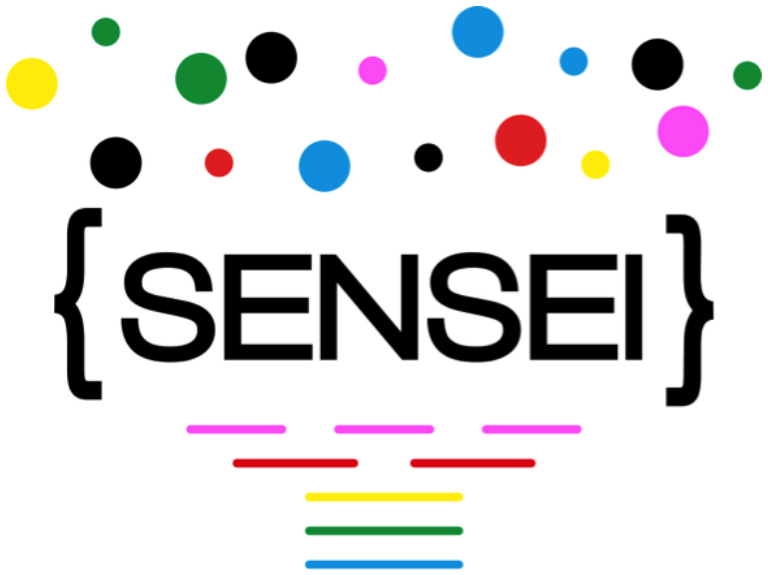

D5.3 – Final conversation analysis and presentation components

Document Number	D5.3
Document Title	Final conversation analysis and presentation components
Version	1.0
Status	Final
Workpackage	WP5
Deliverable Type	Report
Contractual Date of Delivery	30.04.2016
Actual Date of Delivery	30.04.2016
Responsible Unit	USFD
Keyword List	Speech and social media summarization approaches
Dissemination level	PU

Editor
Ahmet Aker (University of Sheffield, USFD)

Contributors

Fabio Celli	(University of Trento, UNITN)
Evgeny Stepanov	(University of Trento, UNITN)
Elisa Chiarani	(University of Trento, UNITN)
Ahmet Aker	(University of Sheffield, USFD)
Adam Funk	(University of Sheffield, USFD)
Robert Gaizauskas	(University of Sheffield, USFD)
A R Balamurali	(Aix Marseille Universit, AMU)
Benoit Favre	(Aix Marseille Universit, AMU)
Mijail Kabadjov	(University of Essex, UESSEX)
Udo Kruschwitz	(University of Essex, UESSEX)

SENSEI Coordinator

Prof. Giuseppe Riccardi

Department of Information Engineering and Computer Science

University of Trento, Italy

giuseppe.riccardi@unitn.it

Document change record

Version	Date	Status	Author (Unit)	Description
0.1	2016-03-02	Draft	Ahmet Aker (USFD)	Initial Outline
0.1	2016-03-10	Draft	Ahmet Aker (USFD)	Wrote section on Comment Clustering
0.2	2016-03-11	Draft	Ahmet Aker (USFD)	Wrote section on Cluster Labeling
0.3	2016-03-15	Draft	Ahmet Aker (USFD)	Wrote section on Abstractive Summarization
0.3	2016-03-15	Draft	Fabio Celli (UNITN)	Wrote section on Agreement/Disagreement
0.3	2016-03-16	Draft	Fabio Celli (UNITN)	Updated section on Agreement/Disagreement
0.4	2016-03-16	Draft	Evgeny Stepanov (UNITN)	Wrote section on Discourse Relations
0.5	2016-03-16	Draft	Adam Funk (USFD)	Added section on sentiment detection
0.6	2016-03-16	Draft	Balamurali A R (AMU)	Updated section on Cluster Labeling
0.6	2016-03-18	Draft	Benoit Favre (AMU)	Writing AMU part of sentiment detection
0.7	2016-03-18	Draft	Mijail Kabadjov (UESSEX)	Added section on Coreference Resolution
0.7	2016-03-21	Draft	Mijail Kabadjov (UESSEX)	Updated section on Coreference, results, discussion
0.7	2016-03-24	Draft	Mijail Kabadjov (UESSEX)	Added illustrative example to section on Coreference
0.8	2016-03-30	Draft	Ahmet Aker (USFD)	Updated several sections
0.8	2016-03-31	Draft	Ahmet Aker (USFD)	Wrote section on extractive summarization
0.8	2016-03-31	Draft	Ahmet Aker (USFD)	Updated section on abstractive summarization
0.9	2016-04-05	Draft	Udo Kruschwitz (UESSEX)	Review of the document
0.9	2016-04-05	Draft	Ahmet Aker (USFD)	Updated several sections to address reviewers comments

Version	Date	Status	Author (Unit)	Description
0.10	2016-04-07	Draft	Benoit Favre (AMU)	Added section on Speech summarization
0.10	2016-04-08	Draft	Mijail Kabadjov (UESSEX)	Glue with intro, cross reference USFD Data and Coreference, overall spell-checking
0.10	2016-04-13	Draft	Evgeny A. Stepanov (UNITN)	Added Section 2.2 on Abstractive Summarization
0.11	2016-04-13	Draft	Elisa Chiarani (UNITN)	Quality check finished
0.12	2016-04-14	Draft	Ahmet Aker (USFD)	Merging contributions, writing Introduction
0.12	2016-04-14	Draft	Benoit Favre (AMU)	Writing introduction for the speech section
0.12	2016-04-14	Draft	Evgeny A. Stepanov (UNITN)	Update on Section Introduction
0.12	2016-04-15	Draft	Ahmet Aker (USFD)	Writing Conclusion
0.13	2016-04-15	Draft	Ahmet Aker (USFD)	Adressing Quality checks
0.14	2016-04-18	Draft	Udo Kruschwitz (UESSEX)	Second review of the document
0.15	2016-04-19	Draft	Ahmet Aker (USFD)	Addressing second review comments
0.16	2016-04-25	Draft	Evgeny A. Stepanov (UNITN)	Addressing second review comments
0.16	2016-04-25	Draft	Fabio Celli (UNITN)	Addressing second review comments
0.17	2016-04-27	Draft	Benoit Favre (AMU)	Addressing second review comments
0.18	2016-04-27	Draft	Rob Gaizauskas (USFD)	Proof reading/minor editorial changes throughout. Revised Introduction, Introduction to Section 3 and Conclusions
0.19	2016-04-28	Draft	Elisa Chiarani (UNITN)	Final quality check finished
1.0	2016-04-28	Final Draft	Ahmet (USFD)	Adressing final quality checks

Executive Summary

In this report we describe the refinement and extensions to the design and evaluation of software conversation analysis and summarisation components developed to generate summaries for both speech and social media use cases. For the speech use case we describe two abstractive template-based summarization systems. For the social media use case we report various components that analyse comments for sentiment, discourse relations, agreement and disagreement and anaphora resolution between different mentions of discourse entities. Furthermore, we describe components that perform clustering of comments into topics and generate labels for those topic clusters. Finally, we present three different summarization systems to summarize comments to online news.

Contents

List of Acronyms and Abbreviations

Acronym	Meaning
THM	Town Hall Model
MCL	Markov Clustering
LDA	Latent Dirichlet Allocation
ACOF	Agent Conversation Observation Form
DECODA	Call center human-human spoken conversation corpus
LUNA	Spoken Language UNderstanding in MultilinguAl Communication Systems
QA	Quality Assurrance
CRF	Conditional Random Fields
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
MMR	Maximal Marginal Relevance
DISCO	DIStributionally similar words using CO-occurrences
SCTM	Specific Correspondence Topic Model

1 Introduction

The overall objective of WP5 is:

> to develop the tools to (1) perform various types of analysis across the representations of multiple conversations produced by the WP3 and WP4 tools, and (2) generate appropriate multimodal output for presentation of the results of this analysis to end users, as specified in WP1. (SENSEI Annex I – Description of Work)

In D5.1 (M12) we reported on the design and implementation of a "conversation data repository" (CDR) whose purpose is to hold the semantic, sentiment and discourse analyses of conversational data that are output by WP3 and WP4 modules and make them available to WP5 analytics and summarization modules.

In D5.2 (M24) we presented clear specifications of the summary outputs we aimed to produce for both the speech and social media use cases, the use cases having being initially specified in D1.1 (M6) and then refined in D1.2 (M12). D5.2 also reported initial versions of modules for conversation analytics and summarization for both the speech and social media use cases, designed to generate the target summary outputs.

In this report we describe further refinements and extensions to the analytic and summarization components developed since D5.2 for both speech and social media use cases. These developments have resulted in:

- improved performance;
- increased functionality;
- fuller exploitation of WP3 and WP4 outputs in the analyltics and summarization components;
- well-engineered integration of WP3 and WP4 at the data interchange level by communication via the conversation data repository.

The rest of this report is structured as follows. The next subsection (1.1) overviews in a bit more details the changes in the analytics and summarization components since the end of Period 2. Then, sections 2 and 3 describe the final conversational analysis and summarization modules for the speech and social media use cases, respectively. These sections include both a description of the algorithms used and, wherever possible, report results of intrinsic evaluation, as well as efforts undertaken to create resources for evaluation. We conclude the report in Section 4

1.1 Follow-up to Period 2 Activities

Speech In the speech use case we have further developed the two abstractive template-based summarization systems described in D5.2. Both systems are based on the generalization of the human-authored synopses to templates with the slots to fill, and linking the summary units to the original

conversations. The difference between the approaches is that the first approach links “concepts” in summaries and conversation, while the second links summary sentences to a set of conversation utterances (a community) and extracts candidates for slot filling. Both approaches are evaluated both on manual and automatic conversation transcripts.

Social Media In the social media use case we have integrated components from WP3 and WP4 for sentiment analysis, detection of agreement/disagreement, discourse parsing and anaphora resolution into the analytics and summarization pipeline. Apart from the sentiment analysis component, the other components feed directly into the clustering component that is used to cluster news comments by topic. Each cluster is also labeled by a topic descriptor. For this we developed two different approaches: an extractive and abstractive approach. We also finalzed three different summarization systems which differ in the way they determine and present the most important pieces of information extracted from the comments to the user. One of these systems is new since D5.2.

2 Conversation Analysis/Summarization Outputs for Speech

Call-centres typically process tens of thousands of calls per day. Current industry-standard technology allows these calls to be recorded, annotates them with metadata and automatically processes them in order to perform speech analytics. Such technology enables keyword search of machine-generated transcripts and assesses basic speaker state, such as emotion, stress, etc. It is the basis for quality assurance activity for rating agents and assessing customer needs, in order to improve the service. While this automation has helped QA supervisors and managers get a broad view of operations, fine-grained analysis on small conversation samples ($< 1\%$) is still at the center of the QA process. In order to analyse and process the entirety of the calls, new technology is necessary for extracting and exposing relevant information. In this section, we describe two summarization approaches developed by the SENSEI project in order to better represent the content of call-centre conversations. Figure 1 gives a example of conversation from the RATP-DECODA call-centre corpus which we aim to summarize.

Automatic summarization is generally based on extractive methods that gather relevant text segments to make a summary. These methods are not well suited for spoken conversation summary generation, due to their spontaneous and interactive nature. By selecting only a few utterances from a conversation, extractive summaries just give a broad picture of a given point in the conversation and not a full synthetic description of what happened between the different participants.

For instance, when summarizing call-centre conversations, it would be good if the summaries could inquire about the issues described by the callers and the solutions outlined by the agents. Often, issues are described in multiple speaker turns, from the caller in interaction with the agent, which is difficult to capture with an extractive approach, especially when length constraints are tight.

Template filling is a summarization approach which has shown success when the domain of the conversations is restricted [1]. It consists of filling hand-crafted templates with information extracted from the conversations transcripts. In the case of call-centre conversations, this method can be tackled for generating short narrative summaries which recount of what happened during the call. However, in addition to hand-writing templates, and annotating transcripts with template slots, this approach is limited in that it cannot handle situations that have not been imagined by the template creators. In the following, we describe two approaches for sparing the template writing labor: by recombining and generalizing existing summaries, and by automatically generating novel templates.

2.1 Templates generation by recombining existing synopses

Instead of requiring experts to engineer templates from a collection of conversations, we generate such templates from a corpus of existing synopses annotated with slot variables. Essentially, the approach consists of composing novel templates from sentences selected in the training synopses, and filling them with the method developed previously.

The approach is based on template filling. Each slot in a template is filled depending on the analysis

Agent: ⟨name⟩ hello
Caller: yes hello
Agent: hello madam
Caller: are buses uh 172 and 186 running?
Agent: unfortunately on the 172 and uh 186, we got the information this morning, there's a notice from the B depot in Vitry so it was known uh yesterday evening and this morning
Caller: uh yes
Agent: so the buses are very disrupted uh this morning huh some uh, uh have gone out and others not, so there are very major disruptions on these two bus lines huh
Caller: whew that's really irritating because what will people who are working do
Agent: unfortunately yeah, it's annoying huh I understand that uh actually
Caller: further there was a notice that was uh
Agent: frankly not uh
Caller which in fact creates in the private... who risk their post, if they're not going to work because those gentlemen have decided to strike
Agent: it's me I somewhat agree with you
Caller: someone from the RATP who agrees with me...

Figure 1: Extract from Decoda conversation (20091112-RATP-SCD-0042) translated from French. Sample reference synopses for that conversation: *Are buses 172 and 186 running? No, disrupt because of Vitry depot strike, complaint and compassion* (annotator 1), *Query of information on the status of buses 172 and 186. Major disruption on these lines due to a strike. Complaint from the caller.* (annotator 2).

of conversation transcripts in order to generate a *synopsis*, which is a short summary of the whole conversation. We propose in this study to generate templates dynamically thanks to a training corpus made of pairs of *conversation transcripts* and *synopses*. Our training process consists of the following steps:

1. Concept slot detection: conversation transcriptions and synopses of the training corpus are parsed in order to detect slots corresponding to the *concepts* relevant to characterize conversations. The list of concepts used is related to the application domain of the corpus.
2. Sentence template generation: all sentences in the synopsis corpus are generalized by replacing concept values by labels in order to produce *sentence templates*. Examples of such templates can be found in Table 1.
3. Concept linking: this task consists of linking concepts occurring in a summary to the same concepts in the corresponding conversation. A classifier is trained in order to predict, for all concepts detected in a given conversation, if they would occur in its corresponding summary.

Once the sentence templates and concept-linking classifier have been obtained, the summarization process of a new conversation transcription is as follows:

1. Relevant concept detection: the concept-linking classifier is used in order to detect the *relevant* concepts in the conversation transcription. A concept is considered *relevant* if it occurs in the synopsis of the conversation.
2. Sentence template selection: this step consists of dynamically choosing sentence templates from the template repository according to the slots detected in the previous step.

Table 1: Templates for selecting Decoda topics (translated from French).

Code	Topic	Template
HORR	Schedule	Query for schedules (using $TRANSPORT)? from $FROM to $TO.
ITNR	Itinerary	Query for itinerary (using $TRANSPORT)? from $FROM to $TO (without using $NOT_TRANSPORT)?. (Take the $LINE towards $TOWARDS from $START_STOP to $END_STOP.)*. Query for location $LOCATION.
NVGO	Navigo pass	Query for (justification|refund|fares|receipt) for $CARD_TYPE. Customer has to go to offices at $ADDRESS.
OBJT	Lost&found	$ITEM lost in $TRANSPORT (at $LOCATION)? (around $TIME)?. (Found, to be retrieved from $RETRIEVE_LOCATION |Not found).
TARF	Fares	Query for fares from $FROM to $TO. The fare is $COST.
ETFC	Traffic	Query for state of line $TRANSPORT. (Frequency is $FREQUENCY |Not running because of $ISSUE |Cannot get information because of $ISSUE)
-	Other	Call corresponding service at $PHONE_NUMBER. Send a mail query to $ADDRESS.

3. Synopsis generation: all the sentence templates selected are filled with the concept values found in the conversation transcription; then this set of sentences is ordered in order to produce the final synopsis.

These processes are described in more detail in the next sections.

Linking: propagation to conversation transcripts Given a synopsis manually annotated with concept slots, the task consists of propagating the annotation to conversation transcripts. This linking task is performed through the following steps:

- Transcripts are automatically annotated with syntactic and semantic parses with the Macaon tool chain [2].
- Each slot in the annotated synopses is compared with all the phrases from the transcription thanks to Levenshtein alignment and a specific cost function based on character-level matching. Text is first lower-cased and diacritics are removed, the distance is computed at the character level.
- The slot value is associated with the phrase for which the alignment has lowest cost.

This method aligns 316 slots in the 380 annotated synopses (83.16% alignment rate). The unaligned variables are in most cases due to manual annotation errors, overly generic references that cannot be detected at the word level, or a total mismatch between the synopsis and the conversation (i.e. the word does not appear in the conversation, which is the case when the author of the synopsis generalized a concept using a different word.)

Slot prediction features The previous step leads to the creation of a corpus associating slots from the synopses and values from the transcripts. This data can be leveraged to train a slot classifier. Again

we take advantage of the parses generated by Macaon [2] for feature extraction. For each phrase in a conversation, the classifier is trained to predict a type of slot among 19 available plus the NULL label indicating that the phrase is not a concept. The classifier uses the following features as input:

- **Syntactic head of the phrase:** word, lemma, part-of-speech tag, named entity tag.
- **Governor of syntactic head:** lemma, part-of-speech tag, dependency label.
- **Phrase:** length, bag of n-grams of words ($n \leq 3$), bag of n-grams of part of speech ($n \leq 3$).
- **Conversation and discourse:** number of named entities of the same type since the beginning of the conversation, number of occurrences of the head lemma since the beginning of the conversation, topic of the conversation, relative position of the phrase in the conversation, speaker role (agent or caller).

Given these features, the scores output by the classifier are passed through a *softmax* function to represent probabilities between 0 and 1. For a conversation, at test time, scores for the NULL class are discarded and for each slot type, all phrases which exceed a decision threshold θ are selected for use in synopsis generation.

Using conversation and discourse features is not conventional in the concept or named entity recognition tasks. They help address the relevance of the detected concepts. For instance, a number of bus stops might be referred to in the conversation while only one is relevant for filling the template.

Synopsis generation Synopsis generation is performed by combining fragments of synopses gathered in the training data, and replacing their concept slot values with those detected in the transcript. In a way, this approach can be considered as extractive except that existing synopses are leveraged instead of conversation utterances.

First, synopses from the training set are split into sentences and slot values are replaced by tokens indicating their type. Those sub-templates can be selected and filled depending on the content of a conversation. Then, slots are detected in the transcript according to the approach described in the previous sections. The selection process tries to saturate the sub-templates with detected slot values which match the expected slot types, under the constraint that a slot type can only be used once. From this population of saturated sub-templates, the generated synopsis is necessarily started with a sentence which was first in its original synopsis. Then, other sub-templates are concatenated arbitrarily. We decided to rely on this heuristic because in our data the first sentence of a synopsis always contains the right description of the issue of the call.

The advantage of this approach is that sub-template selection is driven by the detected slots. This both limits the risk to accidentally instantiate sub-templates based on misdetected information, and it also allows for the approach to cope with a limited quantity of novelty in the conversations: situations that are combination of already seen situations.

2.1.1 Evaluation and Results

Experiments are performed on 141 conversations from the RATP DECODA corpus manually annotated in synopsis. Each conversation has between one and three unique synopses for a total of 381 synopses manually annotated with slot segments and type. In the following experiments we make both use of manual transcriptions with the reference linguistic annotations (syntactic and semantic) and automatic transcriptions generated with the LIUM ASR system [3] (with a WER of 35%), along with automatic linguistic annotations generated by the Macaon pipeline [2].

The corpus is split in 71 conversations for training, 43 for testing and 27 for development. All parameters of the system, including the θ threshold are set in order to maximize performance on the development set.

We compare our approach (synopsis recombination) with manual template filling and a few extractive baselines and toplines. One manual template was written for each conversation topic in the corpus in order to cover most of the information in the synopses for that topic. The topline consists of manually filling the hand-crafted template with the most relevant slot values.

For slot value predictions, we use three classifiers: adaboost [4] with 1000 rounds of boosting, a deep neural network (called DNN thereafter) implemented with Chainer[1], and the libLinear classifier [5].

We follow the experimental setup of the CCCS shared task [6] except that we have a larger test set. The length limit for synopses is 7% of the conversation words, evaluation is performed with the ROUGE-2 metric, both on lexical units (the regular ROUGE) and word embeddings trained on the DECODA corpus [7]. The following systems are compared:

- **Topline**: manual templates filled with reference slots
- **Human**: the average of the performance obtained by putting aside each reference synopsis and scoring it against the other references.
- **Templates**: manual templates filled with predicted slots.
- **Recombined**: the proposed approach.
- **MMR**: maximal marginal relevance.
- **Longest**: longest speaker turn in the conversation.
- **Longest@25**: longest speaker turn in the first quarter of the conversation.

The results detailed in Table 2 show that our method (Recombined) yields better results than both the extractive baselines and the manual templates (significant at $p < .05$). This is expected because by combining sentences from multiple synopses, the system can cover situations that could not be handled by a single template per topic. This seems to also be linked to the quality of slot prediction as the topline which relies on reference slots has a much better ROUGE score. The human synopses score is worse

[1] http://chainer.org – Parameters: 1 hidden layer, ReLU activations, 4 epochs of training, no dropout. Searched for from a range of configuration to maximize classification accuracy on the dev set.

System	Transcript	Slots	ROUGE-2	ROUGE-2-WE
Topline	manual	manual	0.20491	0.11131
Human	-	-	0.11848	0.06229
Templates	manual	lcsiboost	0.06818	0.04264
	manual	libLinear	0.03735	0.02579
	manual	DNN	0.02041	0.01595
Recombined	manual	lcsiboost	0.08200	0.04715
	manual	libLinear	**0.08390**	**0.05014**
	manual	DNN	0.04830	0.03268
MMR	manual	-	0.03145	0.01751
Longest	manual	-	0.02688	0.01825
Longest@25	manual	-	0.04046	0.02564
Templates	ASR	lcsiboost	0.05270	0.03416
	ASR	libLinear	0.02921	0.02053
	ASR	DNN	0.01775	0.01359
Recombined	ASR	lcsiboost	**0.08471**	**0.04845**
	ASR	libLinear	0.08100	0.04828
	ASR	DNN	0.04033	0.02763
MMR	ASR	-	0.02093	0.01464
Longest	ASR	-	0.01734	0.01190
Longest@25	ASR	-	0.01734	0.00950

Table 2: ROUGE results on the test set for all the systems, according to the transcript source, as well as how the slots were predicted. The proposed approach is called "Recombined." Regular ROUGE-2 results are compared with word-embedding-based ROUGE-2 results. Higher ROUGE scores are better, however the absolute value is not relevant since it dependends on the content of the reference summaries. Results should be interpreted relative to the Topline (manual selection and filling of templates), human performance computed by comparing references against eachother, and the baselines (Longest, etc) which are often reported in other work.

than the topline because humans tend to diverge when writing summaries, and they were evaluated with one-less reference than the systems.

Also, it seems that the choice of the classifier does not matter except for the DNN which is not as good as the other classifiers, probably because it is trained on so little data (note that its configuration has been optimized on the development set). Finally, an interesting outcome is that ASR output and automatic prediction of linguistic annotation does not have a large impact on performance. This comes from the fact that ASR transcripts are of relatively good quality, and that relevant slot values are generally repeated multiple times by both speakers in a conversation. The choice of the decision threshold on the development set seems appropriate, as evidenced by Figure 2.

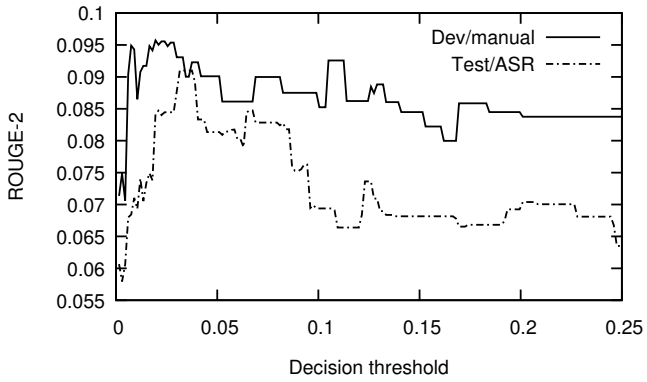

Figure 2: Variation of performance of the proposed system according to the decision threshold on the Icsiboost classifier scores. The choice of θ on the development set is relatively robust on the test set in the ASR condition.

2.2 Template generation for abstractive speech summarization

This section describes the extension of the abstractive template-based summarization approach based on automatically generated templates that was described in D5.2. First, we present the template and summary generation system, then evaluate the system on manual and automatic conversation transcripts.

On top of what was described in D5.2, here we provide finer details and a fuller description of the system and the evaluation on ASR output.

2.2.1 System Description

The conversation summarization pipeline can be partitioned into community creation, template generation, and summary generation components. The whole pipeline is depicted in Figure 3.

Template Generation Template Generation follows the approach of [8] and, starting from human-authored summaries, produces abstract templates applying slot labeling, template clustering and template fusion steps. The information required for the template generation is: part-of-speech tags, noun and verb phrase chunks, and root verbs from dependency parsing.

In the slot labeling step, noun phrases from human-authored summaries are replaced by the WordNet [9] SynSet IDs of the head nouns (rightmost). No real word-sense disambiguation is used, instead the SynSet ID of the most frequent sense is selected with respect to the part-of-speech tag. To get hypernyms for Italian we use MultiWordNet [10].

Clustering of the abstract templates generated in the previous step is performed using the WordNet [9] hierarchy of the root verb of a sentence. The similarity between verbs of sentences is computed as a cosine similarity between the shortest path that connects the senses in the hypernym taxonomy

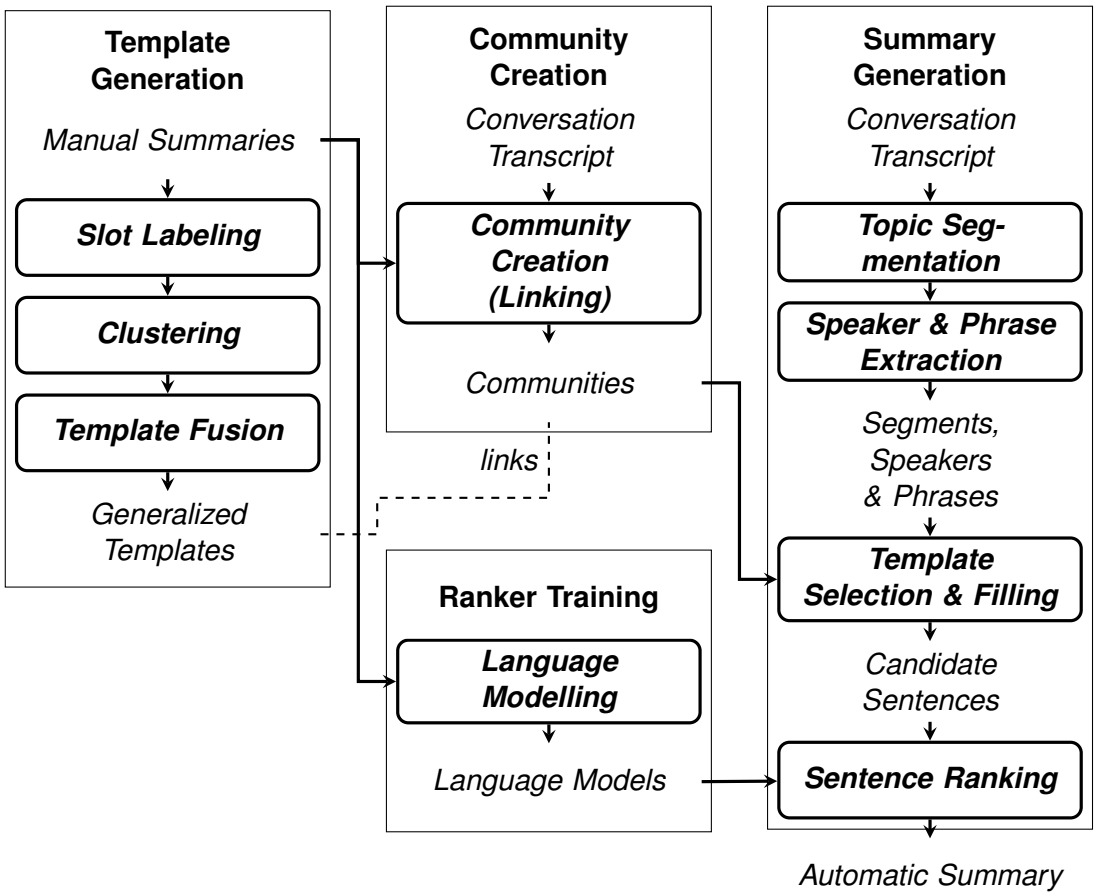

Figure 3: Abstractive Summarization Pipeline, which is partitioned into Template Generation, Community Creation, Ranking Training and Summary Generation phases.

of WordNet. The template graphs, created with respect to this similarity, are then clustered using the Normalized Cuts method [11].

The clustered templates are further generalized using a word graph algorithm extended to templates in [8]. The paths in the word graph are ranked using language models trained on the abstract templates and the top 10 are selected as a template for the cluster.

Community Creation In a few corpora (e.g. the AMI Meeting Corpus), sentences in human-authored summaries are manually linked to a set of the sentences/utterances in the meeting transcripts, referred to as communities. It is hypothesized that a community sentence covers a single topic and conveys vital information about the conversation. For automatic community creation we explore four heuristics.

- *H1* (baseline): take the whole conversation as a community for each sentence;
- *H2*: The 4 closest turns with respect to cosine similarity between a summary and a conversation sentence.
- *H3*: The 4 closest turns with respect to cosine similarity after replacing the verbs with WordNet SynSet ID.
- *H4*: The 4 closest turns with respect to cosine similarity of the averaged word2vec [12] vectors.

The Italian vectors are trained on the Europarl corpus [13].

Summary Generation The first step in summary generation is the segmentation of conversations into topics using a lexical cohesion-based domain-independent discourse segmenter – LCSeg [14]. The purpose of the step is to cover all the conversation topics. Next, all possible slot 'fillers' are extracted from the topic segments and ranked with respect to their frequency in the conversation.

An abstract template for a segment is selected with respect to the average cosine similarity of the segment and the community linked to that template. The selected template slots are filled with the 'fillers' extracted earlier. Final automatic sentences are selected from this filled template using the token and part-of-speech tag ngram language models to rank them (also other parameters could be used as in [8]).

Ranker Training and Sentence Ranking Since the system produces many sentences that repeat the same information, a set of sentences needs to be selected. The sentence ranking is based on the ngram language models trained on the tokens and part-of-speech tags from the human-authored summaries.

2.2.2 System Evaluation

In this section we evaluate the template-based abstractive summarization methodology with automatic community creation heuristics, which we described in the previous section, on the Italian LUNA Human-

Table 3: ROUGE-2 recall with 7% summary length limit for the extractive baselines and the abstractive summarization systems on the Italian LUNA Corpus using automatic community creation heuristics with Cosine Similarity at WordNet SynSet ID level and averaged word2vec vectors on manual and automatic conversation transcripts. Results are for the systems trained on 100 manual transcripts and evaluated on 100 and 50 manual and 50 automatic transcripts.

Method	**Manual (100)**	**Manual (50)**	**ASR (50)**
Baseline-L	0.015	0.016	0.017
Baseline-LB	0.027	0.024	0.022
MMR	0.020	0.022	0.021
(H1) Whole Conversation	0.018	0.013	–
(H2) Top 4 turns: token	0.021	0.020	–
(H3) Top 4 turns: SynSetID	0.025	0.025	0.027
(H4) Top 4 turns: word2vec	**0.029**	**0.028**	**0.030**

Human corpus [15]. We compare the system performances to the extractive baselines defined in [16] and reported earlier in D5.2:

- the longest turn in the conversation up to the length limit (7% of a conversation) (*Baseline-L*) [17];
- the longest turn in the first 25% of the conversation up to the length limit (*Baseline-LB*) [17];
- Maximal Marginal Relevance (*MMR*) [18] with $\lambda = 0.7$.

The performances of the system with automatic community creation heuristics are given in Table 3. Among the community creation heuristics word2vec based cosine similarity metric for the conversation sentence selection performs the best. The next best results is of the heuristic that abstracts sentence verbs to their WordNet SynSet IDs prior to the similarity computation.

For the LUNA Corpus (Italian) on manual transcripts, the extractive baseline that selects the longest utterance from the first quarter of a conversation, proved to be the strong baseline with ROUGE-2 recall of 0.027 [16]. This is not surprising, since the longest turn from the beginning of the conversation is usually a problem description, which appears in human-authored summaries. The word2vec based automatic community creation heuristic, however, achieves recall of 0.029 successfully outperforming it.[2]

In a realistic setting, the summaries are to be generated on automatic transcripts, i.e. the output of Automatic Speech Recognition. Thus, it is important to assess the effect of the automatic transcripts on the performance. The ASR system for the LUNA corpus is trained by using the Kaldi [20] speech recognition toolkit. The ASR uses mel-frequency cepstral coefficients (MFCC) that are transformed by linear discriminant analysis (LDA) and maximum likelihood linear transform (MLLT). These features are then spliced in the window of $[-3, +3]$. The acoustic models are further trained by "speaker adaptive training". The speaker adaptation during decoding is performed by feature-space maximum likelihood linear regression (fMLLR) [21]. The LM for the ASR is a modified Kneser-Ney trigram model that is

[2]Statistical significance testing with the bootstrap method [19] and paired t-test yields significant differences ($p < 0.05$) for the 1 point variation in the 3rd digit of ROUGE-2.

built over the training data. The word error rate for the ASR is 35.4%. Due to the overlap between the CCCS Shared Task test set and the training set of the ASR, the evaluation on automatic transcription considers a set of 50 dialogs.

From the table we can observe that the extractive baselines, manual and automatic transcript behave similarly. The extractive baseline that selects the longest turn from the first 25% of the conversation (Baseline-LB) performs the best, as it was the strongest of the extractive baselines on Italian LUNA corpus. There is a steady drop in performance from manual to automatic transcripts. However, for the baseline that selects the longest turn from whole conversation (Baseline-L), the ASR output yields better results in ROUGE-2 recall with 7% summary length limit (0.017 vs. 0.016 using manual transcripts).

For the abstractive spoken conversation summarization system with the automatic community creation heuristics, for both heuristics we observe that the summary generation from automatic transcripts performs better than the summary generation from manual transcripts – 0.025 vs. 0.027 for WordNet SynSet ID level and 0.028 and 0.030 for word2vec vectors using manual and automatic, respectively. The increase in performance is hard to explain, and needs to be investigated further.

In both settings, the heuristic with word2vec vectors performs the best. Consequently, automatic community creation with word embeddings for similarity computation is the best technique for the abstractive summarization of spoken conversations.

2.3 Discussion

Both approaches to abstractive template-based spoken conversation summarization rely on generalization of human authored summaries and linking summary entities to conversation transcripts. The summary generation step in both approaches consists of slot filling. The difference however, is in the unit of text used for linking and subsequently entity extraction for slot filling.

Summary generation from automatic transcripts (output of Automatic Speech Recognition), for both systems yields mixed results.

We originally planned to apply the same methods on both the French and Italian data in order to compare them. However, the difference in genre between the two corpora make them not comparable and not suitable for a direct application. For example, in the LUNA corpus, synopses do not rely on precise slots but rather on broad descriptions of the problems, which are difficult to extract from the transcripts (potentially requiring statistic text generation), and the DECODA corpus relies heavily on named entities which do not appear in WordNet and cannot be captured by the method developed for the LUNA corpus. We are exploring a unified model but results are not available at the time of writing this document.

2.3.1 Quality of Summaries and the ROUGE score

The quality of the automatically produced spoken conversation summaries is to be evaluated extrinsically in Work Package 1. In this section, on the other hand, we address this question through the literature on summarization.

The spoken conversation summarization approach we follow in Section 2.2 is based on [8]. In the paper,

the authors have conducted crowdsourced evaluation of the automatic summaries comparing them with the human-authored and extractive ones. The evaluation considered criteria such as Quality, Fluency and Informativeness. With the ROUGE-2 score of 0.068 for English, the abstractive template-based summarization system was found to be significantly better than the extractive system on all criteria (second after the human-authored summaries). The authors conclude that the user group preferred template-based summaries over human-annotated extractive summaries. How ROUGE score correlates with the quality of a summary on other SENSEI languages (Italian and French) has not yet been established.

3 Conversation Analysis/Summarization Outputs for Social Media

Online news outlets attract large volumes of comments every day. *The Huffington Post*, for example, received an estimated 140,000 comments in a 3 day period[3], while *The Guardian* has reported receiving 25,000 to 40,000 comments per day[4]. These figures suggest that online commenting forums are important for readers as a means to share their opinions on recent news. The resulting vast number of comments and information they contain makes them relevant to multiple stakeholders in the media business. All user groups involved in online commenting on news would profit from easier access to the multiple topics discussed within a large set of comments. For example, comment posters would be able to gain a quick overview of topics already discussed and insert their contributions at a relevant place in the discussion. Journalists who wrote the news article would have access to multiple conversation topics that their article has triggered and would be able to engage with their readers in a more focused way. Editors would be able to monitor the topics that are most interesting to readers, comment forum moderators' work would be easier and marketers could use conversations grouped around topics for developing personalized marketing strategies.

In the previous deliverable D5.2 we provided a clear specification of the main kind of reader comment summary we aim to produce, this specification in turn being based on the "Town Hall Summary" use case presented in D1.2. In short, we aim to produce (ideally) a summary that a) identifies the main issues discussed in a set of reader comments and b) characaterises opinions offered on these issues, identifying alternative viewpoints, indicating the strength of interest in an issue or support for different viewpoints (aggregation), indicating consensus or agreement among the comment, indicating disagreement among the comment, indicating qualitatively how opinion was distributed (e.g. using phrases like *"Many said this; others said that"*, *"some said"*, *"most said"*), indicating evidence or grounds for a viewpoint and indicating whether the discussion was particularly emotional/heated and if so over what. D5.2 also described initial software components developed to address the challenge of producing such summaries, specifically modules for article-comment linking, topical clustering, cluster labelling and extractive and template-based summarization.

In this section we describe refined and extended versions of the software modules introduced in D5.2. These modules have been extended to improve functionality and performance and to better integrate them into an overall pipeline of modules that communicate via the Conversation Data Repository. The end result is a set of modules that generate three different textual summary types: an extractive summary, an abstractive summary and a template-based summary. A graphical summary is also generated, based on the clustering and cluster labelling methods described below (details of the graphical summary may be found in D6.2).

For extractive and abstractive summarization the following steps are followed. First topic clusters are constructed from the comments for each news article, i.e. clusters containing comments addressing the same topic. To do clustering, features are extracted as described in Section 3.2 and a supervised machine learning approach along with a graph-based clustering method is applied. As features we use

[3] http://goo.gl/3f8Hqu

[4] http://www.theguardian.com/commentisfree/2014/aug/10/readers-editor-online-abuse-women-issues

statistical methods as well as approaches which perform agreement/disagreement analysis, discourse relation analysis and also coreference resolution over the comments. Once the topic clusters are constructed (see Section 3.3 for details) labels capturing the topics are generated. For this we considered both an extractive and an abstractive approach (see Sections 3.4 and 3.5). Finally, summaries are constructed based on the generated labels. For extractive summaries we use the labels to select representative comments from the clusters. The abstractive summaries are generated by extending the labels to full natural sentences. The extractive summarization approach is described in Section 3.6 and the abstractive approach is detailed in Section 3.6.

In addition to these two summarization approaches we have developed a template-based summarization approach. The template-based summaries are filled with data from three different modules (topic extraction, mood prediction and agreement/disagreement detection) and metadata from the article (extractive summary). The template has been designed following the Town Hall Meeting use case, which requires the following information to be included: headline/article summary, key contributors, list of main topics in the article and comments, intensity of emotions associated to topics, consensus or divided opinion on topics. The template-based summary includes: an introduction with the title and subtitle of the article, a list of the most frequent topics, a development, with the emotions and consensus associated to topics, and a conclusion, with the most active contributor. The template is filled in 4 stages: 1) the system extracts the topics from article and comments with LDA, 2) the system predicts agreement/disagreement and mood for each comment and 3) the system matches topics, agreement/disagreement and mood at comment level. Finally 4) the system computes the rank of most active contributors. The template-based summarization approach is described in Section 3.8.

The rest of this section is organised as follows. As most of our components for social media conversation analysis and summarization make use of the USFD gold standard dataset we begin by briefly describing it (section 3.1). Then, since multiple components in the analytics and summarisation pipeline make use of various features extracted from the outputs of modules built in WP3 and WP4, we describe these features (section 3.2). These include: agreement/disagreement between comment pairs; discourse relations between comment pairs; anaphoric relations between comments and between comments and the news article; sentiment. Following this we describe our approach to clustering (section 3.3), our extractive and abstractive approaches to cluster labelling (sections 3.4 and 3.5) and finally our three approaches to summarization: extractive (section 3.6), abstractive (section 3.7) and template-based (section 3.8).

3.1 USFD Gold Standard Data

The USFD gold standard data consist of 18 articles and associated comment sets (at least 100 comments per comment set). Two annotators were used to generate summaries from each comment set; all are coherent and fluent summaries. As part of the process of creating each summary, annotators have first generated and then grouped “labels” (short paraphrases) of each comment. Human-authored reference summaries are then generated from the condensed representation of the comments that label groups provide. However, these label groups also allow us to assemble the comments they summarise into topically related clusters, which can be used as gold standard clusters for evaluating our clustering algorithms. On average there are 8.97 human-created clusters per comment set in the gold standard

cluster data set. A detailed description of this gold standard dataset can be found in deliverable D5.2. The news articles in the gold standard dataset as well as their comments are also currently being annotated for coreference (see Section 3.2.3).

3.2 Feature Extraction

Feature extraction from the conversational social media data serves various downstream components in the analytics and summarization pipeline. For instance, for clustering of user comments by topics, features are used to determine link strength between comments. Based on the strength of the links, comments are either put into the same cluster or not. For our clustering approach we investigated statistical features of words in the comment (see D5.2) but now also features extracted using SENSEI modules such as anaphora resolution, discourse parsing and agreement extraction. Anaphora resolution is used to determine whether two comments refer to the same antecedent. We hypothesise that this information is evidence that can help to determins whether two comments belong in the same topical cluster or not. Similarly, the agreement/disagreement information between two comments as well as discourse relational information is may help to decide whether two comments should be put into the same cluster or not.

Apart from clustering, the features described below are also used by other SENSEI modules. For instance, agreement/disagreement information is also fed into the template-based summarization system as is the sentiment information.

3.2.1 Agreement/Disagreement Extraction

It is very important to understand the level of consensus between bloggers in news conversations. For this purpose, we identified Agreement/Disagreement classification as the main parasemantic argument extraction technique. It consists of the automatic classification of agreement/disagreement labels from text retrieved from social media conversations. Agreement/disagreement classification in asynchronous conversations like social media is a quite novel task [22], and there is no unique definition of it. Some have defined Agreement/Disagreement relations as Quote-Response message pairs and triplets: chains of three messages such that the third one is a response to the second one which is itself a response to the first one. These pairs and triplets are linked by the structure of the thread, where each message is a reply to its parent and is about the same topic [23]. Others defined Agreement/Disagreement as relations between pairs of sentences, belonging to messages in a parent/child relation defined by the thread [24]. To annotate Agreement and Disagreement, some used binary classes (“agree” or “disagree”), some use three (“agree”, “disagree” or “none”) and some others use a scale [25].

First, we defined the agreement and disagreement relation in a formal way [26] as a function that maps pairs of bloggers and messages to polarity values between 1 (“agree”) and -1 (“disagree”), where 0 is neutral.
We defined the agreement/Disagreement relations as:

$$agree(m_{ij}; m_{i'j'}) = \{-1, 1\}$$

where m_{ij} is the parent blogger/message pair, and $m_{i'j'}$ is the child blogger/message pair. The parent m_{ij} temporally precedes the child $m_{i'j'}$. The child $m_{i'j'}$ is the j'^{th} message of blogger $p_{i'}$, referred to the j^{th} message of blogger p_i. In our definition, conversation contains a set of topics appearing inside messages. This is formalized as $T = \{t_{ijk}, ..., t_{nmo}\}$, where t_{ijk} is the k^{th} topic of the conversation contained into the j^{th} message of blogger p_i.

Following this definition we designed the guidelines for the annotation of agreement and disagreement in Italian, a target language for the SENSEI project where there were no corpora available. We identified and extracted the data from the platform of Corriere della Sera[5], one of the most popular Italian-language news websites, attracting over 1.6 million readers every day.

Like other platforms, Corriere provides the structure of replies as metadata, which we used to identify child messages that are direct replies to parent messages. Following this structure, we extracted pairs of comments and built the CorEA corpus. The CorEA corpus contains asynchronous conversations started from 27 news articles of different news categories, from politics to gossip. The corpus contains 2887 messages (135K tokens). The average number of messages per conversation is 106.4. We manually annotated the corpus with Agreement/Disagreement labels. To do so, we recruited two expert annotators, we trained them on the annotation guidelines and evaluated the annotation at message level with Inter-Annotator-Reliability over two and three labels using κ statistics [27]. Results are reported in Table 4.

task	examples	classes	κ
messages	100	3	0.57
messages	50	2	0.85

Table 4: Inter-annotator reliability scores on the annotation of ADRs at message (msg) and sentence (sent) level. The score is computed with *k* over 3 and 2 classes.

Exploiting the gold-standard annotation, we trained and tested language-independent models for the Agreement/Disagreement classification module. There are two versions of the module for agreement/disagreement prediction, Version 1 is taken as the baseline and version 2 is the improved system. Both systems are supervised and based on cross-language models trained on Italian (CorEA Corpus). The module for Agreement/Disagreement prediction version 1 is based on the following features:

- Ratio of @symbols
- Ratio of Punctuation
- Ratio of Apices
- Character-word ratio
- Ratio of Uppercase words
- Avg string Similarity between uppercased words

[5]http://corriere.it

- Median of the radius between word pairs

We trained the system using a 66% of the CorEA corpus and tested it on 33%. We used a Linear Regressor as algorithm, and obtained a Mean Absolute Error (MAE) of 0.42 over a majority baseline (predicting always the mean) of 0.45. We considered these result as the baseline for version 2. Version 2 of the agreement/disagreement prediction module makes use of the following features:

- Ratio of words of 1 character
- Ratio of words of 2 characters
- Ratio of exclamation marks
- Ratio of colons
- Ratio of semicolons
- Ratio of quotes
- Direct reply to the article
- Ratio of Numbers
- Ratio of operators
- Ratio of open parentheses
- Ratio of closed parentheses
- Ratio of positive emoticons

As we did in the previous version, we trained the system using 66% of the CorEA corpus and tested it on 33%. We used a Support Vector Regressor (SMOreg) as algorithm, and obtained a Mean Absolute Error (MAE) of 0.32.

Both versions of the Agreement/Disagreement module output a standardized score that can be exploited as a feature for clustering. Since we developed a language-independent system trained and tested on Italian, we plan to evaluate it also in English.

3.2.2 Discourse Relations

The goal of the Discourse Relation module for Social Media is to detect whether two arbitrary comments to the same article are related to each other discourse-wise. Here we are following Penn Discourse Treebank (PDTB) [28] framework, in which discourse relations are strictly binary: a discourse connective, considered as the predicate, takes exactly two arguments – Arg1 and Arg2 – where Arg2 is the argument syntactically attached to the connective. A discourse connective is a member of a well

defined list of 100 connectives and a relation expressed via such connective is an Explicit discourse relation.

A discourse relation can also hold without the presence of a connective. In the PDTB adjacent sentences within a paragraph were additionally annotated for such Implicit discourse relations. In the implicit discourse relations, a connective can be inserted, but is left implicit. In case a connective cannot be inserted while there is a discourse relation between sentences, the pair is annotated as having an Alternative Lexicalization (AltLex) discourse relation. In the PDTB, in case an adjacent pair of sentences has neither explicit, nor implicit nor AltLex discourse relations, it is additionally inspected for whether the two sentences involve the same entity. Such sentences are annotated as having Entity-based Coherence Relation (EntRel). If the pair does not involve the same entity, it is annotated as No Relation (NoRel). The PDTB is used to train the Discourse Relation Detection module for Social Media, taking as a basis the Discourse Parser of [29].

Traditionally, Discourse Parsing is designed to parse a document, following its structure, i.e. it respects order of sentences in a document. To extend the parser to any arbitrary set of comments it has to undergo modifications. Since explicit relation depends on the presence of a discourse connective, which have a strong preference on the location of its arguments; the module cannot utilize the full discourse parsing pipeline. Otherwise, in case a comment contains a connective, all the 'paired' comments will be detected as involved in a discourse relation. Consequently, the module relies on a Non-Explicit Discourse Relation Detection model of [29].

For training the classification models we have generated No-Relation pairs using reference PDTB annotation, excluding all the sentences involved in inter-sentential relations. Additionally, since in the PDTB arguments of non-explicit relations are stripped of leading and trailing punctuation, the No-Relation pairs were pre-processed. The model is trained using a single feature type – Cartesian product of Brown Clusters of all the tokens from both arguments. The classification is cast as a binary relation vs. no-relation task. The performance of the model on the PDTB development set has F_1 of 0.69 (for relations).

The PDTB models are mainly trained on adjacent sentence pairs; a Social Media comment, on the other hand, often consists of several sentences. Thus, the classification of comments into relation vs. non-relation is performed for all sentence pairs from the pair of comments; and the classifier decisions are aggregated. Since discourse relations are asymmetric, i.e. the order of sentences matters; classification is performed both ways: from comment A to comment B, and from comment B to comment A.

We have experimented with two modes of aggregation: (1) a comment is related discourse-wise if any of the pairs is related, and (2) a comment is related discourse-wise, if the average posterior probability of all the pairs surpasses a certain threshold (e.g. 0.5).

3.2.3 Anaphora Resolution

In this section we describe the Coreference Resolution module, one of the summarization modules for the social media use case.

Coreference (including Anaphora) Resolution [30] is a key intermediate step between core NLP pro-

cessing like tokenisation, sentence splitting, part-of-speech tagging and syntactic chunking or parsing, and higher semantic levels of processing or end goal applications, such as Question Answering [31], Text Summarisation [32] or Information Extraction (ever since the MUC series [33, 34]).

There is a lot of work on Coreference Resolution in the news domain (see [30] for a comprehensive survey on the subject matter), however, as with many other Natural Language Processing tasks, adapting to new domains is an open area of research [35, 36]. One such novel domain is that of online conversation threads as found in social media and online forums, which is one of the two primary target domains of the project SENSEI.

Resolving coreferences in online conversation threads is very challenging. Consider the following example:

(1) C_1**:** The First Sea Lord, Sir George Zambellas, came closest to expressing it, calling the 3bn ship "a national instrument of power". Who is he planning to invade now?
$\hookrightarrow$ C_2**:** Which 'nation'?
$\hookrightarrow$ C_3**:** He said it was an example of a big nation demonstrating what they do ... spend countless billions on a vessel that will at best have no aircraft for at least 6-10 years and when there is enough support vessels to defend this hulking lump. Lets gloss over the anti ship ballistic missiles that could render them sitting ducks.
$\longrightarrow$ C_4**:** Agree regarding the time scale for fixed wing aircraft, however I'm not so sure about your statement with regards to anti ship missiles...
$\longrightarrow$ C_5**:** There's no such thing as an anti-ship ballistic missile...
$\longrightarrow$ C_6**:** I doubt that a ballistic missile would render them sitting ducks because they have to be aimed at a fixed point and aircraft carriers are not fixed, they are moving...

In Example 1, comments $C_{[1-6]}$ constitute a typical online conversation thread; comments C_2 and C_3 are replies to comment C_1 (first level) and comments $C_{[4-6]}$ are replies to comment C_3 (second level). A key entity in this conversation thread is *anti ship ballistic missiles* and the context in which the whole thread lives in is a news article titled 'Supercarrier made in Britain hailed as flagship for Better Together campaign'. Now whilst the news article triggered numerous discussions on war ships, carriers and the Scottish Independence referendum, this was the only conversation thread which discussed *anti ship ballistic missiles*, and hence, this entity is not accessible from outside this thread.

As part of our work in adapting the Coreference Resolution toolkit, BART,[6] to resolve in online conversation threads, we have annotated the OnForumS corpus [37] for coreference, following an annotation scheme which is a variant of the LiveMemories annotation scheme [38], which in turn is based on the ARRAU annotation scheme [39]. In this corpus all noun phrases are taken as mentions, and the whole noun phrase is considered (with all its embedded NPs). All anaphoric relations of identity between any pairs of mentions are annotated. Coordinations are also treated as mentions, and annotated.

We are also currently annotating the USFD data set as described in Section 3.1, which has already been annotated with gold cluster topic labels and summaries, for coreferences following the same annotation methodology as for OnForumS.

The key work on extending BART is on defining new features capturing and exploiting the structure of online conversation threads (see Example 1). We define features around the notion of 'accessibility' (in the discourse sense), which indicates whether a potential antecedent for an anaphor is accessible or not to the anaphor depending on its position in the thread. Currently we are working with three features:

[6] See also Section 4.2 in deliverable D4.2.

1. *Strict Accessibility*: on iff antecedent is on the path of the reply/layout structure
2. *Loose Accessibility*: on iff antecedent is anywhere within the same conversation thread
3. *Blogger Quote*: specifically devised to deal with bloggers quoting other bloggers when replying

We added feature *Strict Accessibility* into BART by including an extractor and modifying its discourse model accordingly and we ran a preliminary 10 fold cross validation experiment on the OnForumS data set. The results are shown in Table 5. The upper bound and baseline are the same as the ones for the cross domain coreference experiments presented in Section 4.2 in deliverable D4.2.

	Recall	**Precision**	**F1**
Standard domain (news) – upper bound	$54.5\%(\sigma = 9.6)$	$68.5\%(\sigma = 13.6)$	$60.7\%(\sigma = 9.6)$
In-domain (online forums) 10XVal – baseline	$49.2\%(\sigma = 5.7)$	$56.6\%(\sigma = 4.8)$	$52.5\%(\sigma = 5.0)$
+ strict accessibility	$48.75\%(\sigma = 5.8)$	$56.54\%(\sigma = 4.9)$	$52.28\%(\sigma = 5.1)$

Table 5: Coreference resolution performance for strict accessibility (with std. dev. across folds or files within brackets).

From Table 5 we can see that there is a slight decrease in performance from $F1 = 52.5$ to $F1 = 52.28$ and a slightly higher standard deviation of $\sigma = 5.1$. We are currently carrying out an error analysis to work out the reason for the decrease in performance.

3.2.4 Sentiment Extraction

We previously adapted existing GATE [40, 41] tools from the ARCOMEM project[42] into a component for carrying out the following tasks for English:

- standard NLP (e.g., tokenization, POS-tagging, lemmatization) to a high standard;
- named-entity recognition to a high standard;
- event detection at a baseline level on our data;
- sentiment detection at a baseline level on our data.

The GATE pipeline is wrapped in a Java component which works well with the conversational repository; this component polls the repository (using the advanced query system described in the repository documentation in previous deliverables) for batches of documents that it has not yet processed, processes them, and then sends annotation sets and document features back to the repository, including a flag document feature used in subsequent queries to distinguish the processed documents.

The Java component is highly configurable so it can be used to run other GATE pipelines over repository documents and send back to the repository any specified document features and annotations.

Since this work was last reported, we have improved the repository integration of this component to make it easier for other components to use it. Work in progress includes tuning it for better performance on our data, making use of its output in the prototype user interface, and evaluating its output's usefulness as input to other components.

Another route on sentiment extraction that is taken is based on Convolutional Neural Networks (CNN). CNNs using word representations as input are well suited for sentence classification problems [43] and have been shown to produce state-of-the-art results for sentiment polarity classification [44, 45]. Pre-trained word embeddings are used to initialize the word representations, which are then taken as input of a text CNN. Our approach consists of learning polarity classifiers for three types of embeddings, based on the same CNN architecture. Each set of word embedding models the input text according to a different point of view: lexical, part-of-speech and sentiment. A final fusion step is applied, based on concatenating the hidden layers of the CNNs and training a deep neural network for the fusion.

Lexical		**Part-of-speech**		**Sentiment**	
good	**bad**	**good**	**bad**	**good**	**bad**
great	good	great	good	great	terrible
bad	terrible	bad	terrible	goid	horrible
goid	baaad	nice	horrible	nice	shitty
gpod	horrible	gd	shitty	goood	crappy
gud	lousy	goid	crappy	gpod	sucky
decent	shitty	decent	baaaad	gd	lousy
agood	crappy	goos	lousy	fantastic	horrid
goood	sucky	grest	sucky	wonderful	stupid
terrible	horible	guid	fickle-minded	gud	:/
gr8	horrid	goo	baaaaad	bad	sucks

Table 6: Closest words to "good" and "bad" according to different regimes for creating word embeddings: lexical, part-of-speech and sentiment (described later in the paper).

We train different word representations because classical lexical word2vec embeddings have been shown to represent positive and negative words, such as 'good' and 'bad' at the same location in the embedding space (see Table 6). In our system, in addition to lexical embeddings, we train embeddings based on concatenated words and part-of-speech tags. These embeddings can better model word senses that are disambiguated by part of speech tags [46, 47, 48]. The final embedding variant is sentiment embeddings [49] that are trained on tweets that contain smileys. The polarity of the smiley is concatenated to the words used in the context window of the skip-gram model. In order to account for efforts to create high quality polarity lexicons, we concatenate the word representations with binary features based on a range of such lexicons (MPQA [50], Opinion Lexicon [51] and NRC Emotion lexicon [52]) as well as simple sentence-level morphological features (lenthening, emoticons, punctuation).

The system consists of three CNNs over 1- to 5-gram word windows (convolutional feature map of size 500), with ReLU activations, max-pooling and hidden layers of 500 units. In a first stage, those systems are trained on the target task (3-class sentiment polarity prediction) and the activations at the hidden layer level are used as input of a Deep Neural Network (DNN) trained again on the target task. At that point, sentence level features (lexicon and morphological) are happend to the rest of the input. Word representations are pre-trained on a 90 million tweet unlabelled corpus collected by the project.

Corpus	SENSEI-LIF	Rank
Twt2016	**63.0**	**2**
Twt2015	66.2	2
Twt2013	70.6	3
Twt2014	74.4	1
SMS2013	63.4	3
LvJn2014	74.1	1
TwtSarc2014	46.7	8

Table 7: Overall performance of the SENSEI-LIF sentiment analysis systems.

On the Semeval 2016 sentiment polarity task, that system ranked 2nd out of 34 participants, with non significant differences with the best system. As shown in Table 7, the system also performed very well on out-of-domain corporq such as the Live journal 2014 and SMS 2013 corpora. The full description of the system is available in the Semeval proceedings, and the component is being used for sentiment analysis in the various SENSEI systems and evaluations.

3.3 Comment Clustering

As outlined in Section 3 clustering is an important aspect of conversational analysis. It is used to determine comments that are topically similar. In this section we describe our clustering approach to address this challenge.

3.3.1 Method

In the deliverable D5.2 we described a graph-based clustering algorithm. Since then it has been updated with new features including features described in the previous sections. However, the core idea is the same as described in the deliverable D5.2. In brief our graph-based clustering approach is based on the Markov Cluster Algorithm (MCL) [53]. The nodes (V) in the graph $G(V, E, W)$ are the comments. Edges (E) are created between the nodes and have associated weights (W). Each comment is potentially connected to every other comment using an undirected edge. An edge is present if the associated weight is greater than 0. Such a graph may be represented as a square matrix M of order $|V|$, whose rows and columns correspond to nodes in the graph and whose cell values $m_{i,j}$, where $m_{i,j} > 0$, indicate the presence of an edge of weight $m_{i,j}$ between nodes V_i and V_j. Following the recommendation in [53] we link all nodes to themselves with $m_{i,i} = 1$. Other edge weights are computed based on comment-comment similarity features. In D5.2 we used only a few features to compute the similarity between comments. In the following section we give full detaisl about the previous and the additional features we use in our clustering approach.

3.3.2 Clustering Features

To weight an edge between two comments C_1 and C_2 we use the features below. When computing these features, except the *NE*$_{overlap}$ feature, we use terms to represent a comment instead of words, since we have found that terms are more suitable for computing similarity between short texts than single words [54]. Terms are noun phrase-like word sequences of up to 4 words. Terms are extracted using POS-tag grammars such as *NN NN*. To extract terms we first POS tag the words. Once this is done sequences of POS tags (consecutive POS tags) are constructed up to 4 tags in length. Finally, for each sequence we check whether it occurs in the POS-tag grammar list. If yes, the corresponding words are taken as the term. The POS-tag grammars are automatically generated. Details about the grammar generation can be obtained from [55].

We use a weighted linear combination of the features to compute comment-comment similarity:

$$Sim\_Score(C_1, C_2) = \sum_{i=1}^{n} feature_i(C_1, C_2) * weight_i \tag{2}$$

To obtain the weights we train a linear regression[7] model using the gold standard clusters as described in Section 3.1 . We used 6 fold cross validation, i.e. trained on 15 and tested on 3 articles[8].

We create an edge within the graph between comments C_i and C_j with weight $w_{i,j} = Sim\_Score(C_i, C_j)$ if Sim_Score is above 0.3, a minimum similarity threshold value set experimentally.

Previous Features

- **Cosine raw count:** Cosine similarity [56] is the cosine of the angle between the vector representations of C_1 and C_2 :

$$cosine(C_1, C_2) = \frac{V(C_1) \cdot V(C_2)}{|V(C_1)| * |V(C_2)|} \tag{3}$$

  where $V(.)$ is the vector holding the raw frequency counts of terms from the comment.

- **Cosine TF*IDF:** Similar to the first cosine feature but this time we use the tf*idf measure for each term instead of the raw frequency counts. The idf values are obtained from 3,362 news articles and their comments. These news articles have been obtained from the Guardian.

- **Cosine modified:** Liu et al. [57] argue that short texts can be regarded as similar when they have already a predefined *D* terms in common. We have adopted their modified cosine feature:

$$cosine_{modified}(C_1, C_2) = \begin{cases} \frac{V(C_1) \cdot V(C_2)}{D}, & \text{if } V(C_1) \cdot V(C_2) \leq D \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

  We have set D experimentally to 5.

[7] We used Weka (http://www.cs.waikato.ac.nz/ml/weka/) implementation of linear regression.
[8] We also apply regularisation to filter or downgrade the contribution of non-useful features.

- **Dice:**

$$dice(C_1, C_2) = \frac{2 * |I(C_1, C_2)|}{|C_1| + |C_2|} \tag{5}$$

where $I(C_1, C_2)$ is the intersection between the set of terms in the comments C_1 and C_2.

- **Jaccard:**

$$jaccard(C_1, C_2) = \frac{|I(C_1, C_2)|}{|U(C_1, C_2)|} \tag{6}$$

where $U(C_1, C_2)$ is the union of sets of terms in the comments.

- **NE overlap:**

$$NE_{overlap}(C_1, C_2) = \frac{|I(C_1, C_2)|}{|U(C_1, C_2)|} \tag{7}$$

where $I(C_1, C_2)$ is the intersection and $U(C_1, C_2)$ is the union set between the unique named entities (NEs) in the comments. We use the OpenNLP tools[9] to extract NEs.

- **Same thread**: Returns 1 if both C_1 and C_2 are within the same thread otherwise 0.
- **Reply relationship**: If C_1 replies to C_2 (or vise versa) this feature returns 1 otherwise 0. Reply relationship is transitive, so that the reply is not necessarily direct, instead it holds: $reply(C_x, C_y) \wedge reply(C_y, C_z) \Rightarrow reply(C_x, C_z)$

Additional Features We distinguish between features that come from upstream SENSEI modules (described about in section 3.2) and those that do not. The following are from the second category, i.e. ones that are not produced by upstream SENSEI components.

- **Different variations of Previous Features**: Features (Cosine to Jaccard) which use terms to compute the similarity between two comments are replicated with n-grams. This means we run each feature with single words (uni-gram), two, three and four consecutive words (bi-grams, three-grams and four-grams).
- **Same author**: Returns 1 if both C_1 and C_2 are written by the same person.
- **Word2Vec**: Word embeddings using Word2Vec [58] have been extensively used to measure the semantic similarity between words. Our word embeddings comprise the vectors published by Baroni et al. [59]. To measure the similarity between a pair of comments we first remove from each comment stop-words as well as punctuation, query for each word its vector representation and create a averaged sum of the word vectors. The number of remaining words in each comment is used to average that comment. Finally, we use the resulting averaged sum vectors and determine their similarity using the cosine similarity measure.

The following features are computed by upstream SENSEI modules. We have integrated outputs of the anaphora resolution module into clustering as intuition suggests that if two comments refer to the same

[9] https://opennlp.apache.org/

antecedent then those two comments may well belong to the same topical cluster.[10] We also expllited output from the agreement/disagreement classifier as well as from the discourse relation module. In the following we detail the features from these upstream SENSEI modules that are used along with all the previous features to determine similarity between two comments:

- **Has same annotation id**: This is a binary feature and returns 1 if both comments share a common annotation id otherwise 0.
- **Cosine between the annotation IDs**: We determine all the annotation' ids from both comments and compute the cosine angle between annotation ids. Beforehand a dictionary vector is created consisting of all the unique annotation ids, e.g. "1,2,3,4,5". Each comment is then represented using this vector. For each annotation id either a "1" or "0" is included in the vector depending whether that id appears in the comment or not. E.g. if the comment contains annotation ids "2" and "4" its vector gets the spape "0,1,0,1,0". Based on these vectors we determine the cosine angle between two comments.
- **Cosine between the mentions in the anaphora chain**: A chain in the anaphora resolution output consists of an antecedent and all anaphors referring to it. For each anaphor in each comment we determine its anaphora chain, collect the words in the antecedent (e.g. "Buckingham Palace") and as well as of the anaphors (e.g. "The Palace"). Then we merge the words of the anaphora chains in word vectors and compute the cosine angle between the merged versions of the word vectors. In the cosine vectors we use single words.
- **Cosine between the contents enriched with the mentions in the anaphora chain**: In the previous feature we used only the anaphora chains. In this feature we use the words in the comments and the anaphora chains (enriching the comment contents with words from the anaphora chains) to compute the cosine score.
- **Cosine between the mentions in the anaphora chain, in term fashion**: This is similar to the *Cosine between the mentions in the anaphora chain* feature except we treat the entire antecedent/anaphor as one unit within the cosine method. This means instead of e.g. using "Buckingham" and "Palace" as separate entries in the cosine vectors we use "Buckingham Palace" as one unit. This is similar to the notion of terms described earlier.
- **Cosine between the contents enriched with the mentions in the anaphora chain, in term fashion**: We do as the previous feature and also add the terms extracted from the comments. In fact this feature is similar to the *Cosine between the contents enriched with the mentions in the anaphora chain* feature except we use terms instead of single words.
- **Word2Vec on mentions in the anaphora chain**: This feature is similar to the *Cosine between the mentions in the anaphora chain* feature. The difference is that we first retrieve word vectors using

[10] The anaphora approach discussed earlier in this report aims to tackle the anaphora resolution with awareness of the comment conversations. However, at the time of writing this report the integration of this conversation-aware anaphora resolution has not been finished. Thus the experiments and results reported here are with the a version of the BART system that is not thread-aware. To perform the experiments in the deliverable we merged the article and the comments in one file where the conversation structure is flattened out. This file was processed by BART for anaphora resolution and the output was taken for clustering purposes.

Word2Vec and use these in the cosine computation instead of the words themselves. The cosine computation follows the same idea as presented in the *Word2Vec* feature described earlier.

- **Word2Vec on mentions in the anaphora chain as well as the comment contents**: In addition to the previous feature we also add the comment contents and compute the cosine between the Word2Vec representations.
- **Agreement Feature**: This features returns 3 categories: 1 if the two comments agree each other, -1 if they disagree and 0 if there is no such relationship.
- **Discourse Relation**: This features returns a score between -1 and 1. The positive scores (> 0) indicate that two comments relate to each other. The closer the score is to 1 the more confidence there is in a positive relationship. Negative scores cover non-relationship. I.e. if the score is negative there is no discourse relationship between the two comments.

3.3.3 Evaluation

For evaluation the automatic clusters are compared to the gold standard clusters described in Section 3.1. Amigo et al. [60] discuss several metrics to evaluate automatic clusters against the gold standard data. However, these metrics are tailored for hard clustering. Although our graph-based approach performs hard clustering, the gold standard data contains soft clusters. Therefore, the evaluation metric needs to be suitable for soft-clustering. In this setting hard clusters are regarded as a special case of possible soft clusters and will likely be punished by the soft-clustering evaluation method. We use fuzzy BCubed Precision, Recall and F-Measure metrics reported in [61] and [62]. According to the analysis of formal constraints that a cluster evaluation metric needs to fulfil [60], fuzzy BCubed metrics are superior to Purity, Inverse Purity, Mutual Information, Rand Index, etc. as they fulfil all the formal cluster constraints: *cluster homogeneity*, *completeness*, *rag bag* and *clusters size versus quantity*. The fuzzy metrics are also applicable to hard clustering.

Llewellyn et al. [63] apply LDA and K-Means, as well as simple metrics such as the cosine measure to cluster news comments. The authors report LDA to be the best performing system. Thus we use LDA as our baseline and replicate the experiments reported by them. For each test article we train a separate LDA model on its comments. In training we include the entire comment set for each article in the training data, i.e. both the first 100 comments that are clustered and summarised by human annotators, as well as the remaining comments not included in the gold standard. In building LDA model we treated each comment in the set as separate document.

3.3.4 Results

The results of the evaluation are shown in Table 8.

From the results in Table 8 we can see that the performance over the different settings stays stable. The results also show that the features without the SENSEI modules already achieve the same performance as with adding any SENSEI module. From the resutls we see that our systems significantly outperform the LDA baseline that was report by Llewellyn et al. [63] as the best performing system for

	baseline Features	agreement	discourse	BART	ALL	LDA	D5.2	Human to Human
Fuzzy B^3Precision	0.517	0.517	0.516	0.517	0.512	0.23	0.30	0.59
Fuzzy B^3Recall	0.375	0.375	0.375	0.376	0.376	0.17	0.33	0.58
Fuzzy B^3FMeasure	0.434	0.434	0.434	0.434	0.433	0.18	0.31	0.58

Table 8: Cluster evaluation results. The scores shown are macro averaged. For all systems the metrics are computed relative to the average scores over Human1 and Human2.

news comment clustering. However, the automatic results compared to *Human to Human* scores are significantly lower and indicate that there is still room to improve.

Note that as described above these results are obtained based on regression models trained using the USFD gold standard data. In the deliverable D5.2 we reported clustering results using models trained on an automatically derived training corpus. This training corpus consisted of training examples (pairs of comments) collected based on shared quotes (both comments quote the same sentence) from the article (positive examples) and negative comment pairs taken from different comments sets, i.e. random pairs of comments from two different news articles. In D5.2 we also used only the baseline features excluding the features added afterwards ("previous features"). the column "D5.2" shows these results. As we can see from these figures the clustering performance went significantly up. Since we have used more or less the same regime of features we think that this substantial improvement is due to using the USFD gold standard data for training.

3.4 Cluster Labelling: Extractive Approach

In many application domains such as search engine snippet clustering [64], summarizing YouTube video comments [65] or online comments to news [66], grouping text segments by topic has been identified as a major requirement for efficient search or exploration of text collections. This has given rise to a substantial body of work in statistical topic modelling.

In the online news domain, thousands of reader comments are produced daily, and the ability to identify topics in comment streams is of vital importance for all interested in gaining a quick overview of what readers say in their comments. However, to be of use to the end users, the topic clusters need to be accurately labelled in such a way that the content of a cluster is clear and easily accessible to the user.

Producing "good labels" is challenging since what constitutes a good label is not well defined. The most common way to label topic clusters is withthe top-n key terms that characterise the topic. This approach has been repeatedly reported as less suitable than generating "textual labels" [67] for topics, which do not consist of key terms, but meaningfully represent the topic cluster [68, 69].

Such textual labels in most studies are still extractive in that the most likely label is directly extracted from textual sources [68, 69]. To overcome the limitation of key term-based labelling, which relies on the label being actually present in the topic cluster, many studies use external resources, most notably Wikipedia, for deriving topic labels. In the online news domain the news article triggers the comments

and the readers will have some expectations on the content of comments with respect to what they have read in the article. For this reason it seems plausible to use the news article as an external source for labeling topic clusters of comments.

Fewer approaches have been reported which abstract away from the content and attempt to label topic clusters with concepts that most likely represent the topics. Hulpus et al. [70], for example, present a graph-based approach to labelling that uses DBpedia concepts. The advantage of such an approach is a more abstract semantic representation of the topic cluster that may be more related to the way humans would label the clusters. This approach has been evaluated for the online news domain in Aker et al. [71] and attains good scores on an information retrieval task, but it is unclear how useful these labels are for the end users.

Despite an abundance of work on topic labelling in test collections, studies on topic labelling in conversational data are comparatively rare and recent. Louis and Cohen [72] tackle the problem of identifying labelled topics in technical forums using grammar models. Chang et al. [73] label multimedia contributions on Google+ using a supervised ensemble learning approach with crowdsourced training and gold standard data. Aker et al. [71] apply an abstractive labelling method from Hulpus et al. [70] on labelling topic clusters of comments to news. The details of our abstractive labeling approach are provided in Section 3.5. The labelling method performs well on a information extraction task (similar to the evaluation in Aletras et al. [67]). Joty et al. [74] identify and label topics in e-mail conversations and comments to blogs.

3.4.1 Method

The work on topic labelling in text collections has made extensive use of Wikipedia article segments as an external source of information for better labels (e.g. Lau et al. [68]) . However, for conversational data, Joty et al. [74] argue that external resources like Wikipedia titles used in previous work are too broad for their e-mail and blog domain as indicated by the fact that none of their human-created labels in their development set appears in a Wikipedia title. Chang et al. [73], however, use human-generated labels for posts, suggesting that post-internal information is not suitable for deriving labels.

In our approach we aim to do both. We extract labels from the comments as well as from external resources which is in our case the news article that triggered the comments. We adopt a phrase or term as the most suitable linguistic unit to represent labels, as evidenced by several previous studies [69, 74, 71].

Our labeling approach is supervised. Using the entire manually annotated gold standard set of summarised Guardian articles plus comment sets (see details in Section 3.1), where the annotation includes the manually clustered comments, human summaries and backlinks between summary sentences and human clusters, we collect training data to train a regression model for extracting labels for automatic clusters.

To do this we first extract terms from the article as well as comments and represent them with features. We also assign each term a score that varies between 0 and 1. A score of 0 indicates that the term is not a good label whereas a score of 1 means that the term is an excellent label for the comment cluster. We obtain the term scores using the human summaries generated for the Guardian articles.

For these human summaries we have the information about what sentences in the summary link to which human assembled clusters. If the question is to answer whether the term *X* is a good label for cluster *Y* then we collect the sentences from the human summaries that are linked to the *Y* cluster and compare that term *X* with terms extracted from the summary sentences. The comparison is based on a Word2Vec similarity computation and results in a score that varies between 0 and 1. Following this approach we collect training data consisting of terms represented by features and the similarity score to be predicted. Once we have such training data we use linear regression[11] to train a regression model where the combination of the features is based on a weighted linear combination.

In the test case, i.e. running the cluster labeling approach on a cluster to generate a new label, we again determine terms from the article and the comments, extract features, use the regression model to score the terms and select the best scoring term as the label for that cluster.

In the next section we will give detail description about the features we used for representing candidate labels.

3.4.2 Features

In the cluster labeling approach we use several features extracted from the news article and the comments. Features extracted from the article are based on the following motivation. To define features for the terms extracted from the news article we have investigated a set of 1.7K Guardian news articles along with their user generated comments. On average we have 206 comments per news article. From each news article we have extracted terms and analysed whether they have been also used in the user generated comments. Our analysis shows that 35% of the terms extracted from the news article are also mentioned in the comments. We also found out that mostly terms from the title and first sentence (55% and 60% respectively) were mentioned in the comments. Terms extracted from other parts sentences from 2 to 6 and sentences from 7 till the end of the article) were mentioned only around 45% and 33% respectively. Around 43% of comments mentioned at least one or more terms extracted from the article.

Based on this analysis we derived the following features:

- **#Term in title**: This feature counts how many times a term occurs in the article title.
- **#Term in first sentence**: This feature counts how many times a term occurs in the first sentence of the article.
- **#Term in 2-6 sentences**: This feature counts how many times a term occurs in the sentences including 2-6.
- **#Term in sentences after 6**: This feature counts how many times a term occurs in the 7th and the following sentences till the end of the article.
- **#Term in the entire article**: This feature counts how many times a term occurs in the entire article.

[11] We use the Weka's implementation of linear regression. Weka can be obtained by following the link: http://www.cs.waikato.ac.nz/ml/weka/.

- **Article centroid similarity**: This feature computes the cosine similarity between the term and the article centroid. The similarity is based on Word2Vec.

In addition to these "article" based features we also compute the following features:

- **Term length**: This feature returns the number of words in the term.
- **#Term in all comments**: This feature counts how many times a term occurs across all comments on the article.
- **#Term in all comment in the cluster**: This feature counts how many times a term occurs in all comments within a cluster.
- **Cluster centroid similarity**: This feature computes the cosine similarity between the term and the cluster centroid. The similarity is based on Word2Vec.
- **#Term in article + comments**: This features counts how many times a term occurs together in the article and all the comments.

3.4.3 Evaluation

We are currently working on setting up an evaluation exercise to assess the performance of this extractive labeling approach.

3.5 Cluster Labelling: Abstractive Approach

In contrast to our extractive approach we also developed an abstractive approach based on the graph-based topic labelling algorithm of Hulpus et al. [75] which uses DBPedia [76]. We modified it for comment cluster labelling.

3.5.1 Method

Our use of the Hulpus et al. [75] method proceeds as follows.

1. Topics of the cluster are extracted using an LDA model. The LDA was trained on large collection of Guardian new articles along with their associated comments. Topic words spewed out by the LDA model can be noisy. The hyperparameters were trained by observing the output. Additionally a large stop word list was included to reduce the effect of the same. The number of topics (k) to assign was determined empirically. We experimented by varying k between 2 and 10 and chose k=5 based on the clarity of the labels generated.

I was working in the south of France during the 2003 heatwave and I would wear a hat which I kept soaked in water while at work, lie in a bath full of cold water before going to bed to get my body temperature down and drink gallons of water through out the day and before going to bed. My thermometer only went up to 50 degrees centigrade so I don't know how hot it was during the day but at 3 in the morning it was 32 degrees centigrade. I also avoided alcohol. It was hot. Here in Devon it's rarely been under 75 since June. In these conditions, dogs die due to stupid owners leaving them in cars, people under-hydrate and pass out, wild-fires start due to fag ends on dry grass, all sorts of stuff happens. It depends where you are, but some of the UK is baking.....

Table 9: **System Generated Label**:*Occupational safety and health*

2. A separate label is created for each such topic, by using the top 10 words of the topic (according to the LDA model) to look up corresponding DBPedia concepts. We take the most-common sense. The 10 word limit is to reduce noise. Each topic word corresponds to DBPedia concept, and can be treated as a graph node on the DBpedia. This forms the concept graph for each topic word. Less than 10 DBPedia concepts may be identified, as not all topic words have an identically-titled DBPedia concept.

3. The individual concept graphs so-identified are then expanded using a restricted set of DBPedia relations. To limit noise, we include only *skos:broader, skos:broaderOf, rdfs:subClassOf, rdfs*. The graph expansion is limited to two hops. Upon increasing the expansion perimeter more abstract concepts will be included.

4. The DBPedia merge operation is then applied on the sub graphs to create a larger graph encompassing all topic words and their peripheral concepts.

5. The central node of the merged graph is identified, providing the label for the topic. To do so, graph centrality measures are used. Among different centrality measures, *Closeness_centrality* is used to find the central node. The intuition is that the label thus obtained should encompasses all the abstract concepts that the topic represents.

A sample cluster along with the label generated by the above method is shown in Table 9.

3.5.2 Evaluation

To evaluate the association of comment clusters with labels created by the cluster labelling algorithm, we create an annotation task by randomly selecting 22 comment clusters, developed using method mentioned in section 3.3, along with their system generated labels. In the annotation bench for each comment cluster label, three random clusters are chosen along with the comment cluster for which the system generated the label. Three annotators (A, B, C) are chosen for this task. Annotators are provided with a cluster label and asked to choose the comment cluster that best describes the label from a list of four comment clusters. As the comment clusters are chosen at random, the label can correspond to more than one comment cluster. The annotators are free to choose more than one instance for the label, provided it abstracts the semantics of the cluster in some form.

Annotators	A-B	B-C	C-A	Overall
Agreement	0.76	0.45	0.64	0.61

Table 10: Annotator agreement (Fleiss Kappa) for comment labelling over 22 comment clusters

In some instances, the comment label can be too generic or even very abstract. It can happen that a label does not correspond to any of the comment clusters. In such cases, the annotators are asked not to select any clusters. These instances are marked NA (Not Assigned) by the annotation bench. Inter-annotator agreement is measured using Fleiss Kappa metric [77]. The output of the cluster labelling algorithm is then evaluated with the annotated set using standard classification metrics.

3.5.3 Results

Tables 10 and 11 present results from the evaluation of the automatically generated comment cluster labels. Table 10 shows the agreement between pairs of annotators and overall, as measured by Fleiss' Kappa on the decision: given the label, which cluster does it describe best. Overall there is a *substantial agreement* of $\kappa = 0.61$ between the three annotators. The annotator pair B-C, however, achieves only *moderate agreement* of $\kappa = 0.45$, suggesting that some annotators make idiosyncratic choices when assigning more generic abstractive labels to clusters.

Table 11 shows the evaluation scores for the automatically generated labels, given as precision, recall and F scores results, along with the percentage of labels not assigned (NA) to any cluster. Overall, annotators failed to assign labels to any cluster in 40.9% of cases. In the remaining cases, where annotators *did* assign the labels to clusters, this was done with fairly high precision (0.8), and so as to achieve an overall average recall of 0.5, suggesting that meaningful labels had been created.

Annotator	Precision	Recall	F-score	NA%
A	0.78	0.32	0.45	59.1 (13/22)
B	1.00	0.45	0.62	54.5 (12/22)
C	0.62	0.73	0.67	9.1 (2/22)
mean	0.80	0.50	0.58	40.9

Table 11: Evaluation results of the cluster labeling system for each of the 3 annotators. NA corresponds to the number of labels not assigned.

3.6 Extractive Summarization

We modified our extractive summarization described in the deliverable D5.2 by basing it on the labels generated from the clusters. Our motivation behind this is that labels extracted from the cluster determine the topic of the cluster and comments entailing the label(s) are a good topic representative of the cluster for the reader. The steps of the extractive summarizer are detailed below:

System	R1	R2	R-SU4
D5.3-Extractive	0.355	0.051	0.143
D5.2-Baseline	0.42	0.05	0.13

Table 12: Results for the extractive summaries.

- **Generating comment clusters**: This is performed using our graph-based clustering approach as described in Section 3.3.
- **Generating cluster labels**: This is performed using our clustering labeling approach as described in Section 3.4. Note the output of this process is a ranked list of cluster labels.
- **Selecting representative comments**: We select from each cluster an entire comment that best represents the cluster. For this we determine the comment that best covers the labels of that cluster. The coverage is determined using word2Vec similarity. We compute the cosine angle between the vector representation of the labels with the vector representation of the comments. The vectors are obtained using word2Vec. The comment with highest cosine is selected for inclusion in the summary.

3.6.1 Evaluation

To assess the quality of extractive summarization we use the gold standard summaries described in Section 3.1. As our baseline system we use the best scoring system reported in D5.2. In this baseline system we take the centroid of each thread and compute its similarity to the lead part of the article using cosine similarity. The value of the cosine measure is used to sort the threads from most to least similar. After sorting the threads, comments closest to the thread centroid are included in the summary.

ROUGE In the assessment we compared the automatically generated summaries against model summaries written by humans using ROUGE [78]. Following the Document Understanding Conference (DUC) evaluation standards we used ROUGE 1, ROUGE 2 (R2) and ROUGE SU4 (RSU4) as evaluation metrics [79] . ROUGE 1 and 2 give recall scores for uni-gram and bi-gram overlap respectively between the automatically generated summaries and the reference ones. ROUGE SU4 allows bi-grams to be composed of non-contiguous words, with a maximum of four words between the bi-grams.

3.6.2 Results

From the results in Table 12 we see that the current extractive summaries are lower in R1 scores, more or less equal in R2 and better in RSU4.

3.7 Abstractive Summarization

Unlike the extractive summarization system described in Section 3.6 the aim of the abstractive summarization is to use pieces of information from the comment clusters and write a natural language summary. Abstractive summarization is a complex process. In SENSEI, without going into this complex process, we make use of cluster labels extracted as described in Section 3.4 and glue them together using manually generated template patterns. The patterns are used to introduce label that is assumed to entail a topic.

3.7.1 Method

The input to our abstractive summarization approach is a set of comments along with the news article. To summarize these comments our abstractive summarization approach consists of five steps:

1. **Generating comment clusters**: This is performed using our graph-based clustering approach as described in Section 3.3.
2. **Generating cluster labels**: This is performed using our clustering labeling approach as described in Section 3.4. Note the output of this process is a ranked list of cluster labels.
3. **Ordering the labels**: From each cluster the best scoring label is taken. The labels are then sorted according to the size of the cluster they came from. The size of the cluster is determined based on its number of comments.
4. **Selecting patterns to glue with the labels**: For each label we select the pattern to glue together. Note the pattern always precedes the label and together they make a full sentence. The patterns are ordered. The first pattern introduces the label that comes from the cluster with most comments in it. The next pattern introduces the next label with the second greatest cluster size, etc.
5. **Selecting example sentences from the cluster**: In addition to step 4 we also select for each cluster label an example sentence extracted from the comments of that cluster. This step involves first the determination of identifying sentences in the comments, determining terms in each sentence and finally computing the Word2Vec based similarity of terms of each sentence to the centroid of the cluster. In addition to this we also compute the cluster label Word2Vec similarity to the terms from the candidate sentence and add it to the centroid similarity. The summed similarity figures are used to score the sentences and sort them in descending order. We select the sentence with the highest score. In the final abstractive summary the sentences extracted as example follow the sentences containing the cluster label.

3.7.2 Evaluation

Similar to the extractive summarization approach described in Section 3.6 we evaluate the abstractive summaries.

System	R1	R2	R-SU4
D5.3-Abstractive	0.393	0.058	0.165
D5.3-Extractive	0.355	0.051	0.143
D5.2-Baseline	0.42	0.05	0.13

Table 13: Results for the abstractive summaries.

3.7.3 Results

Table 13 shows the results for the abstrative summaries.

From Table in 13 we can see that the abstractive summaries are substantially better compared to the extractive summaries. This is also the case for the D5.2 baselines summaries with respect to R2 and R-SU4. Figure 4 shows an example output of the abstractive summarizer.

Most of the comments talk about the topic “people with mental health issues” . For example people say ”My brother in law has a number of mental health issues including paranoid schizophrenia.”

A good amount of contributors discuss the matter “police officers to classify people”. An example of such discussion is “The police aren’t doctors and they shouldn’t try to be.”

Some people also share their opinions about the topic “police access”. An example of such opinion is “This is sadly what can happen when the police become involved with the vulnerable Moreover what difference would it have made had the police access to his records?”

Furthermore, a few discussions entail the subject “school talk to social services” . E.g. “Do you actually know what data social services and the police hold about you and whether it’s accurate?”

Another few mention the topic about “data protection act principles” . A good example for this is the comment extract “Don’t forget we are talking about sensitive personal data here.”

In addition, some minor discussions are about the topic “police officer to preserve freedom”. An examplar of such discussion is “It should be recognised as the duty of every police officer to preserve freedom.”

Finally, in few comments the discussion goes around the topic “crime pre distribution pre taxation” The following sentence is an evidence for such a discussion “Pre crime Pre distribution Pre taxation It’s all about the state screwing the public and the left are cheering it on.”

Figure 4: Example abstractive summary.

3.8 Template-based Summarization

During Y2 we designed a concise template-based summary that included all the requirements of the Town Hall Meeting scenario, namely:

1. headline/article summary,

2. key contributors,
3. list of main topics in the article and comments,
4. intensity of emotions associated to topics,
5. consensus or divided opinion on topics.

The template-based summary, described in detail in Deliverable 5.2, is depicted in Figure 5:

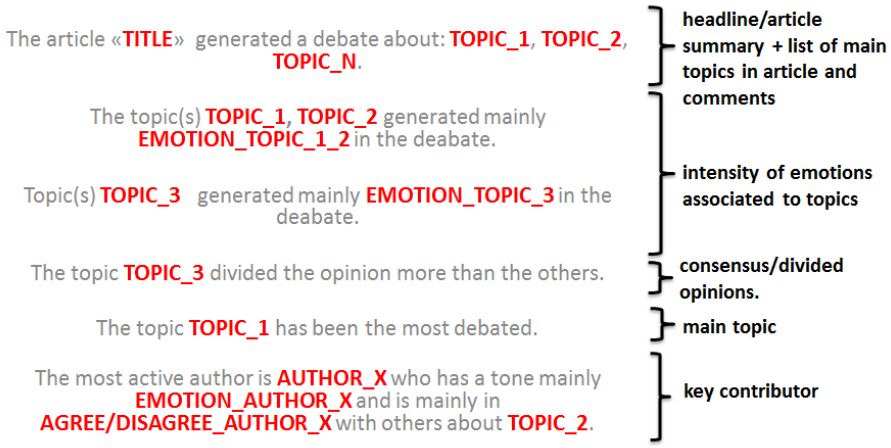

Figure 5: Template-based summary schema. In red and bold there are the variable fields to be filled in, in black the fixed template.

There are fixed parts (in black) and slots filled by the aggregation of the outputs of different modules (in red), such as topic, agreement/disagreement and mood extraction modules. From Y2, there are two main improvements in the template-based summarization:

- improvements in the performance of agreement/disagreement and mood extraction modules,
- translation into the languages of the SENSEI project: (English, Italian and French).

We integrated the template-based summary in the three languages into the UNITN social media summarization demo. The availability of language tags, provided by Websays, allow us to automatically detect the language of incoming conversations and display the template accordingly. An example of the outcome of the template-based summary in Italian is depicted in Figure 6.
As evauation, we tested the supervised components generating fillers of the slots in the template-based summary: namely the agreement/disagreement extraction module (MAE is 0.32) and mood prediciton module (average RMSE on five moods is 0.18).

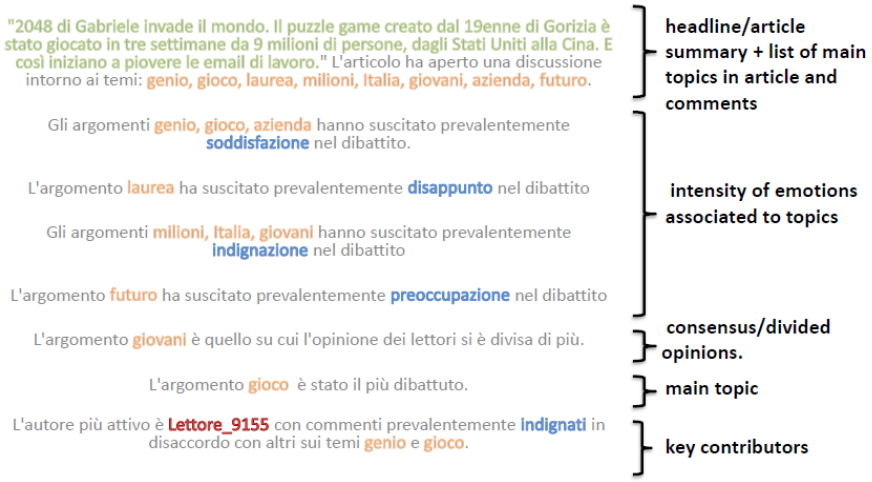

Figure 6: Example of the Template-based summary in Italian. The summary contains an extract of the article title (in green), topics (yellow), moods (blue), and the name of the most active blogger (red).

4 Conclusions

In this deliverable we have presented SENSEI's final conversation analysis and summarization components for both the speech and social media use cases. For the speech use case we outlined two abstractive template-based summarization systems which are based on generalizing human-authored synopses to templates with the slots to fill and then linking the summary units to the original conversations. For the social media use case we have reported several components to analyse news comments for sentiment, discourse relations between the comments, agreement and disagreement between a comment and its antecedent (either the source news article or the comment that it is a reply to) and anaphora resolution between different mentions of discourse entities. In addition to these components we described a graph-based approach to perform clustering of comments into topics and outlined two different approaches to label comment clusters. Finally, we presented three different summarization systems to summarize comments to online news.

While some components used in summarization are shared across the speech and social media scenarios, the core summarization systems are different. This has less to do with the modality of the conversational interaction (speech vs text) and more to do with differences in the use cases that underlie the speech and social media scenarios. In the speech scenario (call centres) users call with specific problems to be addressed. These conversations are two party and the problems addressed, in general, fall into a finite number of classes that recur again and again. As a consequence it is possible to induce from synopses templates whose slots reflect the semantics of the specific problem (e.g., in the Decoda domain, lost luggage, route enquiries, etc.). For social media, where conversations are multiway and can involve scores of participants, reader comments may address any issue or topic whatsoever, inducing topic-specific templates is impossible. Hence different approaches to the two tasks appear to be necessary and at this point it is not easy to see how the techniques for these very different tasks might converge. However, it would be interesting to explore potential transfer of approaches from one task setting to the other to see whether new insights might be gained.

Taken together the refinements and extensions described in this report have led to: improved performance on some tasks, new functionality and a more integrated and robust software infrastructure. Of course much remains to be done to improve performance further. To gain deeper insight into what is and is not working and how useful the technologies that have been developed are for end users further evaluation is needed, both intrinsic and extrinsic. This is on-going and will be reported in D1.4.

References

[1] Michael White, Tanya Korelsky, Claire Cardie, Vincent Ng, David Pierce, and Kiri Wagstaff. Multi-document summarization via information extraction. In *Proceedings of the first international conference on Human language technology research*, pages 1–7. Association for Computational Linguistics, 2001.

[2] Thierry Bazillon, Melanie Deplano, Frederic Bechet, Alexis Nasr, and Benoit Favre. Syntactic annotation of spontaneous speech: application to call-center conversation data. In *LREC*, pages 1338–1342, 2012.

[3] Carole Lailler, Anas Landeau, Frdric Bchet, Yannick Estve, and Paul Delglise. Enhancing the RATP-DECODA corpus with linguistic annotations for performing a large range of NLP tasks. In *LREC*, 2016.

[4] Benoit Favre, Dilek Hakkani-Tür, and Sebastien Cuendet. Icsiboost. `http://code.google.come/p/icsiboost`, 2007.

[5] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[6] Benoit Favre, Evgeny Stepanov, Jérémy Trione, Frédéric Béchet, and Giuseppe Riccardi. Call centre conversation summarization: A pilot task at multiling 2015. In *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 232, 2015.

[7] Jun-Ping Ng and Viktoria Abrecht. Better summarization evaluation with word embeddings for rouge. *arXiv preprint arXiv:1508.06034*, 2015.

[8] Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proc. of the 8th International Natural Language Generation Conference (INLG 2014)*, pages 45–53, 2014.

[9] Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

[10] Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. Multiwordnet: developing an aligned multilingual database. In *Proceedings of the first international conference on global WordNet*, volume 152, pages 55–63, 2002.

[11] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[13] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Machine Translation Summit X*, 2005.

[14] Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. Discourse segmentation of multi-party conversation. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 562–569. ACL, 2003.

[15] Marco Dinarelli, Silvia Quarteroni, Sara Tonelli, Alessandro Moschitti, and Giuseppe Riccardi. Annotating spoken dialogs: from speech segments to dialog acts and frame semantics. In *Proc. of EACL Workshop on the Semantic Representation of Spoken Language*, pages 34–41, Athens, Greece, 2009.

[16] Benoit Favre, Evgeny A. Stepanov, Jérémy Trione, Frédéric Béchet, and Giuseppe Riccardi. Call centre conversation summarization: A pilot task at MultiLing 2015. In *The 16th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 232–236, Prague, Czech Republic, September 2015. ACL.

[17] Jérémy Trione. Méthodes par extraction pour le résumé automatique de conversations parlées provenant de centres d'appels. In *16éme Rencontre des ètudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RECITAL)*, pages 104–111, 2014.

[18] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proc. of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336. ACM, 1998.

[19] Philipp Koehn. Statistical significance tests for machine translation evaluation. In *EMNLP*, pages 388–395. Citeseer, 2004.

[20] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. The Kaldi speech recognition toolkit. In *Proceedings of ASRU*. IEEE, 2011.

[21] C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, pages 171–185, 1995.

[22] Amita Misra and Marilyn Walker. Topic independent identification of agreement and disagreement in social media dialogue. In *Proceedings of the SIGDIAL 2013 Conference*, pages 41–50, Metz, France, August 2013. Association for Computational Linguistics.

[23] Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. A corpus for research on deliberation and debate. In *LREC*, pages 812–817, 2012.

[24] Jacob Andreas, Sara Rosenthal, and Kathleen McKeown. Annotating agreement and disagreement in threaded discussion. In *LREC*, pages 818–822. Citeseer, 2012.

[25] Lu Wang and Claire Cardie. Improving agreement and disagreement identification in online discussions with a socially-tuned sentiment lexicon. *ACL 2014*, page 97, 2014.

[26] Fabio Celli, Giuseppe Riccardi, and Arindam Ghosh. Corea: Italian news corpus with emotions and agreement. In *Proceedings of CLIC-it 2014*, pages 98–102, 2014.

[27] Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. The measurement of interrater agreement. *Statistical methods for rates and proportions*, 2:212–236, 1981.

[28] Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. The penn discourse treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, 2008.

[29] Evgeny A. Stepanov, Giuseppe Riccardi, and Ali Orkan Bayer. The UniTN discourse parser in CoNLL 2015 shared task: Token-level sequence labeling with argument-specific models. In *The SIGNLL Conference on Computational Natural Language Learning (CoNLL)- Shared Task*, pages 25–31, Beijing, China, July 2015. ACL.

[30] Massimo Poesio, Roland Stuckardt, and Yannick Versley, editors. *Anaphora Resolution: Algorithms, Resources and Applications*. Springer–Verlag, 2016.

[31] Tom Morton. Coreference for NLP applications. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2000.

[32] Josef Steinberger, Massimo Poesio, Mijail A. Kabadjov, and Karel Ježek. Two uses of anaphora resolution in summarization. *Information Processing and Management*, 2007. Special Issue on Text Summarisation.

[33] Defense Advanced Research Projects Agency. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, CA., 1995. Morgan Kaufmann.

[34] L. Hirschman. MUC-7 coreference task definition, version 3.0. In N. Chinchor, editor, *Proceedings of the 7th Message Understanding Conference*. NIST, 1998. Available online at http://www-nlpir.nist.gov/related_projects/muc/proceedings/muc_7_toc.html.

[35] Orphée De Clercq, Véronique Hoste, and Iris Hendrickx. Cross-domain dutch coreference resolution. In *Proceedings of Recent Advances in Natural Language Processing*, pages 186–193, Hissar, Bulgaria, 2011.

[36] Olga Uryupina and Massimo Poesio. Domain-specific vs. uniform modeling for coreference resolution. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC)*, Istanbul, Turkey, 2012.

[37] Mijail Kabadjov, Udo Kruschwitz, Massimo Poesio, Josef Steinberger, Marc Poch, and Hugo Zaragoza. The OnForumS corpus from the Shared Task on Online Forum Summarisation at MultiLing 2015. In *Proceedings of LREC*, Portoroz, Slovenia, 2016.

[38] K. Rodriguez, F. Delogu, Y. Versley, E. W. Stemle, and Massimo Poesio. Anaphoric annotation of wikipedia and blogs in the live memories corpus. In *Proceedings of LREC*, Floriana, Malta, 2010.

[39] Massimo Poesio and Ron Artstein. Anaphoric annotation in the arrau corpus. In *Proceedings of LREC*, Marrakesh, Morocco, 2008.

[40] H. Cunningham, V. Tablan, A. Roberts, and K. Bontcheva. Getting more out of biomedical documents with gate's full lifecycle open source text analytics. *PLoS Comput Biol*, 9(2), 2013.

[41] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. *Text Processing with GATE (Version 6)*. University of Sheffield, 2011.

[42] Diana Maynard, Gerhard Gossen, Marco Fisichella, and Adam Funk. Should I care about your opinion? detection of opinion interestingness and dynamics in social media. *Journal of Future Internet*, 2014.

[43] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[44] Duyu Tang, Furu Wei, Bing Qin, Ting Liu, and Ming Zhou. Coooolll: A deep learning system for twitter sentiment classification. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 208–212, 2014.

[45] Aliaksei Severyn and Alessandro Moschitti. Unitn: Training deep convolutional neural network for twitter sentiment classification. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Association for Computational Linguistics, Denver, Colorado*, pages 464–469, 2015.

[46] Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. Learning sense-specific word embeddings by exploiting bilingual resources. In *COLING*, pages 497–507, 2014.

[47] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv preprint arXiv:1504.06654*, 2015.

[48] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.

[49] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL (1)*, pages 1555–1565, 2014.

[50] Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210, 2005.

[51] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *SIGKDD*, pages 168–177, 2004.

[52] Saif M Mohammad and Peter D Turney. Nrc emotion lexicon. Technical report, NRC Technical Report, 2013.

[53] Stijn Marinus Van Dongen. Graph clustering by flow simulation. *Ph.D thesis.*

[54] Ahmet Aker, Emina Kurtic, Mark Hepple, Rob Gaizauskas, and Giuseppe Di Fabbrizio. Comment-to-article linking in the online news domain. In *Proceedings of MultiLing, SigDial 2015*, 2015.

[55] Ahmet Aker, Monica Lestari Paramita, Emma Barker, and Robert J Gaizauskas. Bootstrapping term extractors for multiple languages. In *LREC*, pages 483–489, 2014.

[56] G. Salton and M. Lesk, E. Computer evaluation of indexing and text processing. In *Journal of the ACM*, volume 15, pages 8–36, New York, NY, USA, 1968. ACM Press.

[57] Cheng-Ying Liu, Ming-Syan Chen, and Chi-Yao Tseng. Incrests: Towards real-time incremental short text summarization on comment streams from social network services. *Knowledge and Data Engineering, IEEE Transactions on*, 27(11):2986–3000, 2015.

[58] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[59] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL (1)*, pages 238–247, 2014.

[60] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486, 2009.

[61] David Jurgens and Ioannis Klapaftis. Semeval-2013 task 13: Word sense induction for graded and non-graded senses. In *Second joint conference on lexical and computational semantics (* SEM)*, volume 2, pages 290–299, 2013.

[62] Eyke Hüllermeier, Maria Rifqi, Sascha Henzgen, and Robin Senge. Comparing fuzzy partitions: A generalization of the rand index and related measures. *Fuzzy Systems, IEEE Transactions on*, 20(3):546–556, 2012.

[63] Clare Llewellyn, Claire Grover, and Jon Oberlander. Summarizing newspaper comments. In *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.

[64] Ugo Scaiella, Paolo Ferragina, Andrea Marino, and Massimiliano Ciaramita. Topical clustering of search results. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 223–232. ACM, 2012.

[65] Elham Khabiri, James Caverlee, and Chiao-Fang Hsu. Summarizing user-contributed comments. In *ICWSM*, 2011.

[66] Zongyang Ma, Aixin Sun, Quan Yuan, and Gao Cong. Topic-driven reader comments summarization. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 265–274. ACM, 2012.

[67] Nikolaos Aletras, Timothy Baldwin, Jey Han Lau, and Mark Stevenson. Representing topics labels for exploring digital libraries. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 239–248. IEEE Press, 2014.

[68] Jey Han Lau, Karl Grieser, David Newman, and Timothy Baldwin. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1536–1545. Association for Computational Linguistics, 2011.

[69] Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499. ACM, 2007.

[70] Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 465–474. ACM, 2013.

[71] Ahmet Aker, Emina Kurtic, Balamurali A R, Monica Paramita, Emma Barker, Mark Hepple, and Rob Gaizauskas. A graph-based approach to topic clustering for online comments to news. In *Proceedings of the 38th European Conference on Information Retrieval*, 2016.

[72] Annie Louis and Shay B Cohen. Conversation trees: A grammar model for topic structure in forums. In *Proceedings of EMNLP*, 2015.

[73] Shuo Chang, Peng Dai, Jilin Chen, and Ed H Chi. Got many labels?: Deriving topic labels from multiple sources for social media posts using crowdsourcing and ensemble learning. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 397–406. International World Wide Web Conferences Steering Committee, 2015.

[74] Shafiq Joty, Giuseppe Carenini, and Raymond T Ng. Topic segmentation and labeling in asynchronous conversations. *Journal of Artificial Intelligence Research*, pages 521–573, 2013.

[75] Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 465–474, New York, NY, USA, 2013. ACM.

[76] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data*. Springer, 2007.

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

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

[79] Hoa Trang Dang. Overview of duc 2005. In *Proceedings of the document understanding conference*, 2005.