semantic structures learnt by Chambers and Jurafsky (2009) and McIntyre and Lapata (2009) only encode temporal relations between the events. Another relation that could be added to the event structure is "causality". For example, the event "convict" causes one to be "jailed". If such relations are also present in the schemas, they would be useful information for text analysis in addition to temporal relations between the events.

Also for the sentence ordering task, local models of organization discussed in Soricut and Marcu (2006) can be used together with discourse relation information. In this way, sentences could be ordered by also considering the likely sequence of discourse relations in human-written texts.

Use in applications Among the approaches that we have surveyed, those focusing on organization had a

more application-oriented approach. We have discussed the use of Soricut and Marcu (2006) and Sauper and Barzilay (2009)'s methods for generation tasks. Content models which we introduced in the context of Soricut and Marcu (2006)'s work have also been shown to be useful for summarization (Barzilay and Lee, 2004; Fung and Ngai, 2006). Entity Grid model has also been used for evaluating the quality of machine generated summaries and for predicting whether a text would be easy or difficult for humans to read (Barzilay and Lapata, 2008).

The use of semantic knowledge from text in actual applications is less clear. The stories produced by McIntyre and Lapata (2009)'s generation system lack focus and any meaning as actual stories. The usefulness of the schemas learnt for news text (Chambers and Jurafsky, 2009) is also not explored. It would be interesting to see the use of such schemas and ideas of event relatedness in applications such as summarization and question answering. During these tasks, schemas can guide the system to provide additional information related to user query or information need. One could also consider the possibility of generating stories by instantiating schema-like information. Multiple related schemas can be combined into a story by filling in participant slots appropriately. This approach would have the advantage of creating stories with a much more clear plan and global structure.

References

R.P. Abelson and R.C. Schank. 1977. *Scripts, plans, goals and understanding: An inquiry into human knowledge structures*. Lawrence Erlbaum Associates.

E. Althaus, N. Karamanis, and A. Koller. 2004. Computing locally coherent discourses. In *Proceedings of ACL*, pages 399–406.

C. F. Baker, C. J. Fillmore, and J. B. Lowe. 1998. The berkeley framenet project. In *Proceedings of COLING-ACL*, pages 86–90.

R. Barzilay and M. Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of ACL*.

R. Barzilay and M. Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

R. Barzilay and L. Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of NAACL-HLT*, pages 113–120.

F. Biadsy, J. Hirschberg, and E. Filatova. 2008. An unsupervised approach to biography production using wikipedia. In *Proceedings of ACL-HLT*, pages 807–815.

S. Blair-Goldensohn, K. McKeown, and O. Rambow. 2007. Building and refining rhetorical-semantic relation models. In *Proceedings of HLT-NAACL*, pages 428–435.

P.F. Brown, V.J.D. Pietra, S.A.D. Pietra, and R.L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

N. Chambers and D. Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-HLT*, pages 789–797.

N. Chambers and D. Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of ACL-IJCNLP*, pages 602–610.

T. Chklovski and P. Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of EMNLP*, pages 33–40.

W. Cohen, R. Schapire, and Y. Singer. 1998. Learning to order things. In *Proceedings of NIPS*, pages 451–457.

M. Collins. 2002. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proceedings of ACL*, pages 489–496.

M. Elsner, J. Austerweil, and E. Charniak. 2007. A unified local and global model for discourse coherence. In *Proceedings of NAACL-HLT*, pages 436–443.

P. Fung and G. Ngai. 2006. One story, one flow: Hidden markov story models for multilingual multidocument summarization. *ACM Transactions on Speech and Language Processing*, 3(2):1–16.

T. Grenager and C. Manning. 2006. Unsupervised discovery of a statistical verb lexicon. In *Proceedings of EMNLP*, pages 1–8.

B. Grosz, A. Joshi, and S. Weinstein. 1995. Centering: A framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.

A. Korhonen, Y. Krymolowski, and T. Briscoe. 2006. A large subcategorization lexicon for natural language processing applications. In *Proceedings of LREC*.

M. Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of ACL*, pages 545–552.

B. Lavoie and O. Rambow. 1997. A fast and portable realizer for text generation systems. In *Proceedings of the fifth conference on Applied Natural Language Processing*, pages 265–268.

C. Lin and E. Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of HLT-NAACL*.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL*, pages 768–774.

W.C. Mann and S.A. Thompson. 1988. Rhetorical structure theory: Towards a functional theory of text organization. *Text*, 8.

D. Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26(3):395–448.

M. Mateas and P. Sengers. 1999. Narrative intelligence. In *Proceedings AAAI Fall Symposium on Narrative Intelligence*, pages 1–10.

N. McIntyre and M. Lapata. 2009. Learning to tell tales: A data-driven approach to story generation. In *Proceedings of ACL-IJCNLP*, pages 217–225.

C. Mellish, A. Knott, J. Oberlander, and M. ODonnell. 1998. Experiments using stochastic search for text planning. In *Proceedings of International Conference on Natural Language Generation*, pages 97–108.

M.J. Nederhof and G. Satta. 2004. IDL-expressions: A formalism for representing and parsing finite languages in natural language processing. *Journal of Artificial Intelligence Research*, 21(1):287–317.

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, page 167.

A. Paivio, J.C. Yuille, and S.A. Madigan. 1968. Concreteness, imagery, and meaningfulness values for 925 nouns. *Journal of experimental psychology*, 76(1 Pt 2):1–25.

E. Pitler, A. Louis, and A. Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. In *Proceedings of ACL-IJCNLP*, pages 683–691.

R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. Joshi, and B. Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of LREC*.

V.I.A. Propp. 1968. *Morphology of the folktale*. University of Texas Press.

J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, et al. 2003. The timebank corpus. In *Corpus Linguistics*, volume 2003, page 40.

S. Russell and P. Norvig. 1995. *Artificial intelligence: A modern approach*. Prentice Hall.

F. Salanger-Meyer. 1990. Discoursal movements in medical English abstracts and their linguistic exponents: A genre analysis study. *Interface: Journal of Applied Linguistics*, 4(2):107–124.

C. Sauper and R. Barzilay. 2009. Automatically generating wikipedia articles: A structure-aware approach. In *Proceedings of ACL-IJCNLP*, pages 208–216.

B. Snyder and R. Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *Proceedings of NAACL-HLT*, pages 300–307.

R. Soricut and D. Marcu. 2005. Towards developing generation algorithms for text-to-text applications. In *Proceedings of ACL*, page 66.

R. Soricut and D. Marcu. 2006. Discourse generation using utility-trained coherence models. In *Proceedings of COLING-ACL*, pages 803–810.

C. Sporleder and A. Lascarides. 2008. Using automatically labelled examples to classify rhetorical relations: An assessment. *Natural Language Engineering*, 14:369–416.

A. Wray. 2002. *Formulaic language and the lexicon*. Cambridge University Press.