Chapter 6

Corpus based Multi-document Telugu Text Summarization in Indian languages

6.1 Introduction

Extraction of a single summary from multiple documents has gained interest since mid 1990s, most applications being in the domain of news articles. Several web based news clustering systems were inspired by research on multi-document summarization, for example Google News, Columbia News Blaster, or News In Essence This departs from single-document summarization since the problem involves multiple sources of information that overlap and supplement each other, being contradictory at occasions. So the key tasks are not only identifying and coping with redundancy across documents, but also recognizing novelty and ensuring that the final summary is both coherent and complete [89].

The field seems to have been pioneered by the NLP group at Columbia University [64], where a summarization system called SUMMONS was developed by extending already existing technology for template-driven message understanding systems. Although in that early stage multi-document summarization was mainly seen as a task requiring substantial capabilities of both language interpretation and generation, it later gained autonomy, as people coming from different communities added new perspectives to the problem. Extractive techniques have
been applied, making use of similarity measures between pairs of sentences. Approaches vary on how these similarities are used: some identify common themes through clustering and then select one sentence to represent each cluster, others generate a composite sentence from each cluster [64], while some approaches work dynamically by including each candidate passage only if it is considered novel with respect to the previous included passages, via maximal marginal relevance [38]. Some recent work extends multi-document summarization to multilingual environments [46].

The way the problem is posed has also varied over time. While in some publications it is claimed that extractive techniques would not be effective for multi-document summarization [64], some years later that claim was overturned, as extractive systems like MEAD [70] achieved good performance in large scale summarization of news articles. This can be explained by the fact that summarization systems often distinguish among themselves about what their goal actually is. While some systems, like SUMMONS, are designed to work in strict domains, aiming to build a sort of briefing that highlights differences and updates across different news reports, putting much emphasis on how information is presented to the user, others, like MEAD, are large scale systems that intend to work in general domains, being more concerned with information content rather than form. Consequently, systems of the former kind require a strong effort on language generation to produce a grammatical and coherent summary, while latter systems are
probably more close to the information retrieval paradigm. Abstractive systems like SUMMONS are difficult to replicate, as they heavily rely on the adaptation of internal tools to perform information extraction and language generation. On the other hand, extractive systems are generally easy to implement from scratch, and this makes them appealing when sophisticated NLP tools are not available.

### 6.2 Abstraction and Information Fusion

As far as we know, SUMMONS [64] is the first historical example of a multi-document summarization system. It tackles single events about a narrow domain (news articles about terrorism) and produces a briefing merging relevant information about each event and how reports by different news agencies have evolved over time.

Rather than working with raw text, SUMMONS reads a database previously built by a template-based message understanding system. A full multi-document summarizer is built by concatenating the two systems, first processing full text as input and filling template slots, and then synthesizing a summary from the extracted information. The architecture of SUMMONS consists of two major components, a content planner that selects the information to include in the summary through combination of the input templates, and a linguistic generator that selects the right words to express the information in grammatical and coherent text. The latter component was devised by adapting existing language generation tools, namely the FUF/SURGE system. Content
planning, on the other hand, is made through summary operators, a set of heuristic rules that perform operations like “change of perspective”, “contradiction”, “refinement”, etc. Some of these operations require resolving conflicts, i.e., contradictory information among different sources or time instants, others complete pieces of information that are included in some articles and not in others, combining them into a single template. At the end, the linguistic generator gathers all the combined information and uses connective phrases to synthesize a summary [89].

While this framework seems promising when the domain is narrow enough so that the templates can be designed by hand, a generalization for broader domains would be problematic. This was improved later by McKeown [34], where the input is now a set of related documents in raw text, like those retrieved by a standard search engine in response to a query. The system starts by identifying themes, i.e., sets of similar text units (usually paragraphs). This is formulated as a clustering problem. To compute a similarity measure between text units, these are mapped to vectors of features, which include single words weighted by their TF-IDF scores, noun phrases, proper nouns, synsets from the Wordnet database and a database of semantic classes of verbs. For each pair of paragraphs, a vector is computed that represents matches on the different features. Decision rules that were learned from data are then used to classify each pair of text units either as similar or dissimilar; this in turn feeds a subsequent algorithm that places the
most related paragraphs in the same theme. Once themes are identified, the system enters its second stage, information fusion. The goal is to decide which sentences of a theme should be included in the summary. Rather than just picking a sentence that is a group representative, the authors propose an algorithm which compares and intersects predicate argument structures of the phrases within each theme to determine which are repeated often enough to be included in the summary. This is done as follows, first, sentences are parsed through Collins' statistical parser [39] and converted into dependency trees, which allows capturing the predicate-argument structure and identify functional roles. Determiners and auxiliaries are dropped.

The comparison algorithm then traverses these dependency trees recursively, adding identical nodes to the output tree. Once full phrases (a verb with at least two constituents) are found, they are marked to be included in the summary. If two phrases rooted at some node, are not identical but yet similar, the hypothesis that they are paraphrases of each other is considered, to take this into account, corpus driven paraphrasing rules are written to allow paraphrase intersection. Once the summary content (represented as predicate-argument structures) is decided, a grammatical text is generated by translating those structures into the arguments expected by the FUF/SURGE language generation system.

6.3 Topic-driven Summarization and MMR
Carbonell and Goldstein [34] made a major contribution to topic-driven summarization by introducing the maximal marginal relevance (MMR) measure. The idea is to combine query relevance with information novelty, it may be applicable in several tasks ranging from text retrieval to topic-driven summarization. MMR simultaneously rewards relevant sentences and penalizes redundant ones by considering a linear combination of two similarity measures. Let $Q$ be a query or user profile and $R$ a ranked list of documents retrieved by a search engine. Consider an incremental procedure that selects documents, one at a time, and adds them to a set $S$. So let $S$ be the set of already selected documents in a particular step, and $R \setminus S$ the set of yet unselected documents in $R$. For each candidate document $D_i \in R \setminus S$, its marginal relevance $MR(D_i)$ is computed as:

$$MR(D_i) := \lambda \text{Sim}_1(D_i, Q) - (1 - \lambda) \max_{D_j \in S} \text{Sim}_2(D_i, D_j)$$

(5)

where $\lambda$ is a parameter lying in $[0,1]$ that controls the relative importance given to relevance versus redundancy. Sim1 and Sim2 are two similarity measures; in the experiments both were set to the standard cosine similarity traditionally used in the vector space model.

### 6.4 Graph Spreading Activation

Mani and Bloedorn [60] describe an information extraction framework for summarization, a graph-based method to find similarities and dissimilarities in pairs of documents. Albeit no textual summary is
generated, the summary content is represented via entities (concepts) and relations that are displayed respectively as nodes and edges of a graph. Rather than extracting sentences, they detect salient regions of the graph via a spreading activation technique.

6.5 Centroid-based Summarization

Although clustering techniques were already being employed by McKeown and Barzilay [34] for identification of themes, Radev [70] pioneered the use of cluster centroids to play a central role in summarization. A full description of the centroid-based approach that underlies the MEAD system can be found in Radev [71]. Perhaps the most appealing feature is the fact that it does not make use of any language generation module, unlike most previous systems. All documents are modeled as bags-of-words. The system is also easily scalable and domain-independent.

6.6 Multilingual Multi-document Summarization

Evans [46] addresses the task of summarizing documents written in multiple languages, this had already been sketched by Hovy and Lin [49]. Multilingual summarization is still at an early stage, but this framework looks quite useful for newswire applications that need to combine information from foreign news agencies.
Evans [46] considered the scenario where there is a preferred language in which the summary is to be written, and multiple documents in the preferred and in foreign languages are available. In their experiments, the preferred language was English and the documents are news articles in English and Arabic. The rationale is to summarize the English articles without discarding the information contained in the Arabic documents. The IBM's statistical machine translation system is first applied to translate the Arabic documents to English. Then a search is made, for each translated text unit, to see whether there is a similar sentence or not in the English documents. If the sentence is found relevant enough to be included in the summary, the similar English sentence is included instead of the Arabic-to-English translation. This way, the final summary is more likely to be grammatical, since machine translation is known to be far from perfect. On the other hand, the result is also expected to have higher coverage than using just the English documents, since the information contained in the Arabic documents can help to decide about the relevance of each sentence. In order to measure similarity between sentences, a tool named SimFinder was employed, this is a tool for clustering text based on similarity over a variety of lexical and syntactic features using a log-linear regression model.

6.7 Our Approach

Our basic approach attempts to generate a text summary from the leading news article of Telegu news papers, while avoiding the
repetition of identical or similar information and presenting the information in such a way that makes sense to the reader. With this in mind we decided on the following algorithm:

- **Summarize each document**
  - The first step of the process of generating a multi-document summary is to create individual single-document summaries for all documents in the set.

- **Group the summaries in clusters**
  - The second step of the multi-document (MD) summarization process is the grouping of the individual summaries into clusters. Documents are ‘clustered’ on the basis of the contents of their summaries where a cluster consists of summaries that describe a similar topic.
  - For those documents that seem to discuss a similar topic, representative segments are eventually chosen for the MD summary while the other ones are ‘hidden’, i.e., not shown but still accessible to the user.

- **For each cluster select representative passage(s) that will contribute to the final summary.**
  - The third step of the multi-document summarization process involves selecting a member of a cluster as a representative summary for the cluster. When the user wants a topical summary, the topic description is used to pick the summary that has most similarity to the topic.
In the case of a generic summary, the representative summary chosen is one that best represents the cluster. In this case, the summary that has most occurrences of the common terms across documents in a cluster is chosen. Since clusters can be overlapping it is possible that the same segment(s) is chosen to represent several clusters.

- Organize these passages in a logical way.
  - Finally, the last step of the multi-document summarization process involves organizing the selected passages in an order that makes sense to the reader. Such an order might depend on the topic (in case of topical summaries), the user and the task at hand, among others. Currently, we organize the selected passages based upon topic similarity.

### 6.8 Results

The experiments are done by taking the corpus, which was developed by the articles from Eenaadu newspaper, Sakshi, Andhra Jyoti, Vaartha and Andhrabhoomi over 100 days. The corpus includes 270 articles totaling to 4.5 Million words. All of the 970 documents in the Telugu News Articles corpus, from different News Papers and are from politics section.

Table 6.1: Comparison between Human and Machine generated summaries.
<table>
<thead>
<tr>
<th>Leading Telugu Daily</th>
<th>Number of Articles</th>
<th>Number of Articles</th>
<th>Machine</th>
<th>Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eenadu</td>
<td>01</td>
<td></td>
<td>270</td>
<td>91.12%</td>
</tr>
<tr>
<td>Sakshi</td>
<td>01</td>
<td></td>
<td>270</td>
<td></td>
</tr>
<tr>
<td>Andhra Jyoti</td>
<td>01</td>
<td>93.45</td>
<td>270</td>
<td>91.12%</td>
</tr>
<tr>
<td>Andhra bhoomi</td>
<td>01</td>
<td></td>
<td>270</td>
<td></td>
</tr>
<tr>
<td>Vaartha</td>
<td>01</td>
<td></td>
<td>270</td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
<td>93.45%</td>
<td></td>
<td>91.12%</td>
</tr>
</tbody>
</table>