A Summariser based on Human Memory Limitations and Lexical Competition

Yimai Fang
Computer Laboratory
University of Cambridge
15 JJ Thomson Avenue, CB3 0FD, UK
Yimai.Fang@cl.cam.ac.uk

Simone Teufel
Computer Laboratory
University of Cambridge
15 JJ Thomson Avenue, CB3 0FD, UK
Simone.Teufel@cl.cam.ac.uk

Abstract

Kintsch and van Dijk proposed a model of human comprehension and summarisation which is based on the idea of processing propositions on a sentence-by-sentence basis, detecting argument overlap, and creating a summary on the basis of the best connected propositions. We present an implementation of that model, which gets around the problem of identifying concepts in text by applying coreference resolution, named entity detection, and semantic similarity detection, implemented as a two-step competition. We evaluate the resulting summariser against two commonly used extractive summarisers using ROUGE, with encouraging results.

1 Introduction

Kintsch and van Dijk (1978) (henceforth KvD) present a model of human comprehension and memory retention which is based on research in artificial intelligence, experimental psychology and discourse linguistics. It models the processing of incoming text or speech by human memory limitations, and makes verifiable predictions about which propositions in a text will be recalled by subjects later. It has been very influential, particularly in the 1980 and 1990s in educational (Palinscar and Brown, 1984; King, 1992) and cognitive (Paivio, 1990) psychology, and is still today used as a theoretical model of reading and comprehension (Baddeley, 2007; Zwaan, 2003; DeLong et al., 2005; Smith, 2004). It has also been used for improving education, particularly for the production of better instructional text (Britton and Gulgoz, 1991; Pressley, 2006), and for teaching humans how to read for deep comprehension (Coiro and Dobler, 2007; Duke and Pearson, 2002; Koda, 2005; Driscoll, 2005) and to summarise (Hidi, 1986; Brown et al., 1983).

In the summarisation community, the model has been commended for its elegant and explanatory "deep" treatment of the summarisation process (Lehnert, 1981; Spärck Jones, 1993; Endres-Niggemeyer, 1998), but has not lead to any practical prototypes, mainly due the impossibility of implementing the knowledge- and inference-based aspects the model relies on.

We present here an implementation of the model, which attempts to circumvent some of these problems by the application of distributional semantics, and by modelling the construction of the coherence tree as a double competition (firstly of concept partners for word forms, secondly of attachment sites for propositions).

In the KvD model, a text (e.g. Figure 1) is converted into propositions (see Table 1) which have one functor and one or more arguments. The functor can be taken either from a fixed list of grammatical relations (e.g. IS A; AT; BETWEEN; OR) or an open class-set of so-called concepts, (e.g. BLOODY; TEACH). Arguments can be concepts or proposition numbers. Proposition numbers express embedded semantic structures (e.g. #9 in Table 1). Kintsch et al. (1979) assumed that this tranformation is performed manually; they were able to train humans to do so consistently.

> A series of violent, bloody encounters between police and Black Panther members punctuated the early summer days of 1969. Soon after, a group of black students I teach at California State College, Los Angeles, who were members of the Panther Party, began to complain of continuous harassment by law enforcement officers.

Figure 1: First two sentences from the example paragraph *Bumperstickers* by KvD (1978).

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 732–741,
Gothenburg, Sweden, April 26-30 2014. ©2014 Association for Computational Linguistics

No.	Proposition
Cycle 1	
1	series (encounter)
2	violent (encounter)
3	bloody (encounter)
4	between (encounter, police, Black Panther)
5	time: in (encounter, summer)
6	early (summer)
7	time: in (summer, 1969)
Cycle 2	
8	soon (#9)
9	after (#4, #16)
10	group (student)
11	black (student)
12	teach (speaker, student)
13	location: at (#12, Cal State College)
14	location: at (Cal State College, Los Angeles)
15	is a (student, Black Panther)
16	begin (#17)
17	complain (student, #19)
18	continuous (#19)
19	harass (police, student)

Table 1: Propositions for Figure 1.

The KvD algorithm is manually simulated in their work, but is described in a mechanistic manner that should in principle lend itself to implementation, once propositions are created. Propositions form a tree where a proposition is attached to another proposition with which they share at least one argument; attachment higher in the tree is preferred. The tree is built incrementally; blocks of propositions, each of which roughly corresponding to one sentence, are processed in cycles. After each cycle, a process of "forgetting" is simulated by copying only the most salient propositions to the short-term memory (STM). This selection is performed by the so-called leading edge strategy (LES), which prefers propositions that are attached more recently and those attached at higher positions. This algorithms mirrors van Dijk's (1977) model of textual coherence.

When choosing an attachment site for proposition, arguments which are currently in STM are preferred. A resource-consuming search in long-term memory (LTM) is only triggered if a proposition cannot be attached in STM; in that case a bridging proposition is reintroduced into the tree.

The KvD model can be used to explain human recall of stories, and can also to create a summary of a text. The most natural way for a human to summarise from scratch is to replace propositions with so-called macropropositions, and the KvD model prefers this style of summary creation. An example for macroproposition is a statement that generalises over other propositions. This results in a more abstract version of the text. However if for any reason it is not possible to create macropropositions (for instance due to lack of deep knowledge representation), a summary can also be created in a simpler way based only on the propositions contained in the text. In that case, the selection criterion is the number of cycles a proposition has remained in STM.

There are three main stumbling blocks in the way of an implementation of the KvD model:

1. The automatic creation of propositions from text, and of summary text from summary propositions;
2. The automatic creation of concepts from words (including coreference resolution);
3. The creation of macropropositions, which would require sophisticated knowledge representation and reasoning.

We present a fully automatic version of the KvD model based on the following assumptions:

1. Current parser technology allows us to reconstruct the compositional semantics of the text well enough to make the KvD model operational, both in terms of creating propositions from text, and in terms of creating reasonably understandable output text from propositions (even if not fully grammatical).
2. We model the lexical variation of how a concept is expressed in a text probabilistically by semantic similarity and coreference resolution. This creates a competition between plausible expressions for argument overlap.
3. Our core algorithm is modelled as two competitions: (a) the competition between concept matches as mentioned in the point above; and (b) the competition between possible positions in a tree where a proposition could attach.
4. We also observed that KvD's method of choosing the tree root in the first processing cycle, and to never change it afterwards unless texts are truly incoherent (resorting to multiple trees), is too limiting, in particular in combination with their LES. Texts can have topic changes and still be perfectly co-

herent, particularly if they are longer and less linearly structured than the examples used by KvD. We therefore experiment with more flexible root choice strategies.

We have nothing to say on the third and biggest obstacle, the creation of macropropositions. Nevertheless, the experiments presented here test whether our hypotheses 1 – 4 are strong enough to provide our summariser with useful information concerning the discourse structure of the texts. We test this by comparing its performance to that of two current state-of-the-art summarisers, which instead rely on the sole use of lexical information. A psychologically-motivated summariser such as ours should be evaluated by comparison to abstractive, i.e., reformulated human summaries, rather than by comparison to extractive summaries. We do so using ROUGE, an evaluation framework that supports such comparisons (Lin and Hovy, 2003).

The structure of the paper is as follows. In the next section, we will detail our implementation of the KvD model, with particular emphasis on the creation of propositions, probabilistic concepts, proposition attachment, and root choice. In Section 4, we will present experiments comparing our summariser against two research extractive summarisers, MEAD and LexRank. We also test how our inventions including similarity-based concept matching and root choice strategy contribute to performance. We compare to related work in Section 3, and draw our conclusions in Section 5.

2 Our implementation of KvD

Figure 2 shows the structure of our summariser. The *Proposition Creation* module transforms surface text to propositions with the aid of a grammatical parser. Recall that in the original KvD model (shown as “Human (KvD)”), propositions are generated manually. Apart from such, our implementation follows the KvD algorithm as closely as possible. The core of this algorithm is the *Memory Retention Cycle* in the centre of the figure.

A cycle begins with the detection of coherence between the new propositions and the current STM content. This results in a hierarchy of all so-far processed propositions called the *Coherence Tree*. Propositions are attached to the tree by a variety of strategies, as explained in Subsection 2.2.

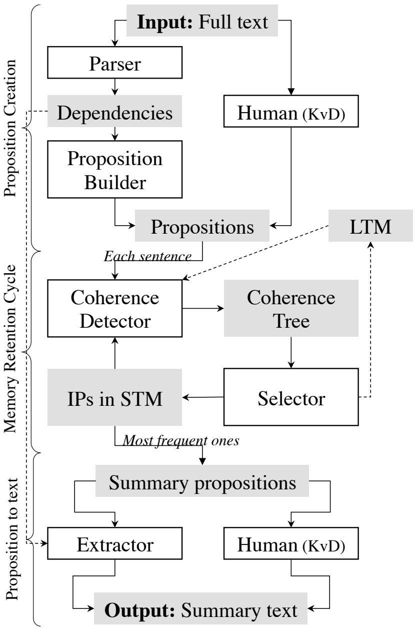

Figure 2: Framework of the summariser.

At the end of each cycle, important propositions (IPs) are selected by the *Selector*, stored in STM, and thus retained for the next cycle, where they are available for new incoming propositions to attach to. The selector is a full implementation of KvD’s LES, which also updates the recency of propositions reinstantiated from the LTM.[1] Less important propositions leave the cycle and go into the LTM, which is conceptually a secondary repository of propositions to provide the “missing links” when no coherence between the STM and the incoming propositions can be established.

After the text is consumed, a propositional representation of the summary is created by recalling the propositions that were retained in STM most frequently. The summary text is then either created manually (in the KvD model), or in our implementation, as a prototype, automatically by extracting words from the parser’s dependencies.

2.1 Proposition builder

We aim to create propositions of comparable semantic weight to each other. This is a consequence of our decision to recast KvD as a competition model (as will become clear in subsection 2.2), because by defining propositions as blocks of arguments they should contain a similar number of

[1] KvD implied this in the last cycle of the *Bumperstickers* paragraph, by placing the two reinstantiated propositions below #37, though they are older than #37.

meaningful arguments to ensure similar potential for overlap.

To achieve suitable granularity of propositions, we aggregate information spread out over several grammatical dependencies, and exclude semantically empty words from participating in argument overlap. We use Stanford Parser (Klein and Manning, 2003), and aggregate subjects and complements of a predicate into a single proposition. Active and passive voices are unified; clauses are treated as embedded propositions; controlling subjects of open clausal complements are recovered.

Some predicates are not verbs, but nominalised verbs or coordination. For instance, KvD model the phrase " *encounters between police and Black Panther Party members* " as BETWEEN (ENCOUNTER, POLICE, BLACK PANTHER). Producing such a proposition instead of two separate ones BETWEEN (ENCOUNTER, POLICE) and BETWEEN (ENCOUNTER, BLACK PANTHER) is advantageous, because this single proposition provides a strong connection between POLICE and BLACK PANTHER which cannot be derived from other dependencies.

However we lack a subcategorisation lexicon that provides information about how many arguments a preposition like "*between*" takes. Therefore we scan conjoined prepositional phrases, aggregate the objects, and attach them to the governors of the prepositional phrases. In this example, the resulting preposition is ENCOUNTER (POLICE, MEMBER). The word "*between*" is excluded because it is semantically empty and may interfere with overlap detection.

We take care to detect and exclude semantically empty material. For instance, the *empty semantic heads* in noun phrases such as "a series of" and "a group of" are detected using a list of of 21 words we collected, and treated by redirecting the dependencies involving the empty heads to the corresponding content heads. In this treatment, the relation between an empty head and its content head is not entirely erased, but encoded as a general modifier relation.

2.2 Probabilistic concept matching

The notion of argument overlap in KvD's model is sophisticated in that it "knows" which surface expressions (pronouns, synonyms, etc) in text refer to the same concept. Concept mapping is the task of forming equivalence classes of surface expressions; each concept then corresponds to one such equivalence class. The KvD model, because it simulates concept mapping and proposition attachment in parallel, conceals some of the choices that a fully automatic model has to make.

Given current technology, concept mapping can only be performed probabilistically. We use the Stanford coreference resolution, named-entity detection (to extend coreference detection to non-same-head references, e.g. mapping "*the tech giant*" to "*Apple Inc.*"[2]); and to find synonymy or at least semantic relatedness, we use a well-known measure of semantic similarity, namely Lin's Dependency-Based Thesaurus (Lin, 1998). We are not committed to this particular measure, but it empirically performed best out of the 11 we tried; especially it outperformed WordNet path-based measures. Note however that only the 200 most similar words for each word are provided by this tool. The similarity measure is normalised by relative ranking to provide the probability that an expression refers to the same concept as another expression. We use WordNet (Miller, 1995) for derivationally related forms (to solve e.g. nominalisation). This establishes the first competition, the one between concept matches.

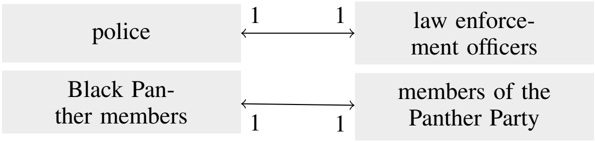

Figure 3: KvD's concept matching.

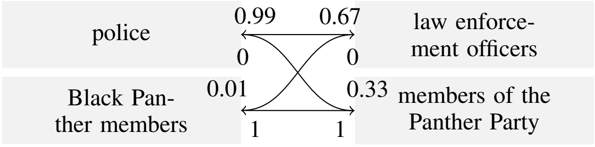

Figure 4: Probabilistic concept matching.

Modelling concepts probabilistically has its implication for the next task: finding the best attachment site for a proposition. Let us explain this with an example. Notice that in the example text in Figure 1, "*police*" (from #4, in the first sentence) and

[2] A WordNet synset is defined for each named-entity type; here "*giant*" is connected to its hypernym "*organization*" via "*enterprise*".

"*law enforcement officer*" (from #19, in the second sentence) refer to the same concept POLICE. Figure 3 illustrates how this is handled in KvD's model, where intelligent concept matching establishes with 100% certainty that the two strings refer to the same concept. Certainty about the argument overlap then enables them to later attach #19 to #4. In their model it is important whether a matching proposition is found in STM or LTM: If the only proposition that mentions "*police*" (#4) is no longer in STM when the proposition containing "*law enforcement officer*" (#19) is processed, and for any reason the other arguments in #19 (i.e. STUDENT) cannot find overlaps either, KvD find no concept match in STM and know therefore, again with full certainty, that an LTM search must be triggered[3], which in this case leads to the successful recall of #19 for #4 to attach to.

Figure 4 illustrates the corresponding situation in our model, where #4 with "*police*" is in LTM, the probability of a concept match between "*law enforcement officer*" and "*police*" is 66.7%, whereas that of a match with "*members*", which is in STM, is 33.3%. The probabilistic concept matching cannot provide enough certainty to single out #4 because of full argument overlap. The probabilities of concept match have to act as a much weaker filter in our model, and all previous propositions have to be considered as potential landing sites for #19. In particular, we do not know whether a concept match within STM is "good enough", or whether a LTM search is needed. There is, in this case, a competition between a weak match in STM (the direct vicinity) and a strong match in LTM (further away), which will hopefully result in a successful match between "*police*" and "*law enforcement officer*". In other words, we always have to search for matches in both repositories.

After obtaining the graph of interrelated expressions, the competition between landing sites for each proposition takes place, whereby higher positions are preferred. This double competition is a core aspect of our model.

2.3 Choice of root

The KvD model almost always maintains the root determined in the first cycle (either by overlap with title concepts or by coverage of the main clause of the first sentence). The model introduces multiple roots if a text is totally incoherent, namely when propositions cannot be attached anywhere and therefore a forest of disjoint trees has to be developed. This strategy does not generalise well to longer texts with topic changes, for example newspaper texts with anecdotal leads. Although these texts are perfectly coherent, KvD cannot treat them appropriately.[4]

Our more flexible rooting strategy is run once in each cycle, assessing whether any of the current root's children in the working memory would make a better root. In case of a root change, the edge between the old and the new root is reversed, and the old root becomes a child of the new root. Then we perform the same strategy on the new tree until no root change is needed.

We denote the current root as i, and a new root candidate (a child of i) j. J is the set of descendants of j (inclusive of j), and I the set of all nodes V excluding J, i.e. $I = V \setminus J$. Then nodes in J will be promoted after the root change, while those in I will go one level deeper. Since edge weights, i.e. attachment strengths, are asymmetric, we denote the weight for j being a child of i as $w_{i,j}$, and $w_{j,i}$ for the reversed attachment. Each node v also carries a weight $x_v = m_v \cdot a^{d_v}$, where m_v is a memory status factor (e.g. $m_v = 1$ if v is in STM, 0.5 if otherwise), $0 < a \leq 1$ is an attenuation factor, and d_v is depth of v in the tree. To decide, we evaluate

$$s = w_{j,i} \sum_{v \in J} x_v - w_{i,j} \sum_{v \in I} x_v \qquad (1)$$

If $s > 0$, the root change is permitted.[5] This evaluation makes root change easier if the edge in question favours i being a child of j, or there are more important nodes that can benefit from the change, and vice versa.

An example of such a root change taken from the *Bumperstickers* is given in Figure 5 (refer to Table 1 for proposition contents). As the central topic of the text changes from the encounters to

[3] KvD only mentioned retrieving embedded propositions as LTM search rarely happens, but the goal is the same as here: to establish overlap.

[4] In our scenario the situation can barely ever arise where absolutely no proposition attachment is possible, as the probabilistic concept mapping is usually able suggest some concept match, albeit with small probability.

[5] In case when multiple candidates are permitted, the one with the highest s is chosen.

that the identity of Panther Party members are actually the author's students, the summariser recognises this change after reading one more sentence, by flipping the edge connecting #3 and #14.

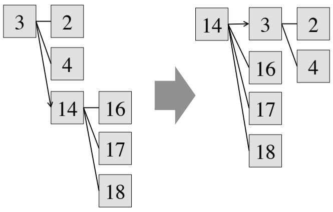

Figure 5: Tree before and after a root change.

3 Related Work

One of the dilemmas in summarisation research is how "deep", i.e. semantics-oriented, a summariser should be. Shallow analysis of lexical similarity between sentences and/or the keywords contained in sentences has lead to summarisers that are robust and perform very well for most texts (Radev et al., 2004; Dorr and Zajic, 2003; Carbonell and Goldstein, 1998). The methods applied include a random-surfer model (Mihalcea and Tarau, 2004; Radev, 2004), a model of attraction and repulsion of similar summary sentences (Carbonell and Goldstein, 1998). There are statistical models of sentence shortening (Knight and Marcu, 2002). While much work in summarisation has concentrated on multi-document summarisation, where the main challenge is the detection of redundant information, the summariser presented here is a single-document summariser.

However, researchers have been attracted by deeper, more symbolic and thus more explanatory summarisation models that use semantic representations of some form (Radev and McKeown, 1998) and often rely on explicit discourse modelling (Lehnert, 1981; Kintsch and van Dijk, 1978; Cohen, 1984). The problem with template-based summarisers is that they tend to be domain-dependent; the problem with discourse structure-based summarisers is in general that they require knowledge modelling and reasoning far beyond the capability of today's state of the art in artificial intelligence. Rhetorical Structure Theory (Mann and Thompson, 1987) provides a domain-independent framework that takes local discourse structure into account, which has lead to a successful prototype summariser (Marcu, 2000). This summarisation strategy does not however look at the lexical content of the propositions or clause-like units it connects, only at the way how the connection is performed.

The summariser presented here is a hybrid: its core algorithm is symbolic, but its limited powers of generalisation come from a semantic similarity metric that is defined via distributionally derived probabilities. Because its core processing is symbolic and based on a simple semantic representation, it is possible to derive an explanation based on the coherence tree and the propositions selected from it. There are some similarities to the idea of summarisation via lexical chains (Barzilay and Elhadad, 1997), as both methods trace concepts (as representatives of topics) across a document. The KvD model arguably uses more informative meaning units, as it is based on the combination of concepts within propositions, rather than on concept repetition alone.

A different, related stream of research looked at the automatic detection of coherence in text. Graesser et al (2004) present a coherence checker based on over 200 coherence metrics, including argument overlap as in KvD. Barzilay and Lapata (2008) use a profiling of texts akin to Centering theory to rank texts according to their coherence. It would be interesting to combine their notion of entity-based coherence with KvD's notion of argument overlap.

4 Experiments

We now perform two experiments. The first tests the contribution of our concept matcher and root change strategy on a small document set we have collected, and compares against two research summarisers. In the second experiment, we test the performance of our summariser on a much larger and standard dataset.

We will use the intrinsic evaluation strategy of comparison to a gold standard. Human judgements would be the most credible, but as a cheap alternative, we use ROUGE-L (Lin, 2004), which has been shown to correlate well to human judgements. For each sentence, ROUGE-L treats it as a sequence of words, and finds the longest common subsequences (LCSs) with any sentence in a gold standard summary. The score is defined as the F-measure of the precision and recall of the LCSs.

The next question is how the gold standard summaries used in ROUGE are defined. Because our summariser is deep and has a fine granularity, it should be compared against human-written summaries on a variety of texts.

For the first experiment, we have collected from volunteers 8 human abstractive summaries for each of the 4 short scientific articles or stories we found in Kintsch and Vipond (1979) (average length: 120 words), and 4 for each of 2 longer political news texts (average length: 523 words). The volunteers were instructed to condense the text to 1/3 of its length for the short texts, and to 100 words for the longer ones. They were also instructed not to paraphrase, but to use the words in the text as much as possible. This was because no summariser in this experiment has a paraphrasing ability. Nevertheless, not all subjects followed this instruction strictly.

For the second experiment, we use the DUC 2002 dataset (Over and Liggett, 2002). There are 827 texts from news media, of a variety of topics and lengths, among which our script is able to extract titles and contents of 822 documents. We use the provided single document abstractive summaries, which are of 100 words in length each, as gold standard summaries. A few of the documents are selected in multiple clusters and therefore have multiple summaries; all of them are used in evaluation.

We compare our summariser against a baseline constructed with the first n words from the original text, where n is the summary length as defined above, and two summarisers: MEAD (Radev et al., 2004) is a research summariser which uses a centroid-based paradigm and is known to perform generally well over a range of texts. LexRank (Radev, 2004) uses lexically derived similarities in its similarity graph of sentences, sharing the same idea of sentence similarity with MEAD. Note that both summarisers are extractive.

We illustrate what our summaries look like in Table 2, where we asked the summariser to give us summaries as close to 20 and 50 word summaries as possible, with Table 3 showing the underlying propositions. In contrast, MEAD can only extract sentences as-is (thus not as flexible in length), and does not have meaning blocks like our propositions.

Encounters between police and Black Panther members. Students to complain of harassment. Automobiles Panther Party signs glued to bumpers.
Bloody encounters between police and Black Panther members punctuated the summer days of 1969. Students to complain of continuous harassment by law enforcement officers. They receiving many traffic citations. Automobiles with Panther Party signs glued to their bumpers. I to determine whether we were hearing the voice of paranoia or reality.

Table 2: Summaries produced by our summariser.

3	encounters (between: police; between: Black Panther members)
16	to complain (students; of: harassment)
34	with: Panther Party signs (automobiles)
35	glued (#34; to: bumpers)

Table 3: Summary propositions for the first summary above.

We create summaries for all three summarisers following this procedure: We provide sentence-split texts and their headlines (not needed by LexRank), and run the summarisers in such a way as to produce a summary of the same length as stipulated for the standard summaries. Our summariser controls word count precisely; we require MEAD to produce summaries close to the length (allowing variations), and for LexRank we allow it to go beyond the limit by less than one sentence and then discard the exceeding part in the sentence with the lowest salience.

The results of Experiment 1 are summarised in Table 4. As is well-known from similar experiments, it is hard beating the first n baseline due to the fact that journalistic style (in the long texts) already puts a summary of each text first. It is slightly surprising that this effect also holds for the short texts (literary style). It is of note that our KvD summariser beats both MEAD and LexRank on this dataset, which is shelved away during development, with statistical significance on the long texts: the 95%-confidence interval of ours is 0.403 – 0.432, and that of MEAD is 0.370 – 0.411.

	Long Texts	Short Texts
Ours	**0.418**	0.333
Ours – without similarity	0.396	0.271
Ours – without word info	0.319	0.185
Ours – without root change	0.388	**0.348**
MEAD	0.391	0.343
LexRank	0.378	0.326
First n words	**0.460**	**0.368**

Table 4: ROUGE-L F-measures for Experiment 1.

	Precision	Recall	F-measure
Our summariser	0.361	0.332	0.344
MEAD	0.366	0.355	0.358
First n words	**0.403**	**0.395**	**0.399**

Table 5: ROUGE-L scores for Experiment 2.

We test whether concept matching is beneficial by switching off similarity derived from distributional semantics, or switching off all "word information" which includes distributional semantics, lemmatisation, and coreference detection, i.e. to consider matching only for the same word. Performance deteriorates when concept matching is switched off, substantially if all word information is off. This confirms our hypothesis that one of the cornerstones of KvD, concept matching, can be at least partially simulated using today's distributional semantics methods. As for root change, turning it off seems to hurt performance on the longer texts, but not so on the shorter ones, which matches our speculation that root change is useful for longer texts, which have some focus shifts.

The result of Experiment 2 is shown in Table 5. This experiment on a large dataset demonstrates that our summariser performs in the ballpark of typical results of extractive summarisers, although it is still statistically a little worse than the state-of-the-art MEAD (whose F-measure 95%-confidence interval is 0.349 – 0.367). Our summariser is good at precision because many summaries produced have not used up the 100-word limit, making the average summary length smaller than that of MEAD's. This indicates that our summariser might be good at very short summaries, or we could improve the memory selection to allow for a more diversified important proposition set. Considering this, and the fact that we have many parameters not tuned for the task, and we have not utilised the structural / positional features (whose importance is shown in the first-n baseline), the result is still encouraging.

5 Conclusions

We present here a first prototype of the feasibility of basing a summarisation algorithm on Kintsch and van Dijk's (1978) model. Our implementation successfully creates flexible-length summaries, highly compressed if desired, and provides some explanation for why certain meaning units appear in the summary. We have avoided some of the hardest aspects of KvD's model, which have to do with the generation of macropropositions and with keeping closer track of larger discourse structures, but we show that some core aspects of the model can be approximated with today's parsing and lexical semantics technology. Although the output summaries are not yet in all cases grammatical, we show that our system performs comparably with extractive state-of-the-art summarisers.

During the implementation, we had to solve several practical problems that the KvD did not give enough procedural detail about, or skipped over in their manual simulation. For instance, we have turned the distinction between LTM and STM to two parallel salience levels from KvD's two disjoint stages, formalised the tree building process and improved KvD's root choice strategy.

The KvD model does not keep track of unique events, but would profit from doing so, for instance in texts where more than one event of the same type is referred to. It has no explicit model of time, but would profit from one. It does not even use information about which entities in a text form the same concept or individual, for selecting all information about that concept into the summary. There are also many interesting ways how the memory cycle could be modified by giving more weight to particular events, concepts and individuals.

On the implementational side, much remains to be tried. Anything that improves the proposition builder should bear direct fruit in the quality of the summaries. The limitations of our proposition builder come from the limitations of parsing technology as well as the fact that semantics is not entirely determined by syntax. For instance, we noticed some problems caused by incorrect prepositional phrase attachment. A better coreference system would also improve this summariser immensely, reducing much uncertainty in the concept matching. The deep nature of the summariser also enables natural language generation to improve the readability of our textual summary.

Acknowledgement

Joint scholarship from the Cambridge Commonwealth, European & International Trust and the China Scholarship Council is gratefully acknowledged.

References

A Baddeley. 2007. *Working memory, thought, and action*. Oxford University Press.

Regina Barzilay and Michael Elhadad. 1997. Using lexical chains for text summarization. In Inderjeet Mani and Mark T. Maybury, editors, *Proceedings of the ACL/EACL-97 Workshop on Intelligent Scalable Text Summarization*.

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

BK Britton and S Gulgoz. 1991. Using kintsch's computational model to improve instructional text: Effects of repairing inference calls on recall and cognitive structures. *Journal of Educational Psychology*.

Ann L. Brown, Jeanne D. Day, and Jones R. S. 1983. The development of plans for summarizing text. *Child development*. was in press in 1983.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21th (SIGIR-98)*, pages 335–336, Melbourne, Australia.

Robin Cohen. 1984. A computational theory of the function of clue words in argument understanding. In *Proceedings of the 10th (COLING-84)*, pages 251–255.

J Coiro and E Dobler. 2007. Exploring the online reading comprehension strategies used by sixthgrade skilled readers to search for and locate information on the internet. *Reading research quarterly*.

KA DeLong, TP Urbach, and M Kutas. 2005. Probabilistic word pre-activation during language comprehension inferred from electrical brain activity. *Nature neuroscience*.

Bonnie Dorr and David Zajic. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *in Proceedings of Workshop on Automatic Summarization*, pages 1–8.

MP Driscoll. 2005. *Psychology of learning for instruction*. Allyn and Bacon.

NK Duke and PD Pearson. 2002. Effective practices for developing reading comprehension. In Alan E. Farstrup and S. Jay Samuels, editors, *What research has to say about reading instruction*.

Brigitte Endres-Niggemeyer. 1998. *Summarizing Information*. Springer-Verlag, New York, NY.

Arthur C. Graesser, Danielle S. McNamara, Max M. Louwerse, and Zhiqiang Cai. 2004. Coh-metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, & Computers*, 36(2):193–202.

V Anderson Hidi. 1986. Producing written summaries: Task demands, cognitive operations, and implications for instruction. *Review of educational research*.

A King. 1992. Comparison of self-questioning, summarizing, and notetaking-review as strategies for learning from lectures. *American Educational Research Journal*.

Walter Kintsch and Teun A. van Dijk. 1978. Toward a model of text comprehension and production. *Psychological review*, 85(5):363–394.

Walter Kintsch and Douglas Vipond. 1979. Reading comprehension and readability in educational practice and psychological theory. In Lars-Göran Nilsson, editor, *Perspectives on Memory Research: Essays in Honor of Uppsala's 500th Anniversary*, pages 329–365. Erlbaum Associates.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1).

Keiko Koda. 2005. *Insights into second language reading: A cross-linguistic approach*. Cambridge Univeristy Press.

Wendy G Lehnert. 1981. Plot units and narrative summarization. *Cognitive Science*, 5(4):293–331.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

William C. Mann and Sandra A. Thompson. 1987. Rhetorical Structure Theory: Description and construction of text structures. In Gerard Kempen, editor, *Natural Language Generation: New Results in Artificial Intelligence, Psychology, and Linguistics*, pages 85–95. Marinus Nijhoff Publishers, Dordrecht, NL.

Daniel Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press.

R Mihalcea and P Tarau. 2004. Textrank: Bringing order into texts. In *Proceedings of the EMLNP*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Paul Over and W Liggett. 2002. Introduction to duc: An intrinsic evaluation of generic news text summarization systems. In *Proc. DUC*. http://www-nlpir.nist.gov/projects/duc/guidelines/2002.html.

A Paivio. 1990. *Mental representations*. Oxford Science Publications.

Aannemarie Sullivan Palinscar and Ann L. Brown. 1984. Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. *Cognition and Instruction*, 1:117–175.

Michael Pressley. 2006. *Reading instruction that works: The case for balanced teaching*. Guildford Press.

Dragomir R. Radev and Kathleen R. McKeown. 1998. Generating natural language summaries from multiple on-line sources. 24(3):469–500.

Dragomir Radev, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Celebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, Danyu Liu, Jahna Otterbacher, Hong Qi, Horacio Saggion, Simone Teufel, Michael Topper, Adam Winkel, and Zhu Zhang. 2004. Mead – a platform for multidocument multilingual text summarization. In *Proceedings of LREC-04*.

Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*.

F Smith. 2004. *Understanding reading: A psycholinguistic analysis of reading and learning to read*. Lawrence Erlbaum.

Karen Spärck Jones. 1993. What might be in a summary? Technical report, Computer Laboratory, University of Cambridge.

Teun A. van Dijk. 1977. *Text and Context: Explorations in the Semantics and Pragmatics of Discourse*. Longman, London, UK.

RA Zwaan. 2003. The immersed experiencer: Toward an embodied theory of language comprehension. *Psychology of learning and motivation*.