EMBEDDIA

Cross-Lingual Embeddings for Less-Represented Languages in European News Media

Research and Innovation Action
Call: H2020-ICT-2018-1
Call topic: ICT-29-2018 A multilingual Next generation Internet
Project start: 1 January 2019 Project duration: 36 months

D3.5: Report generator from multilingual comments (T3.3)

Executive summary
Task T3.3 developed and implemented methods for generating human-readable reports from multilingual comments, from the outputs of the methods developed in Tasks T3.1 and T3.2. We took both extractive and template-based approaches. For the former, we developed methods that select sentences from the given comments that are representative of the identified topics and the main points of view expressed. For the second approach, we developed template-based natural language generation methods to generate human-readable reports of findings acquired with tools developed in other tasks. Finally, these approaches were integrated, together with other analytical models developed in Tasks T3.1 and T3.2, into a unified service that allows users to produce human readable analytical reports from any comment collection they desire.

Partner in charge: UH

Project co-funded by the European Commission within Horizon 2020 Dissemination Level		
PU	Public	PU
PP	Restricted to other programme participants (including the Commission Services)	–
RE	Restricted to a group specified by the Consortium (including the Commission Services)	–
CO	Confidential, only for members of the Consortium (including the Commission Services)	–

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825153

Deliverable Information

Document administrative information	
Project acronym:	**EMBEDDIA**
Project number:	**825153**
Deliverable number:	**D3.5**
Deliverable full title:	**Report generator from multilingual comments**
Deliverable short title:	**Report generator from multilingual comments**
Document identifier:	**EMBEDDIA-D35-ReportGeneratorFromMultilingualComments-T33-submitted**
Lead partner short name:	**UH**
Report version:	**submitted**
Report submission date:	**30/06/2021**
Dissemination level:	**PU**
Nature:	**R = Report**
Lead author(s):	**Hannu Toivonen (UH)**
Co-author(s):	**Leo Leppänen (UH), Aleš Žagar (UL), Marko Robnik-Šikonja (UL)**
Status:	**_ draft, _ final, x submitted**

The EMBEDDIA Consortium partner responsible for this deliverable has addressed all comments received. Changes to this document are detailed in the change log table below.

Change log

Date	Version number	Author/Editor	Summary of changes made
17/05/2021	v0.1	Hannu Toivonen (UH)	Started report.
01/06/2021	v0.2	Aleš Žagar (UL)	Added the unsupervised comment summarization.
03/06/2021	v0.3	Leo Leppänen (UH)	Completed draft.
03/06/2021	v1.0	Leo Leppänen (UH)	Ready for internal review.
10/06/2021	v1.1	Luis Adrián Cabrera Diego (ULR)	Internal review.
17/06/2021	v1.2	Senja Pollak (JSI)	Internal review.
21/06/2021	v1.3	All authors	Addressed comments from internal review.
21/06/2021	v2.0	Leo Leppänen (UH)	Ready for quality control.
27/06/2021	pre-final	Nada Lavrač (JSI)	Quality control.
29/06/2021	final	Leo Leppänen (UH)	Modifications based on quality control.
30/06/2020	submitted	Tina Anžič (JSI)	Report submitted

Table of Contents

1 Introduction

The EMBEDDIA project aims to develop monolingual and cross-lingual technology for news media industry. The overall objective of WP3, named *Cross-lingual Technologies for User-Generated Content*, is to apply EMBEDDIA's cross-lingual technologies to understand the reactions of multilingual news audiences, thus helping news media companies to better serve their audience, and acting as a basis to assure fairness and integrity of participants in public internet spaces.

This deliverable D3.5 is the sole report of Task T3.3. In this task, the objective was to develop techniques for report generation from multilingual comments, using the outputs of the methods developed for comment analysis in Tasks T3.1 (topic modelling) and T3.2 (comment filtering). Additionally, the work in this task draws from WP4 (text summarisation) and WP5 (natural language generation).

To accomplish the task of producing reports on comments, we developed a report generation service that combines both extractive and template-based approaches to text generation. For the former, we developed comment summarization methods that select sentences from the given comments that are representative of the identified topics and the main points of view expressed (Section 3). For the second approach, we developed template-based natural language generation methods to generate human-readable reports of findings acquired with tools developed in other tasks (Section 4).

The report concludes with conclusions, a list of associated outputs, and the related paper and API description included in appendices.

2 Producing Reports from Reader Comments

The overall research problem being tackled within this task is one of 'Text-to-Text' Natural Language Generation (NLG): we are tasked with producing a textual document, and the fundamental inputs of our system are textual comments written by readers of online news articles.

Instead of formulating the problem directly as a text-to-text generation task, however, we split the problem into two conceptually distinct steps: comment analysis and report generation. This decision facilitates the integration of various comment analysis components that are being developed in the EMBEDDIA project (see Figure 1). These have as output either other texts (comment summarization component) or structured data (topic modeling and comment filtering components). Both types of outputs can then be used in the natural language generation (NLG) phase of report generation.

The decision to separate the summarization and the NLG components is supported by an analysis of the state-of-the-art in natural language generation. As indicated by the survey of Gatt and Krahmer (2018), academic works on text-to-text generation tasks, such as summarization, simplification, machine translation or question generation tend to be focused on these individual tasks, and use methods very distinct from those used in the context of data-to-text NLG.

While the data flow of Figure 1 solves the problems associated with a single, unified and joint model, as described above, the same model is problematic from the perspective of control. First, in the model shown in Figure 1 the various analytical models would all need to be separately aware of the Report Generator component, as the interaction would be driven by the analytical models. Second, the input reader comments need to be sent to the various analytical tools separately, and the results to then be collated separately in the reporter. This requires the user to be aware of all the analytical tools, and separately ask them to analyze the comments and then 'push' the results forwards to the NLG component.

These problems can be alleviated by introducing a front-facing control component that provides the system's user with a view that presents the whole system as a single, unified construct. As such, the users do not need to know what analytical models were available to the system, or that the system is constructed in a modular manner in the first place. In addition, the control component allows for the analytical models to be unaware of the NLG generator.

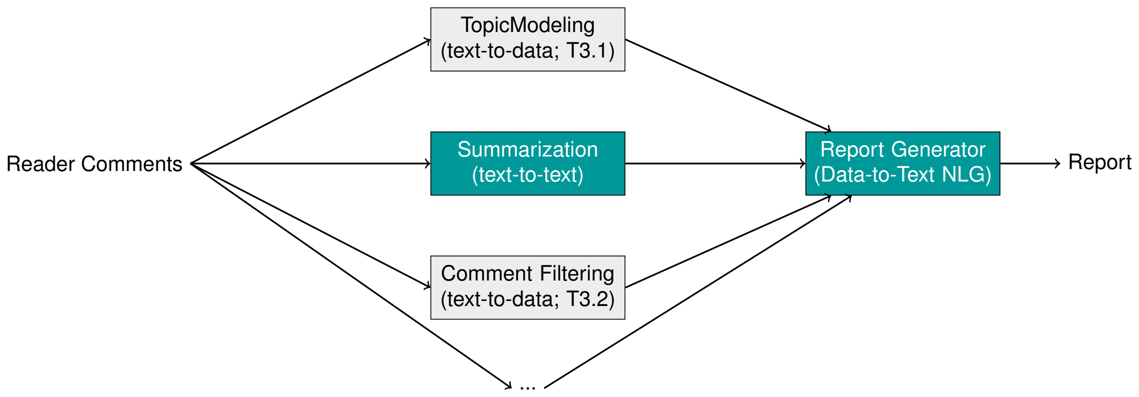

Figure 1: Division of the larger Report Generation task into distinct comment analysis tasks followed by a data-to-text NLG task. (Topic modeling and comment filtering are not part of this deliverable, but are described in Deliverable D3.4 and D3.6 (forthcoming), respectively.)

For practical reasons, we combine the NLG component and the front-facing API and control component into one single software unit and hide the analytical models from the users perspective (Figure 2). This results in a general software architecture that is reminiscent of the 'microservices' architecture commonly used to organize complex software ecosystems, allowing modular addition of new analysis models.

This architecture shows how the textual reader comments are provided as input to the report generator (NLG component), which then queries the various comment analysis models to obtain the structured data from which the reports are generated. To facilitate this type of communication, each of the comment analysis components, as well as the NLG component, is 'wrapped' in a simple web server, providing a JSON API through which the models can be used. Following industry best practices, these microservices are provided as Docker containers.

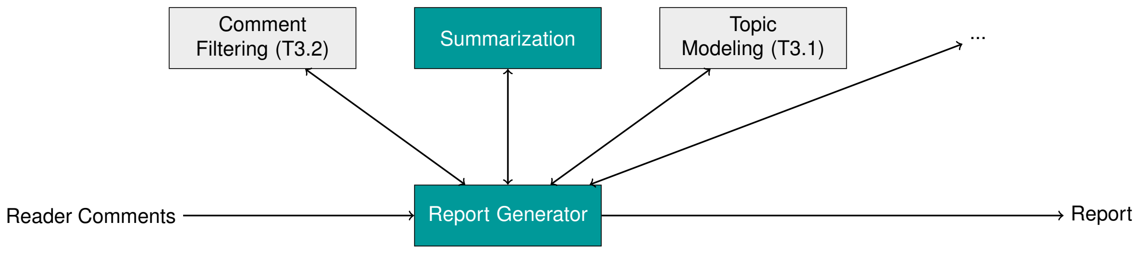

Figure 2: The microservice-like architecture of the report generation software. (Topic modeling and comment filtering are not part of this deliverable.)

In the next sections we will describe the components of the architecture: the comment summarization and analysis methods (Section 3) and the report generator component (Section 4).

The analysis methods are described elsewhere; see Deliverable D3.4 for context and opinion analysis, including topic modelling, and Deliverable D3.6 (forthcoming) or Pelicon, Shekhar, Škrlj, Purver, and Pollak (2021) for comment filtering. Thanks to the flexible architecture, new analysis modules can be added with relative ease. See Section 5 for more information.

3 Comment Summarization and Analysis

The comment analysis capabilities of the report generator include summarization, topic modeling and comment filtering developed in EMBEDDIA. In addition, we have incorporated a third-party sentiment analysis model by Barbieri, Camacho-Collados, Espinosa Anke, and Neves (2020) as a demonstration how non-EMBEDDIA models can be used to extend the analytical capabilities of the report generator. The way these analytical models are concretely integrated into the natural language generation specifically, and to the larger software ecosystem in general, will be discussed in Sections 4.1 and 5 respectively.

3.1 Comment Summarisation (Text-to-Text)

In this section, we give an overview of the comment summarization component. This is an instance of text-to-text generation within NLG literature (Gatt & Krahmer, 2018), taking as input comments written by readers of news.

In order to produce summaries of the textual contents of reader comments, we developed an unsupervised approach to summarization[1], which uses a modern multilingual representation of sentences together with standard extractive summarization techniques. Our method can be divided into three steps, illustrated in Figure 3; see Appendix A for the full original publication.

In the first step, neural sentence encoders represent the text in the form of numeric vectors. We used three competitive multilingual transformer-based sentence encoders: SBERT (Reimers & Gurevych, 2019), LaBSE (Feng, Yang, Cer, Arivazhagan, & Wang, 2020), and CMLM (Yang, Yang, Cer, Law, & Darve, 2020).

Dimensionality reduction is optional and represents the second step. The sentence vectors are mapped to a two-dimensional space with dimensionality reduction techniques (we use PCA or UMAP) to reduce noise or to visually explore the results.

In the third step, we select the most representative sentences to be returned as summaries. To achieve this, we use two groups of approaches: clustering-based (k-means and Gaussian mixture clustering) and a graph-based TextRank approach (Mihalcea & Tarau, 2004). Clustering approaches group similar sentence vectors and select the representative sentences based on the proximity to the centroid vector. Graph-based methods construct a graph based on the similarity of sentence vectors and then use graph node rankings to rank the sentences. The best-ranked sentences are returned as the summary.

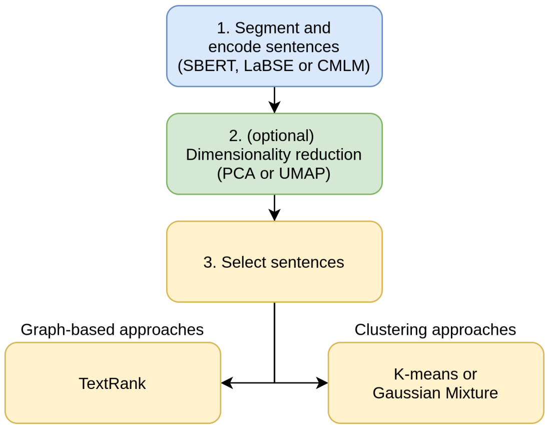
1. Segment and encode sentences (SBERT, LaBSE or CMLM)
2. (optional) Dimensionality reduction (PCA or UMAP)
3. Select sentences
Graph-based approaches
TextRank
Clustering approaches
K-means or Gaussian Mixture

Figure 3: Comment summarization data flow.

To test our approach, we created two new datasets. The news summarization dataset (CroNews) was created from the proprietary corpus[2] of approximately 1.8 million news articles from the popular Croatian 24sata news portal[3]. The second dataset is the comment summarization dataset (CroComments)

[1] https://github.com/EMBEDDIA/xl-user-comments
[2] https://www.clarin.si/repository/xmlui/handle/11356/1410
[3] https://www.24sata.hr/

containing reader comments of 42 articles from the Croatian Večernji list website[4] together with their short human-written abstractive summaries. We focused on the Croatian language, but extended the experiments to English (New York Times Comments[5] consists of 2 million comments and 9k articles) and German (the Austrian daily broadsheet newspaper DER STANDARD (Schabus & Skowron, 2018) is comprised of 1 million comments and 12k articles).

The best performing experimental setup uses the LaBSE sentence encoder, no scaling, and the TextRank algorithm for sentence selection. In our experiments (Table 1), LaBSE achieved on average 0.6 more ROUGE-L (Lin, 2004) points than SBERT and CMLM, which are indistinguishable in terms of performance. UMAP scaling preserved information better than PCA for 0.3 points but achieved 0.4 points less compared to no scaling. TextRank ranking method is superior to clustering for more than 2 points. The results of both datasets are very similar if we rank the models, with the best models being identical. TextRank with CMLM or LaBSE encoder is superior to clustering. Surprisingly, SBERT shows significantly lower performance with both clustering and ranking (with ranking worse than clustering).

Table 1: Summary of comment summarization results on the CroNews dataset. One row depicts the mean, standard deviation, minimum, maximum, and 95% confidence interval of ROUGE-L scores. The results are grouped by sentence encoder, scaling, and type of summarizer. Size corresponds to the number of experimental setups within a group.

Group	Mean	Std	Min	Max	95%CI	Size
Encoder						
LaBSE	36.11	1.47	34.30	39.01	(34.98, 37.25)	9
SBERT	35.49	0.75	34.71	36.99	(34.91, 36.06)	9
CMLM	35.32	1.91	32.58	37.99	(33.86, 36.79)	9
Scaling						
None	35.96	2.01	32.58	39.01	(34.42, 37.50)	9
UMAP	35.63	0.66	34.84	37.06	(35.12, 36.14)	9
PCA	35.33	1.44	34.12	37.99	(34.22, 36.43)	9
Summarizer						
TextRank	37.03	1.18	34.84	39.01	(36.13, 37.93)	9
Clustering	34.94	1.00	32.58	37.04	(34.45, 35.44)	18

We identified a few reasons that explain the lower scores of comment summarization compared to news summarization. The sentence encoders face a more challenging task of encoding the informal language. For the same reason, the accuracy of a sentence tokenizer is also significantly lower. A single CroComment document (containing all comments related to one news article) is usually comprised of texts by several authors, of variable length, and written in different styles. The average length of a document is 19.81 sentences with the standard deviation of 13.16 in comparison to CroNews dataset which contains 7.85 sentences with the standard deviation of 1.42.

Our comparison of different sentence representation approaches coupled with different summarization approaches showed that the most successful combinations are the same in news and comment summarization. We focused our further work on visualization of summaries[6] as it showed promising results, especially in the interactive exploration mode.

The work presented in Section 3.1 is described in full by Žagar and Robnik-Šikonja (2021), attached as Appendix A. An excerpt of the model's output, as received by the NLG component, can be seen in Figure 4. Figure 13 at the end of Section 4 shows how the output is expressed as part of the natural language output of the complete report generation system.

[4]https://www.vecernji.hr/
[5]https://www.kaggle.com/aashita/nyt-comments
[6]https://colab.research.google.com/drive/12wUDg64k4oK24rNSd4DRZL9xywNMiPil?usp=sharing

```
{
  "summary": [
    "Steta mladog zivota, steta svakog zivota naprasno izgubljenog...",
    ...
  ]
}
```

Figure 4: Example output of the summarization service. The output contains a list of sentences (length determined by the caller) that best summarize the input sentences. The output has been significantly abridged for the purposes of this deliverable.

3.2 Comment Analysis (Text-to-Data)

The report generation uses comment filtering, topic modeling and sentiment analysis models in order to obtain an analytical viewpoint to the comments. We give brief descriptions and example outputs of those analytical components below; for more details we refer to the relevant deliverables and articles (see below). Figure 13 at the end of Section 4 shows how similar outputs are eventually expressed in the final natural language output of the complete report generation system.

The comment filtering model – described in Deliverable D3.6 (forthcoming) and Pelicon et al. (2021) – ingests a list of comments, and returns for each comment a label describing whether it contains blockable content (such as hate speech), as well as the model's confidence in the label. An example of the model's JSON API's output is shown in Figure 5.

```
{
  "labels": [
    "Non-Blocked",
    "Non-Blocked",
    ...
  ],
  "confidences": [
    0.9772846698760986,
    0.9974973797798157,
    ...
  ]
}
```

Figure 5: Example output of the comment filtering service. Each input comment is associated with a label indicating whether it contains some type of blockable content and a confidence value. The output has been abridged for this Deliverable, as indicated by the ellipses.

The topic model ingests a list of comments, and returns a description of what the top-5 topics associated with each comment are. The output includes predefined natural language labels for the topics, short topic descriptions, as well as lists of words that are strongly associated with the topics. The topic modeling approach is described in Deliverable D3.4. An example output of the topic model's API is shown in Figure 6.

We also incorporated a multilingual sentiment analysis model developed by Barbieri et al. (2020). As above, this service takes as input the comments as a list of strings, and returns for each comment a sentiment value in the range from -1 (indicating extremely negative sentiment) to 0 (indicating a netural sentiment) to 1 (indicating an extremely positive sentiment). An example of the model's API's output is shown in Figure 7.

```
{
  "suggested_label": [
    "['topic_top0 : Food and fuel prices', 'topic_top1 : Have
        inflections1', 'topic_top2 : Solving problems', 'topic_top3 :
        Have inflections1', 'topic_top4 : Legal procedure']",
    ...
  ],
  "description": [
    "['topic_top0 : water, garbage, fuel, meat, milk - seems like a
        combo of \"pollution\" and \"food and fuel prices\"', '
        topic_top1 : \"have\"  inflections', ...]",
    ...
  ],
  "topic_words": [
    "['topic_top0 : vode, ...', 'topic_top1 : ima, ...', ...]",
    ...
  ]
}
```

Figure 6: Example output of the topic modeling service. The output includes a list of 5 suggested topic labels for each comment, shown here for one comment only. The labels are associated with both short descriptions and word lists. The output has been abridged for this Deliverable, as indicated by the ellipses.

```
{
  "sentiments": [
    -0.9431635737419128,
    -0.9279079437255859,
    ...
  ]
}
```

Figure 7: Example output of the sentiment analysis service. Each input comment is associated with a numeric sentiment value, ranging from -1 (highly negative) to 1 (highly positive). The output has been abridged for this Deliverable, as indicated by the ellipses.

4 Natural Language Generation

In this section, we give an overview of the text generation component. Section 5 then describes how reports are generated as a whole, including both generated text and extractive summaries.

As noted in Section 2, having decoupled the comment analysis from the text generation, we can view the natural language generation task associated with producing the reports as one of 'data-to-text' NLG. In this section, we describe our approach for solving this problem. Importantly, we observe that the language generation task is not meaningfully different from the news generation task conducted in Work Package WP5, which is concerned with producing more traditional news texts for both journalists and general audiences. That is, from the perspective of the NLG component being developed here, the texts being produced are 'news-like' in the sense of being descriptions of some factual phenomena targeted at a specific audience. The distinction to Work Package 5 is solely in the different source of the factual data underlying the text being produced, as well as the audience. Phrased differently, we are producing 'meta news' about news comments.

As such, the requirements imposed by this problem description on the used technologies closely mirror those identified within Deliverable D5.2, describing our approach to news automation. Most notably, once a certain basic level of fluency in the output has been reached, it is far more important that the output be correct and trustworthy than it is for it to be highly fluent prose. Furthermore, the purpose of this Task T3.3 is to produce natural language texts of a type for which we have no example data nor example texts. As such, NLG approaches based on machine learning – insofar as the NLG task is concerned – are not suitable for our purposes. For a more detailed analysis of how these types of system requirements interact with the state-of-the-art in NLG, we direct the reader to Deliverables D5.2, describing several NLG case studies in automated journalism, and D2.4, describing a general NLG architecture for factual domains.

We apply the architecture developed in Task T2.3 and described in Deliverable D2.4, as well as some of the techniques developed in Work Package WP5, to this generation task. To provisionally validate this approach, we manually crafted a hypothetical report of the type we expected the reporter to be able to produce given the – at the time ongoing – research efforts into the various comment analysis models. Using this document as a target, we decided to base the language generation component on a simplified version of the COVID-19 case study system developed in Task T5.1 as it was the simplest of the case study systems, but still seemed to offer a good starting place for developing the report generation system.

In the next subsections, we will describe the NLG 'pipeline' of the report generation system. We assume that the readers are already familiar with the public Deliverable D5.2, describing the COVID-19 case study application, and as such the description of the NLG component of the report generation system below focuses on differences to the systems described in Deliverable D5.2. The general architecture of the NLG system is shown in Figure 8.

The system input is a collection of reader comments. These comments are provided in the JSON format through the API and Control component, which is a very simple HTTP server providing an API. The API and Control component conducts some trivial input verification (principally checking that the input collection of comments is not empty), and then passes the comments as-is to the Message Generation component.

4.1 Message Generation

The Message Generation component is modular in nature and passes the comments to a selection of Paragraph Resources. Specifically, each Paragraph Resource contains a Message Parser component. This Message Parser component takes the input comments, and constructs from them atomic units of information, called *messages*. Each of these messages describes a single piece of information about the input data. To construct the messages, the Message Parsers consult the various Text Analysis Microservices via their microservice JSON APIs (Sections 2 and 5). The generated messages correspond to individual sentences in the final textual output of the system.

Conceptually, each Message Parser is responsible for producing the messages associated with a single paragraph of the output text. As such, the Message Parsers can consult any number of Text Analysis Microservices deemed necessary. This highlights a key difference to the systems built within Task T5.1: while the news automation systems built by UH-CS previously have employed relatively complex document planning methods and estimates of newsworthiness, in this case the structure and the information content of the comment reports is largely predefined.

A further complication to this process is presented by the fact that while most Text Analysis Microservices are multilingual, the topic modeling service employs a Croatian-only topic model. As such, the message generation also accepts an optional parameter describing the language of the comments. If present, this language identified allows the system to also include messages that necessitate language-specific analyses, such as the topic modeling mentioned previously. If the language field is omitted, messages are generated only from those models that are multilingual. This aspect is discussed further in Section 5.

While the messages described in the case studies of Deliverable D5.2 were relatively complex with

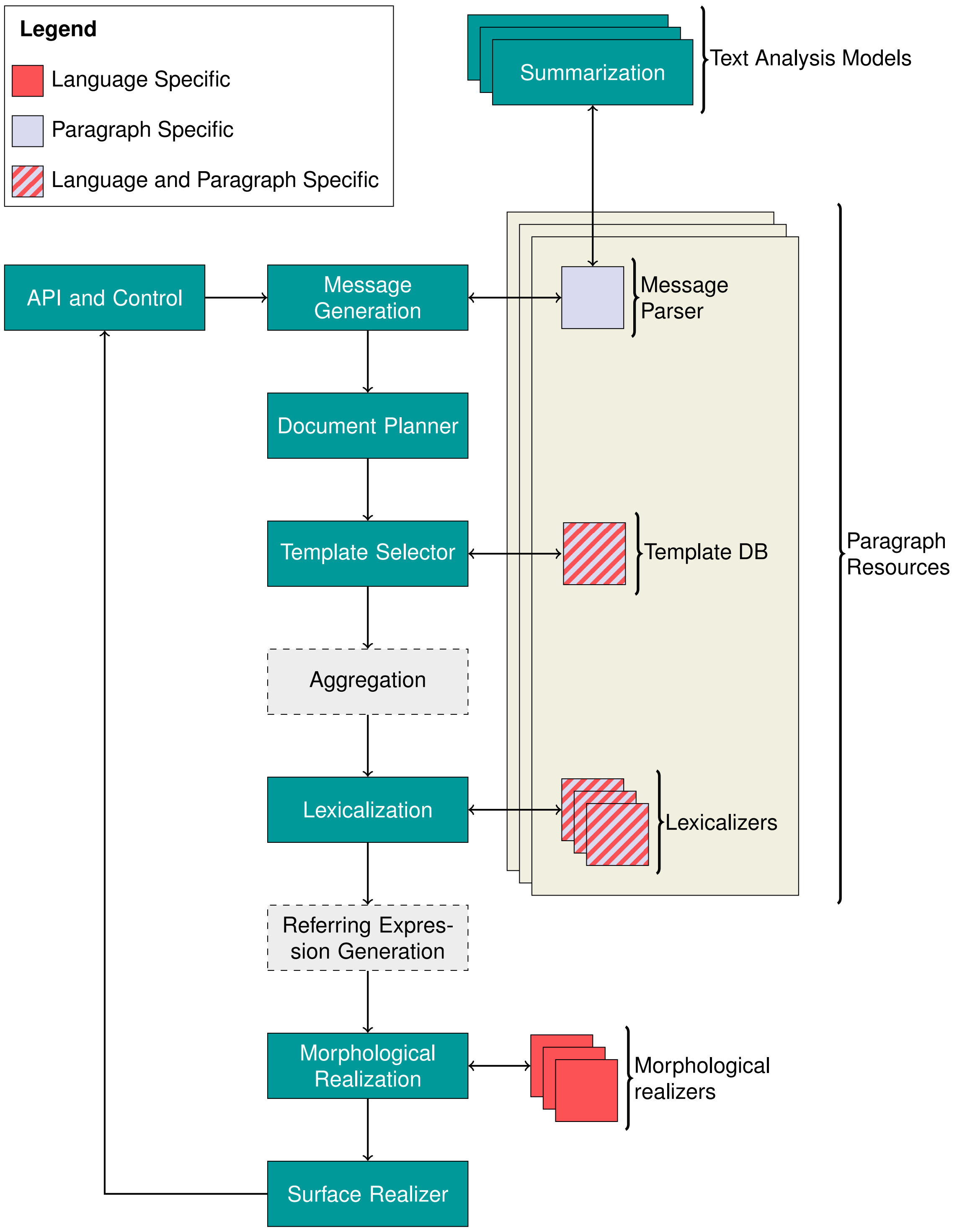

Figure 8: The architecture of the NLG component. The middle column indicates the primary NLG pipeline. The square boxes' colors in the right-most column are detailed in the Legend. The green 'Text Analysis Models' boxes represent the comment analysis microservices, described above. Boxes with dashed outlines in the middle and right column indicate components supported by the architecture, but not present in the current version of the system.

metadata fields such as `location` and `timestamp`, in this case we employ a significantly simpler Message data structure. As with the COVID-19 case study described in Deliverable D5.2, the fields themselves are stored in an immutable data structure called 'fact' inside the message, while the outer, mutable, 'message' data structure allows for additional information, such templates, as discussed below, to be associated with the facts. Technically, a single message can convey multiple facts, but this property is not used within the scope of the work described in this deliverable.

For the purposes of the system described here, it is sufficient that each message contain a `value`, i.e. the (usually numerical) information of the message, as well as a `value_type`, describing the *interpretation* of this numerical information. We encode the semantic information in the `value_type` fields using a hierarchy of colon-separated labels, such as `comment_filtering:blocked:abs` to indicate that the corresponding `value` is the absolute number of comments filtered.

The first label of the `value_type` indicates what paragraph the message is associated with: above, the message belongs to the `comment_filtering` paragraph, which describes the general prevalence of filtered comments as a whole. Other sentences related to comment filtering might still reside in other paragraphs (for example a paragraph describing a subset of comments that discuss some topic identified by topic modeling, might include a sentence about the prevalence of comments filtered among said topic), but they would have a different prefix label.

As we reuse code from the case studies from Task T5.1, we co-opt the `newsworthiness` field for purposes related to document planning. Namely, we use the 'newsworthiness' fields of the message data structures to encode the intended ordering of the messages: each message – again, conceptually a sentence in the final text – is given a importance value of $p \times 100 + s$. Here p is an index over the paragraphs, indexed with the final paragraph having index 1, the second-to-last paragraph having index 2, and so on. On the other hand, s is the index of the sentence within the paragraph, again indexed from the end.

As in the case of the news automation systems described in Deliverable D5.2, the output of this stage is an unordered collection of the Message data structures.

4.2 Document Planning

As the documents have a pre-set structure, the Document Planning process is significantly simplified from the one described in Deliverables D5.2 and D5.3. The document plan tree is generated iteratively, one message and paragraph at a time. To begin a new paragraph, the planner selects the most important message that has not yet been added to the paragraph. Since the importance values are set using the scheme described above, this is a known sentence. Following this, all the messages with the same prefix label (indicating that they belong in the same paragraph) are added to the paragraph in the order of their importance values.

This simple process results in a tree-structured document plan, where messages are collated into paragraphs as per their `value_type` fields and ordered within the paragraphs by their importance values. Notably, the order of the paragraphs is *also* determined by the importance fields: by setting the message importance values as described in Section 4.1, the reverse paragraph index p provides this ordering.

The Document Planning process results in a tree-structured document plan, as shown in Figure 9 which is then provided as input to the Template Selector.

4.3 Template Selection

At the start of the template selection process, the document plan (and the messages in it) are abstract and language-agnostic. To begin the process of transforming the message into natural language, we

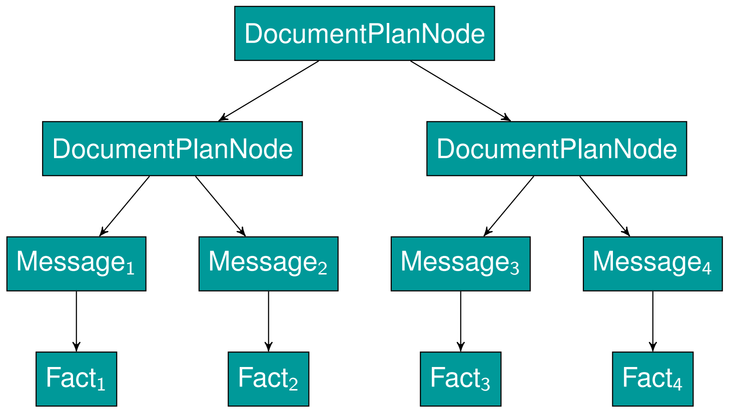

Figure 9: The structure of a hypothetical document plan. The root Document Plan Node corresponds to the whole document, while the mid-level Document Plan Nodes corresponds to paragraphs. The Messages correspond to individual sentences, with Facts being immutable holders of the Messages' most important information.

associate each message with a 'template'. The templates are defined in Paragraph Resource specific template databases using our custom templating language, demonstrated in Figure 10.

```
en:  Of the analyzed comments, a total of {value} were identified as ↩
     containing blockable content.
| value_type = hate_speech:blocked:abs, value > 0

en:  Of the analyzed comments, none were identified as containing blockable ↩
     content.
| value_type = hate_speech:blocked:abs, value = 0
```

Figure 10: Excerpt from a template database. The linebreak identified by the symbol ↩ has been introduced for this Deliverable and is not present in the original template.

Each template consists, fundamentally, of three components: a language identifier, a template string, and a template rule. The template string defines the template as a sequence of literals (usually words) and 'slots' (indicated by curly brackets {}) that are replaced with information from the messages during Lexicalization, as described below. The template strings are defined in groups, wherein each groups contains one or more rules that describe when the template strings in said cluster are applicable. Usually, these rules simply define that the template is able to express any message with a specific `value_type`, but the rules can also inspect e.g. the value of the message, as shown in Figure 10. This latter feature allows us to provide different templates, for example, to messages that have zero, negative and positive values. Both slots and literals can be associated with metadata, indicating e.g. that the realization of a slot should be the absolute value of the slot's value, or that the slot value should be adjusted to be in a specific morphological form, such as a possessive.

Notably, as the rules apply to template groups, and each template is associated with a language identifier, templates for multiple languages can be defined in one group, thus enabling for reuse of the rules between the various languages. As such, while the present version of the Report Generator only contains templates in English, the system can be extended to provide multilingual generation in the same way the case study systems described in Deliverable D5.2 are multilingual.

The Template Selector inspects the document plan, as provided by the Document Planner component above, and for each message in the document plan identifies a template that can be used to express said message. This is done by first limiting the templates to those of the correct language, and then checking which templates have rules that match the message. If there are multiple suitable templates, one is chosen pseudorandomly, i.e. so that the random selection is the same every time the exact same document is generated. This random selection of the templates allows for the system to be extended in the future to include multiple variations of each template to increase the variety of the language produced by the system.

The output of the template selection process is a version of the document plan, where each message is associated with a template consisting of literals and references to fields of the message. An example of such a data structure is shown in Figure 11.

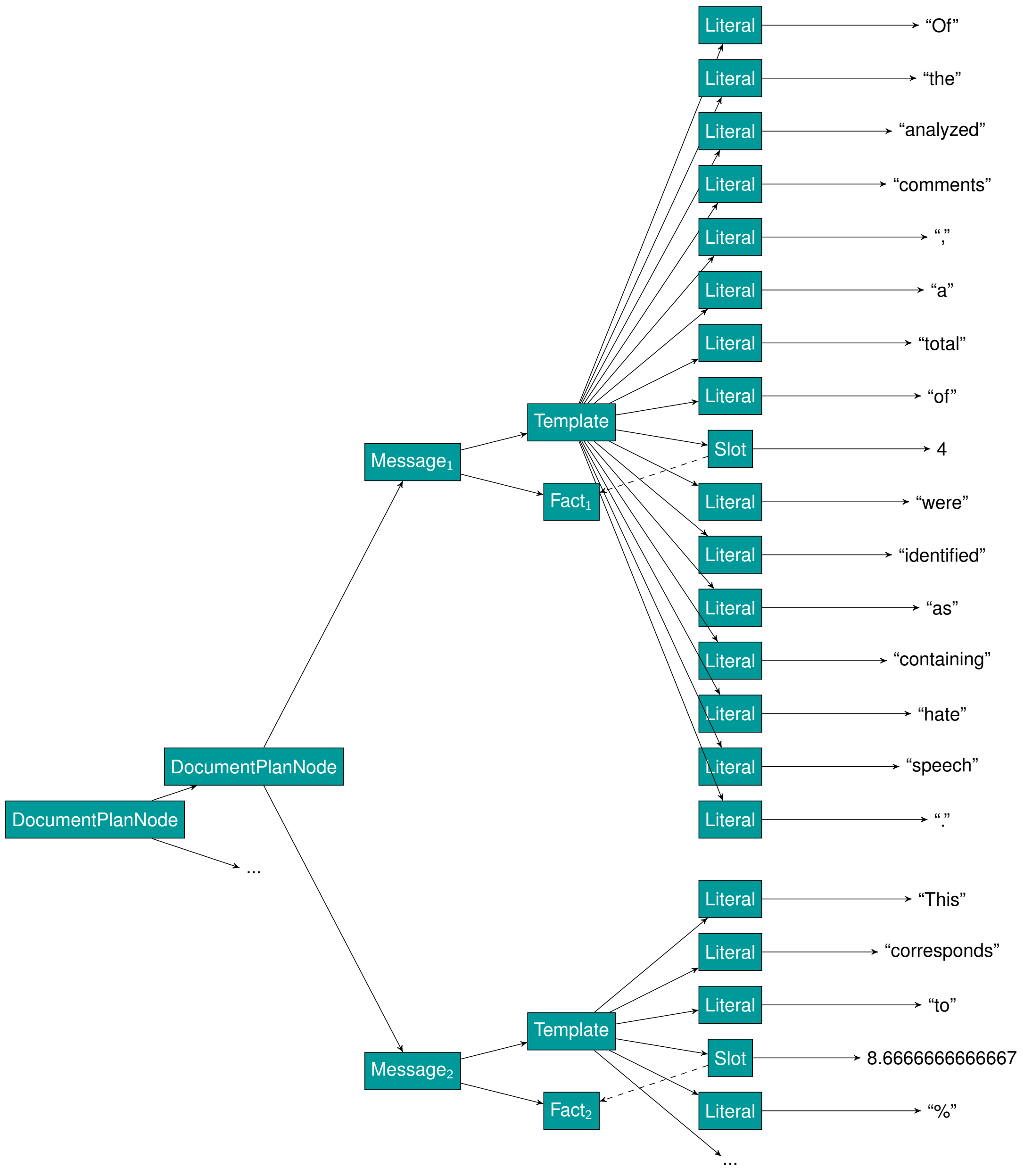

Figure 11: An example of a document plan following the attachment of templates. Templates consist mostly of 'literals', which are strings that will be included in the final output as-is. In addition, templates contain 'slots' which present information defined in on the fields of the associated message's fact, as indicated by the dashed arrows.

4.4 Lexicalization

The document plan, with the attached templates, is provided next to a lexicalization component. This lexicalization component proceeds to determine how to the references to the various fields of the messages (i.e. the non-literal components of the templates) should be realized, i.e. what words ought to be used to express them.

As a concrete example of the kinds of considerations that need to be made at this stage, the lexicalization component needs to decide at what level of rounding various numbers included in the text should be shown. As the same algorithm should be suitable for numbers of many different magnitudes, we need to consider the magnitude of the number in deciding the suitable rounding. For values greater than equal to one, the rounding is conducted by dropping the decimals. For numbers that are strictly less than 1, we identify the first non-zero decimal digit and round to a level where both that digit and the next decimal are shown. In other words, the number 0.0012345 is rounded to 0.0012. In any case, the number is rounded to at most 4 decimal places. After rounding, any trailing zero digits are dropped, except for the first decimal digit. This results in the number 0.0101 being rounded to 0.01, and the value 0.000001 being rounded to 0.0. In addition, the process needs to select the suitable separators to use for thousands and and decimals based on the output language.

As we re-appropriate code from the case study systems described in Deliverable D5.2, the system allows for the Paragraph Resources to be associated with lexicalization components (labeled 'Lexicalizers' in Figure 8), that would provide additional instructions on how to lexicalize more complex values, such as lists of values, for specific messages.

The output of the lexicalization process is a modified version of the Document Plan, as shown in Figure 12.

4.5 Morphological Realization

While lexicalization ensures that all the words that make up the final textual output of the system are present in the document plan, it is possible that some of the words still need to be inflected into suitable morphological words. This is more common in some non-English languages, such as Finnish.

During morphological realization, any words in the templates associated with morphological information are realized into the relevant morphological form. For example, if a hypothetical slot's value was a string, and the slot was associated with a label indicating the contents should be transformed into the possessive form, the morphological realizer component would consult a language-specific component to conduct the transformation.

As the present version of the system conducts generation only in English, we have not found need to conduct such morphological transformations as of yet. However, as a consequence of reusing code from the systems described in Deliverable D5.2, the system contains fully functional morphological realization components, using the UralicNLP library (Hämäläinen, 2019) for both English and Finnish.

4.6 Surface Realization

The final component of the main NLG pipeline, the Surface Realizer component, takes the final version of the document plan, expressed as a tree such as that shown in Figure 12, and transforms it to linear text. This is done by conducting a depth-first traversing all the leaf-nodes of the document.

As part of this transformation, HTML tags are inserted to indicate the division into paragraphs. Furthermore, minor tweaks are made to the whitespace of the text: as a consequence of a limitation of the templating language, some sentence-final periods need to be typed in the templates with a preceding space, which needs to be removed at this stage.

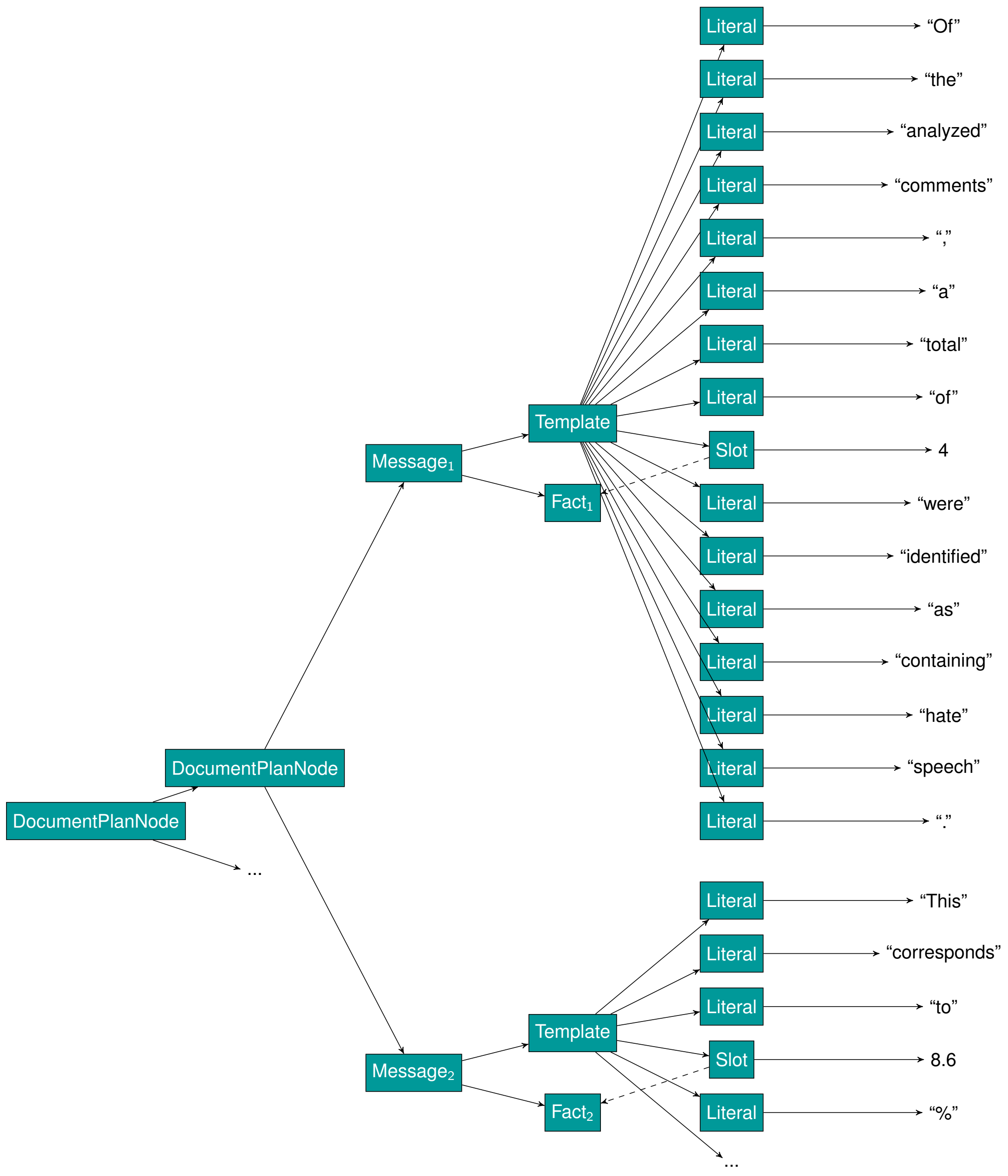

Figure 12: An excerpt from a document plan following lexicalization. Note that, in this case, the lexicalization changes are small, limited to changing the representation of the decimal value in one of the slots to a more sensible format.

The output of this stage is the provided to the Control and API component to be returned to the user of the system. An example of the resulting text is shown in Figure 13.

Output	Models used
This report describes an automated analysis conducted on 19 comments. The report has been generated automatically, in parts using machine learning methods, and no guarantee is made with regard to the accuracy of its contents.	-
Of the analyzed comments, a total of 1 were identified as containing blockable content. This corresponds to 5.3 % of all comments. For example, the following comment was identified as containing blockable content: “Na himnu im zvižde, koja kultura kod ovih incestuoznih i*iota.. Sramota!”	Comment filtering
In general, the analyzed comments are best summarized as follows: “Ideemo svabe razvalite engelzice. Ajmo Deutschland,razbite arogantne engleze. Ajmo svabe, spustite bahate englezice na zemlju. Ne znas koji su gori i bahatiji. Ajmo englezi rasturite te ss trupe!!”	Summarization
The mean sentiment of the comments was -0.47 (zero indicates a neutral sentiment). Of the analyzed comments, 15.8 % were positive (sentiment >= 0.25). 78.9 % of the comments were negative (sentiment <= -0.25). The most positive comment was “Ajmo Englezi, cijela Hrvatska uz englesku braću. Go England”	Sentiment analysis
The most common topic present in the comments was Football. It was featured prominently in 31.6 % of the comments. Comments discussing this topic are best summarized by this comment: “bolje njemačka, baš zato jer su englezi doma pa misle da bi oni trebali osvojiti..........pobjednik ovog dvoboja je vjerojatno u finalu ako nebude nekih većih iznenađenja”	Topic modeling & summarization
The second most common topic present in the comments was Croatian football. It was featured prominently in 21.1 % of the comments. Comments discussing this topic are best summarized by this comment: “ko englezi ne dodu do,finala tko zna kad ce vise ova godina bi mogla bit njihova”	Topic modeling & summarization

Figure 13: Example output of the Report Generation system generated from 19 different Croatian language comments to a 24sata news story. The report contains both generated text from the NLG component (all non-quoted text), a generated summary of all the comments (the first quoted section) and fully quoted comments (all quoted text other than the first quote). The formatting of the quotes has been introduced for this Deliverable: the report generator produces HTML output where the quotes are marked using the `blockquote` tags. The right-hand column indicates what analytical models (See Section 3) were involved in producing each paragraph.

5 Report Generation as a Composition of Microservices

As noted above, the Report Generator component is internally a collective of microservices. The system can be interacted with through a HTTP API both ingesting and serving JSON formatted data. The API is described in Appendix B and is technically a wrapper around the NLG component. This design of the report generator allows for it to be easily integrated into other systems, such as the EMBEDDIA Media Assistant Toolkit or an intranet of a news provider, as it appears as a single coherent unit despite the internal modular architecture.

Both the NLG component (Section 4) and the individual comment analysis components of the system (Section 3) are implemented as Dockerized web services. Docker is an industry-standard containerization service, somewhat analogous to application-specific virtual machines, that allows for easy setup of multiple dockerized services on a single logical host without needing to worry about the concerns such as incompatible software dependencies.

While Dockerization simplifies the process of setting up the services associated with the report generation, to properly function together, these services still need to be integrated on two levels.

First, as already briefly described in Section 2, the Report Generation is driven by the NLG component, which consults the various comment analysis services during generation. In order for this to be possible, the Paragraph Resources (cf. Section 4.1) need to contain the necessary logic to make HTTP calls to the relevant analysis services and to interpret and parse the results returned by them. For example, the Summarization model described in Section 3.1 is integrated into the system as one of these comment analysis microservices. A relevant Paragraph Resource[7] contains the required code to call the summarization API with a selection of comments, and parse from the response the summary. This summary is then turned into a Message, where the `value_type` is '`summary`' and `value` is the produced summary. This Paragraph Resource also contains a template database, with a template telling the system how to express said message.

If a new analysis service was to be integrated into the larger system, a new Paragraph Resource would need to be introduced into the NLG component, containing both the aforementioned logic, as well as any required templates and lexical resolvers needed to express the new information. Alternatively, if the component's analysis was to be included in any of the existing paragraphs, the equivalent modifications would need to be made into the existing Paragraph Resources. While this type of integration is non-trivial, it only needs to be done once.

Second, the NLG pipeline needs to be made aware of where (i.e. at what URLs) the APIs of the comment analysis services reside. This is done by way of a configuration file, containing the addresses of all relevant services, for example as shown in Figure 14. Generation prioritizes a language-specific resource if available and falls back to a multilingual resource if no suitable language-specific resource is found. In a case where a suitable multilingual model is not available (such as with the Croatian-only topic modeling), any textual content dependent on that type of a model are omitted from the resulting report. By allowing the integration of both multilingual and monolingual models, the system is easier to expand, and can facilitate both widely applicable multilingual comment analysis models, as well as highly specialized monolingual models.

As an additional upside, this enables the various components of the Report Generator to be distributed across multiple servers and service provides, where necessary. At the same time, this adds some complexity to the system, requiring this configuration step to be revised for each deployment of the Report Generation service. By distinguishing between this latter type of configuration-based integration from the programming-type integration, we give a large amount of leeway to those deploying the system without necessitating them to modify the code of the system.

[7]See `https://github.com/EMBEDDIA/EMBEDDIA-comment-reporter/blob/master/comment_reporter/resources/general_summary_resource.py` for the actual implementation of this Paragraph Resource.

```
[SUMMARIZATION]
all = http://localhost:8083/summarize

[TOPIC_MODEL]
hr = http://localhost:5001/comments_api/topic_model_list/
```

Figure 14: Excerpt from a configuration file describing where the various microservices are located. URLs labeled with 'all' indicate that the model is multilingual, while URLs labeled with a country code (e.g. 'hr') indicate language specific resources.

Currently, as described in Section 3, the report generation system as a whole includes four text analysis microservices providing access to the following comment analysis models: the multilingual summarization model described in Section 3.1; a monolingual Croatian topic model developed within Task T3.1; the multilingual comment filtering model developed within Task T3.2, and a multilingual sentiment analysis model by Barbieri et al. (2020).

6 Conclusion

In this deliverable, we described a system for generating reports from reader comments to news articles. The system is founded on two main contributions of Task T3.3: an extractive summarization method described in Section 3.1, and a modular, template-based NLG system described in Sections 4–5.

These contributions fulfill the goals of Task T3.3 well. The summarization component employs unsupervised learning and modern multilingual sentence representations and is thus highly multilingual. The NLG component, on the other hand, is built to be agnostic to the languages of the comments, as it offloads comment analysis work to the summarization model and other comment analysis models from other tasks T3.1 and T3.2. The combination of machine learning-based summarization and template-based text production aims to merge the best of both worlds. An evaluation of the methods will be carried out in Task 3.4.

The modular nature of the report generation system has several enticing properties. New comment analysis models can be added to the system by Dockerizing them and wrapping them in an HTTP service. Some additional programming effort is required, but this is not a significant challenge and such services are an industry standard. This effort is a worthwhile investment in order to allow future analysis models to be added.

Being based on the multilingual NLG architecture developed in Task T2.3, the NLG system (and thus the report generation system as a whole) can also be extended to produce the reports themselves in multiple languages, as demonstrated by the multilingual case study systems employing the same basic architecture in Deliverable D5.2.

7 Associated Outputs

Parts of this work are described in detail in the following publication:

Citation	Status	Appendix
Žagar, A., & Robnik-Šikonja, M. (2021, April). Unsupervised Approach to Multilingual User Comments Summarization. In *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation (pp. 89-98).*	Published	Appendix A

The work described in this deliverable has resulted in the following resources:

Description	URL	Availability
Code for the Summarization Microservice	`https://github.com/EMBEDDIA/EMBEDDIA-summarization-service`	Public (MIT)
Code for the sentiment analysis microservice	`https://github.com/EMBEDDIA/EMBEDDIA-sentiment-analysis-service`	Public (MIT)
Code for the NLG Service	`https://github.com/EMBEDDIA/EMBEDDIA-comment-reporter`	Public (MIT)
Visualization of User Comments	`https://github.com/EMBEDDIA/xl-user-comments`	Public (MIT)

References

Barbieri, F., Camacho-Collados, J., Espinosa Anke, L., & Neves, L. (2020, November). TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (pp. 1644–1650). Online: Association for Computational Linguistics. Retrieved from `https://www.aclweb.org/anthology/2020.findings-emnlp.148` doi: 10.18653/v1/2020.findings-emnlp.148

Feng, F., Yang, Y., Cer, D., Arivazhagan, N., & Wang, W. (2020). Language-agnostic BERT sentence embedding. *arXiv preprint arXiv:2007.01852*.

Gatt, A., & Krahmer, E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, *61*, 65–170.

Hämäläinen, M. (2019). UralicNLP: An NLP library for Uralic languages. *Journal of Open Source Software*, *4*(37), 1345. doi: 10.21105/joss.01345

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out* (pp. 74–81).

Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into text. In *Proceedings of the Empirical Methods in Natural Language Processing* (pp. 404–411).

Pelicon, A., Shekhar, R., Škrlj, B., Purver, M., & Pollak, S. (2021). Investigating cross-lingual training for offensive language detection. *PeerJ Computer Science*, *7*, e559. Retrieved from `https://dx.doi.org/10.7717/peerj-cs.559` doi: 10.7717/peerj-cs.559

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 3982–3992).

Schabus, D., & Skowron, M. (2018, May). Academic-industrial perspective on the development and deployment of a moderation system for a newspaper website. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)* (p. 1602-1605). Miyazaki, Japan. Retrieved from `http://www.lrec-conf.org/proceedings/lrec2018/summaries/8885.html`

Yang, Z., Yang, Y., Cer, D., Law, J., & Darve, E. (2020). Universal sentence representation learning with conditional masked language model. *arXiv preprint arXiv:2012.14388*.

Žagar, A., & Robnik-Šikonja, M. (2021). Unsupervised approach to multilingual user comments summarization. In *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation* (pp. 89–98).

Appendix A: Unsupervised Approach to Multilingual User Comments Summarization

Unsupervised Approach to Multilingual User Comments Summarization

Aleš Žagar, Marko Robnik-Šikonja
University of Ljubljana, Faculty of Computer and Information Science
Večna pot 113, 1000 Ljubljana, Slovenia
Ales.Zagar@fri.uni-lj.si Marko.Robnik@fri.uni-lj.si

Abstract

User commenting is a valuable feature of many news outlets, enabling them a contact with readers and enabling readers to express their opinion, provide different viewpoints, and even complementary information. Yet, large volumes of user comments are hard to filter, let alone read and extract relevant information. The research on the summarization of user comments is still in its infancy, and human-created summarization datasets are scarce, especially for less-resourced languages. To address this issue, we propose an unsupervised approach to user comments summarization, which uses a modern multilingual representation of sentences together with standard extractive summarization techniques. Our comparison of different sentence representation approaches coupled with different summarization approaches shows that the most successful combinations are the same in news and comment summarization. The empirical results and presented visualisation show usefulness of the proposed methodology for several languages.

1 Introduction

Readers of news articles are often interested in what others think, what their perspectives are, and whether they can get any additional information from them. User comment sections on news web pages are often a good source for extending, presenting, and challenging their own views. On the other hand, many news providers see user comments sections of their websites as a way to connect to their readers, get relevant feedback, and sometimes even extract complementary information.

Many news articles get a large number of comments in a short time, which is especially true for popular and controversial topics. When dealing with an individual article, users can usually sort comments by relevancy or publishing time. While not ideal, this is satisfactory to get insight into the most popular thread or discussion but lacks in providing an overview of the whole discussion (Llewellyn et al., 2014). This, together with the low amount of time users are willing to spend in reading comments, is one of the reasons to automatically provide comprehensive overviews of discussions.

User comments can be irrelevant, deceiving, and may contain hate speech. Language is often informal with ill-formed sentences full of spelling and grammatical errors that are hard to understand. Because of that, comments are easily dismissed as not worth the attention and time. In addition, non-standard expressed content is difficult to encode into an informative numerical representation as standard embedding techniques are mostly based on more standard language (Gu and Yu, 2020).

The goal of text summarization is to compress original data and present it in a shorter form conveying the essential information (Allahyari et al., 2017). Two main approaches exist, extractive and abstractive. The extractive summarization approach selects essential information and does not modify content; its goal is to copy the most informative non-redundant sentences, phrases, or other units of a text. The abstractive approach is similar to how humans summarise documents. It may use new words and expressions, compress long sentences, combine multiple sentences, replace phrases, etc. Current neural network based abstractive approaches mostly provide useful and fluent summaries for short texts but offer no guarantee concerning text correctness (Dong et al., 2020; Cao et al., 2020).

News article summarization is a well-defined and the most studied task within the field of automatic text summarization with several available datasets suitable for supervised learning (Bommasani and Cardie, 2020). For this task also several unsupervised methods exist, based on graph centrality approaches or clustering. On the other hand, the user comment summarization task is not well-defined and established. In a survey paper on user comments, Potthast et al. (2012) describe it as the extraction of sentences that express an opinion. This proposal categorises it as an information retrieval task, close to comment filtering and comment ranking. We believe that this categorisation is limited as it does not consider many other aspects, such as complementarity of information, coverage of different topics and opinions, impact on public discourse, possibly offensive speech, non-standard language, etc.

Cross-lingual approaches to text processing (Ruder et al., 2019) enable the transfer of trained models from resource-rich languages to low-resource languages. Many multilingual neural sentence representation models were released (Artetxe and Schwenk, 2019; Reimers and Gurevych, 2019; Feng et al., 2020; Yang et al., 2020), which presents an opportunity to improve standard unsupervised extractive approaches (Mihalcea and Tarau, 2004; Erkan and Radev, 2004; Llewellyn et al., 2014) that use sparse representations such as TF-IDF weighted bag-of-words.

In this work, we developed an unsupervised extractive approach to text summarization that combines traditional unsupervised methods (graph and clustering-based) with the above-mentioned state-of-the-art multilingual sentence encoders. We assess these encoders in combination with different extractive summarizers and dimensionality reduction techniques. We used Croatian, English and German datasets containing news articles and user comments.

Our main contributions are:

- To the best of our knowledge, we present the first multilingual unsupervised approach to automatic summarization of user comments using modern neural sentence embeddings.
- We analyse and visualize the performance of state-of-the-art multilingual sentence encoders on both clustering-based and graph-based summarization methods.
- We create a dataset of Croatian news articles appropriate for news summarization task.

The paper consists of six sections. In Section 2, we present the related work. Section 3 contains description of datasets we used. In Section 4, we outline and explain our approach to unsupervised text summarization. Section 5 presents visual and automatic evaluation of the results. In Section 6, we summarize the work done, present limitations of our approach, and ideas for further work.

2 Related work

In this section, we present related research on comment summarization and other related summarization tasks.

User comments can be divided into comments on non-textual resources (photos or videos) and comments on textual resources (news articles, product reviews, etc.) (Ma et al., 2012). Potthast et al. (2012) argue that the most important tasks done on comments are filtering, ranking, and summarization. We focus on the latter two.

Most of the research on user comments summarization uses unsupervised extractive approaches that combine ranking and clustering methods. Khabiri et al. (2011) used LDA for clustering, and ranking algorithms (MEAD, LexRank) to summarize comments on YouTube videos. Ma et al. (2012) developed a topic-driven approach in which they compared clustering methods and ranking methods (Maximal Marginal Relevance, Rating & Length) on comments from Yahoo News. Llewellyn et al. (2014) used standard clustering and ranking methods (K-means, PageRank, etc.) to summarize the comments section of the UK newspaper The Guardian. Hsu et al. (2011) proposed a hierarchical comments-based clustering approach to summarize YouTube user comments. All listed methods use classical text representation approaches, while we propose the use of modern neural sentence embedding methods.

A related task to comment summarization is discussion thread summarization. The distinctive difference is that original posts are very different from news articles. van Oortmerssen et al. (2017) used text mining to analyze cancer forum discussions. In addition to ranking and clustering, Alharbi et al. (2020) use hand-crafted text quality features such as common words between the thread reply and the initial post, a semantic distance between thread reply and thread centroid, etc. The conversation summarization (Murray and Carenini, 2008; Chen and Yang, 2020), email summarization (Kaur and Kaur, 2017), and Twitter Topics summarization (Sharifi et al., 2010) are also relevant related tasks.

3 Datasets

In this section, we first describe the creation of two Croatian summarization datasets used in our research: news articles, and user comments. We also present English and German dataset of user comments.

The CroNews summarization dataset was created from the corpus of approximately 1.8 million news articles from the popular Croatian 24sata news portal[1]. The second dataset (CroComments) is a small evaluation dataset (Milačić, 2020) and contains user comments of 42 articles from Croatian Večernji list website[2], together with their short human-written abstractive summaries[3].

We preprocessed the news articles from the news corpus into a one-sentence-per-line form using the Croatian tokenizer available in the Stanza NLP package (Qi et al., 2020). The user comments in CroComments were already preprocessed in a similar way (Milačić, 2020).

The articles in the original news dataset contained no summaries. We took the first paragraph of an article as a proxy for a summary. In the dataset, this paragraph is named 'lead'. We sampled 5000 (from a total of 17 194) examples that satisfied the next criteria: more than 6 and less than 30 sentences were present in an article (we presupposed that articles with less than 6 sentences are too short for summarization), and the overlap between the abstract (lead) and article text was within 40 and 90 ROUGE-L points. The last criterion was designed to make sure that the first paragraph of an article overlaps with the rest of it in terms of content but we avoided strictly duplicated content. Most of the abstracts have a missing period at the end. We fixed that by appending it at the end of an article. We call the resulting dataset CroNews in the remainder of the paper.

While we focused on the Croatian language, to assess the multilingual potential of the proposed approach, we tested it also on English and German. For English, we used the New York Times Comments corpus[4] with over 2 million comments. For German, we used One Million Posts Corpus (Schabus and Skowron, 2018) with 1 million comments from the Austrian daily broadsheet newspaper DER STANDARD.

[1]https://www.24sata.hr/
[2]https://www.vecernji.hr/
[3]Available upon email request.
[4]https://www.kaggle.com/aashita/nyt-comments

4 Methodology

In this section, we describe our approach to unsupervised (multilingual) summarization which is comprised of two main components:

1. Neural sentence encoders represent the text in a numeric form as described in Section 4.1. This can be done in a cross-lingual manner to project many languages in the same numeric space and makes our approach multilingual.

2. From the numeric representation of sentences in the commentaries below a given article, we select the most representative sentences to be returned as summaries. To achieve that, we use two groups of approaches as described in Section 4.2: clustering-based and graph-based. Clustering approaches group similar sentence vectors and select the representative sentences based on the proximity to the centroid vector. Graph-based methods construct a graph based on the similarity of sentence vectors and then use graph node rankings to rank the sentences. The best-ranked sentences are returned as the summary.

As a further, optional component of our approach, the sentence vectors can be mapped to two-dimensional space with dimensionality reduction techniques (we use PCA or UMAP) and visualized in an interactive graph. To demonstrate these capabilities, we released a Jupyter notebook on Google Colab[5].

4.1 Sentence representation

In order to cluster or rank sentences in user comments, we have to first transform them from a symbolic to numeric form. In our work, we use sentence-level representation, as the extractive summarization techniques we use work on this level. Sentence embeddings aim to map sentences with a similar meaning close to each other in a numerical vector space. Initial approaches to sentence embeddings averaged word embeddings, e.g., GloVe (Pennington et al., 2014) vectors, or created Skip-Thought vectors (Kiros et al., 2015). A successful massively multilingual sentence embeddings approach LASER is built from a large BiLSTM neural network on parallel corpora (Artetxe and Schwenk, 2019).

[5]https://colab.research.google.com/drive/12wUDg64k4oK24rNSd4DRZL9xywNMiPil?usp=sharing

Recently, the Transformer architecture (Vaswani et al., 2017) is the most successful and prevalent neural architecture for the majority of language processing tasks, especially if pretrained on large corpora using masked language model objective, such as the BERT model (Devlin et al., 2019). In sentence embedding, naive solutions, e.g., averaging BERT output layer or using the first CLS token in the BERT architecture, often produced results worse than averaging of word vectors.

We used three competitive transformer-based sentence encoders. Reimers and Gurevych (2019) created siamese and triplet networks to update the weights and enable comparison of sentences. Their model called SBERT adds a pooling operation to the output of BERT to derive a sentence embedding. They trained it on natural language inference (NLI) datasets. Feng et al. (2020) combined masked language model and translation language model to adapt multilingual BERT and produced language-agnostic sentence embeddings for 109 languages. Their model is called LaBSE (Language-agnostic BERT Sentence Embedding). Yang et al. (2020) proposed a novel training method, conditional masked language modeling (CMLM) to learn sentence embeddings on unlabeled corpora. In CMLM, a sentence depends on the encoded sentence level representation of the adjacent sentence.

Our sentence embedding vectors have 768 dimensions. A dimensionality reduction may improve clustering due to noise reduction. To test that hypothesis, we tested two variants of sentence selection approaches (both graph and clustering-based): with and without dimensionality reduction. For the dimensionality reduction down to two dimensions, we tested PCA and UMAP (McInnes et al., 2018) mthods. We set the neighbourhood value of UMAP to 5, the number of components to 2, and the metric to Euclidian.

4.2 Selecting representative sentences

Once the sentences of comments belonging to a certain article are represented as numeric vectors, we have to select sentences for the summary. We use two types of approaches: i) clustering the sentences and returning the most central sentences from each cluster, and ii) representing sentences as nodes in a graph, based on their similarities and selecting the highest-ranked nodes as the summary.

For clustering, we used k-means and Gaussian mixture algorithm. We set the number of clusters to 2 because in our experimental evaluation we decided to extract only the best two sentences. We extracted the best sentences based on their proximity to centroid vectors of the clusters returned by the clustering algorithms. Clustering methods deal well with the redundancy of extracted sentences as the extracted sentences are by construction very different.

Graph-based ranking algorithms score the importance of vertices within a graph. A popular method to determine the importance of a vertex uses the number of other vertices pointing to it and the importance of the pointing vertices. In our case, each vertex in a graph represents a sentence. We used the TextRank (Mihalcea and Tarau, 2004) method, inspired by the PageRank algorithm (Page et al., 1999) that can be intuitively explained with the concept of eigenvector centrality or stationary distribution of random walks. For a similarity measure of sentences, we used the cosine similarity computed on sentence vectors.

We used two baseline summarization methods: i) selecting random $n = 2$ sentences (BaseRand), and ii) selecting the first $n = 2$ sentences (BaseLead).

For both clustering and dimensionality reduction, we used the scikit-learn implementations in python (Pedregosa et al., 2011). For the graph-based approach, we used PageRank from the NetworkX python library (Hagberg et al., 2008).

5 Evaluation

In this section, we first provide visualization of sentence embeddings, followed by the analysis of summarization. The visualization demonstrates the suitability of the proposed cross-lingual sentence representation for unsupervised summarization. In summarization experiments, we first present results of news article summarization, followed by the commentaries.

5.1 Visualization of sentence embeddings

We first visually demonstrate the utility of used sentence embeddings in a multilingual setting. In Figure 1, we show a visual evaluation of the proposed cross-lingual sentence representation for the unsupervised summarization. The dots in the image are sentence vectors of the synthetic sentences (described below). The image was produced using the Gaussian Mixture clustering using the sentence representation produced with the SBERT encoder and PCA dimensionality reduction. Sentences of vari-

92

ous lengths corresponding to three topics (school, weather, and music) were written in Slovene and translated into English, Croatian, and German. The three large colored clusters correspond to three topics, which is an indication that the sentence representation captures different contents well. We can observe also small groups of four sentences (an original Slovene sentence and three translations of it) that confirm the accuracy of the multilingual sentence encoder. The translated sentences are close together which is an indication that the representation is semantically adequate even in the multilingual setting. The rectangle on the top contains the sentences: Šolsko leto se je začelo drugače kot ponavadi; The school year started differently than usual; Školska godina započela je drugačije nego inače; Das Schuljahr begann anders als gewöhnlich. The rectangle on the right shows: Vreme bo jutri lepo; The weather will be nice tomorrow; Vrijeme će sutra biti lijepo; Das Wetter wird morgen schön sein. The rectangle on the left consists of: Kitara je zelo popularen glasbeni inštrument; The guitar is a very popular musical instrument; Gitara je vrlo popularan glazbeni instrument; Die Gitarre ist ein sehr beliebtes Musikinstrument.

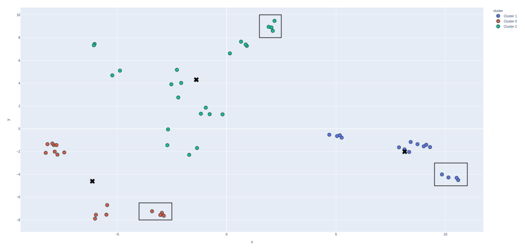

Figure 1: Example of Gaussian Mixture clustering with SBERT encoder and PCA dimensionality reduction of sentences from three topics (school, music, and weather, shown in green, blue, and red, respectively) and four languages. The sentences in the rectangles contain the same text in four languages (Slovene, English, Croatian, and English). The rectangle on the top contains the sentence "The school year started differently than usual.", the right one is "The weather will be nice tomorrow.", and the left one is "The guitar is a very popular musical instrument.".

5.2 News summarization

Due to the shortage of supervised data for automatic evaluation of user comments, we first test our unsupervised approach on the CroNews dataset, constructed as described in Section 3. We expected that the results would give us an insight into the performance of different combinations of methods, described in Section 4.

The results in Table 1 show commonly used ROUGE metric. The best performing experimental setup uses the LaBSE sentence encoder, no scaling, and the TextRank algorithm for sentence selection. The BaseLead baseline is 4.5 points behind the best model and ranked somewhere in the middle of all combinations. This corresponds with the findings of Zhu et al. (2019), who analysed the phenomenon of lead bias in news article summarization task. The BaseRand baseline is near the end of the ranks, as expected.

Enc.	Scaling	Summary	R-1	R-2	R-L
None	None	BaseLead	36.46	24.04	34.52
None	None	BaseRand	35.07	23.69	33.47
CMLM	None	GaussMix	35.29	22.77	33.52
CMLM	None	K-means	34.33	21.87	32.58
CMLM	None	TextRank	39.37	26.95	37.65
CMLM	PCA	GaussMix	35.71	23.90	34.17
CMLM	PCA	K-means	35.69	23.93	34.12
CMLM	PCA	TextRank	39.58	27.61	37.98
CMLM	UMAP	GaussMix	36.99	25.14	35.35
CMLM	UMAP	K-means	37.05	25.15	35.42
CMLM	UMAP	TextRank	38.65	26.94	37.06
LaBSE	None	GaussMix	38.81	26.41	37.04
LaBSE	None	K-means	37.70	25.18	35.92
LaBSE	None	TextRank	40.07	28.42	39.00
LaBSE	PCA	GaussMix	36.04	24.06	34.41
LaBSE	PCA	K-means	35.95	23.85	34.30
LaBSE	PCA	TextRank	38.69	26.80	37.10
LaBSE	UMAP	GaussMix	36.84	24.92	35.28
LaBSE	UMAP	K-means	37.22	25.31	35.63
LaBSE	UMAP	TextRank	37.90	25.86	36.29
SBERT	None	GaussMix	37.36	25.09	35.64
SBERT	None	K-means	37.05	24.65	35.26
SBERT	None	TextRank	38.63	26.55	36.99
SBERT	PCA	GaussMix	36.34	24.34	34.71
SBERT	PCA	K-means	36.42	24.48	34.81
SBERT	PCA	TextRank	37.86	26.11	36.31
SBERT	UMAP	GaussMix	36.94	25.14	35.38
SBERT	UMAP	K-means	36.92	25.06	35.38
SBERT	UMAP	TextRank	36.38	24.48	34.83

Table 1: Results expressed as ROUGE scores on the CroNews dataset. Colors correspond to ranks, darker hues correspond to better scores.

Statistics of different parameters in Table 2 show that LaBSE achieved on average 0.6 more ROUGE-L points than SBERT and CMLM, which are close in terms of performance. UMAP scaling preserved information better than PCA for 0.3 points but achieved 0.4 points less compared to no scaling. TextRank ranking method is superior to clustering for more than 2 points.

MatchSum (Zhong et al., 2020) is currently the

93

Group	Mean	Std	Min	Max	95%CI	Size
Encoder						
LaBSE	36.11	1.47	34.30	39.01	(34.98, 37.25)	9
SBERT	35.49	0.75	34.71	36.99	(34.91, 36.06)	9
CMLM	35.32	1.91	32.58	37.99	(33.86, 36.79)	9
Scaling						
None	35.96	2.01	32.58	39.01	(34.42, 37.50)	9
UMAP	35.63	0.66	34.84	37.06	(35.12, 36.14)	9
PCA	35.33	1.44	34.12	37.99	(34.22, 36.43)	9
Summarizer						
TextRank	37.03	1.18	34.84	39.01	(36.13, 37.93)	9
Clustering	34.94	1.00	32.58	37.04	(34.45, 35.44)	18

Table 2: ROUGE-L scores grouped by sentence encoder, scaling, and type of summarizer.

best extractive summarization model. It was trained on the large CNN/Daily Mail dataset and achieved 44.41 ROUGE-1 and 40.55 ROUGE-L scores. As we can observe from Table 1, our best scores for the Croatian news lag approximately 4.3 ROUGE-1 and 2.5 ROUGE-L points behind these scores which is a relevant difference in performance. However, we have to take into account that we use leads as an approximation for the summaries.

5.3 User commentaries summarization

We used the same experimental setup, as reported in Table 1, to summarize the CroComments dataset. The results of both datasets are very similar if we rank the models, with the best models being identical. TextRank with CMLM or LaBSE encoder is superior to clustering. Surprisingly, SBERT shows significantly lower performance with both clustering and ranking (with ranking worse than clustering).

We identified a few reasons that explain the lower scores of comment summarization compared to news summarization. For comments, the sentence encoders face a more challenging task of encoding the informal language; for the same reason, the accuracy of a sentence tokenizer is also significantly lower, as our inspection revealed. A single CroComment document (containing all comments related to one news article) is usually comprised of texts by several authors, of variable length, and written in different styles. CroComment documents are longer and exhibit a greater length variability. The average length of a document is 19.81 sentences with the standard deviation of 13.16 in comparison to CroNews dataset which contains 7.85 sentences with the standard deviation of 1.42. These differences make the comment summarization task difficult for a model trained on standard language in much shorter news articles.

Enc.	Scaling	Summary	R-1	R-2	R-L
CMLM	None	K-means	24.44	11.50	23.18
CMLM	None	TextRank	33.08	17.24	31.09
CMLM	PCA	GaussMix	19.71	08.53	18.79
CMLM	PCA	K-means	22.30	10.66	20.64
CMLM	PCA	TextRank	26.01	12.50	24.60
CMLM	UMAP	GaussMix	24.83	12.18	23.28
CMLM	UMAP	K-means	23.88	10.44	22.37
CMLM	UMAP	TextRank	23.02	11.78	22.31
LaBSE	None	GaussMix	26.77	13.39	25.77
LaBSE	None	K-means	26.59	12.89	25.01
LaBSE	None	TextRank	34.35	18.50	32.28
LaBSE	PCA	GaussMix	24.15	11.61	22.90
LaBSE	PCA	K-means	25.32	14.17	24.63
LaBSE	PCA	TextRank	28.53	15.60	26.95
LaBSE	UMAP	GaussMix	26.39	12.99	24.28
LaBSE	UMAP	K-means	27.36	14.45	26.04
LaBSE	UMAP	TextRank	24.99	12.50	23.80
SBERT	None	GaussMix	25.34	12.43	23.82
SBERT	None	K-means	26.13	12.84	24.67
SBERT	None	TextRank	25.20	11.71	23.25
SBERT	PCA	GaussMix	21.78	09.98	20.51
SBERT	PCA	K-means	23.96	11.46	22.47
SBERT	PCA	TextRank	25.44	11.40	23.76
SBERT	UMAP	GaussMix	25.29	13.00	24.16
SBERT	UMAP	K-means	24.94	12.04	23.62
SBERT	UMAP	TextRank	24.44	10.92	22.98

Table 3: Results expressed with ROUGE scores on the CroComments evaluation dataset with human-written summaries of comments. Colors correspond to ranks, darker hues correspond to better scores.

As an example, Table 4 shows comments belonging to one selected article. We tokenized comments, encoded them with the LaBSE sentence encoder, and scored with the TextRank algorithm. The sentences with the highest score in each user comment are typeset with red, and two highest scored sentences are shown in green. The value 'ref' in the column 'Id' indicates the human-written abstractive summary of the listed comments; the value 'lead' means the first paragraph of the article. Notice that the human-written summary and the high-scored sentences strongly overlap.

Comment no. 54412 demonstrates how the tokenizer and encoder face a difficult task. It is evident that the comment should have been split into several sentences to improve readability, has missing punctuation, and does not contain letters with the caron. Comment no. 54299 shows the limitation of extractive approaches since it cannot be understood properly without the context. The comment with the lowest score (no. 56141) does not add much to the conversation.

Table 5 shows an example from New York Times

94

Id	Croatian text	English translation
lead	Svaki gost koji je došao u Hrvatsku 2009. godine nije poklonjen, morali smo se za njega izborili. Ovakav učinak, uz ostalo, rezultat je mjera koje smo poduzeli, uz lijepo, sunčano vrijeme. Sunce je ove godine sjalo i u Turskoj, Francuskoj, Španjolskoj, ali očito nešto bolje u Hrvatskoj, slikovit je bio ministar Bajs.	Every guest who came to Croatia in 2009 was not given away, we had to fight for him. This effect, among other things, is the result of the measures we have taken, with nice, sunny weather. This year, the sun was shining in Turkey, France, Spain, but obviously somewhat better in Croatia, Minister Bajs was picturesque.
54279 score: 0.0552	Hrvatski turizam je u plusu za 0,2 Bravo,bravo,bravo . Pravi turizam ce poceti u Hrvatskoj tek tada kad nebude vise nitko od vas smdljivaca u vladi . Otvorite ovi ljudi , pa austrija napravi vise novaca od turizma nego Hrvatska . Svaku godinu smo u plusu a love nigdje pa naravno kad od 10-15% ostane samo 0.2 % . Koji su to muljat3ori i od kuda imate taj podatak . Revolucija je jedini spas , skidam kapu Rumunjima , oni su to fino rijesili . Bog i Hrvati	Croatian tourism is in the plus by 0.2 Bravo, bravo, bravo. Real tourism will start in Croatia only when there are no more of you smugglers in the government. Open these people, and Austria will make more money from tourism than Croatia. Every year we are in the red and the money is nowhere to be found, so of course when only 0.2 % of 10-15 % remains. What are these scammers and where do you get that information from. Revolution is the only salvation, I take my hat off to the Romanians, they solved it fine. God and Croats
54299 score: 0.0587	To vam je tako : 1999 godine Amerikanci su sredili stanje na Kosovu i cijela Europa a i druge države dale su zeleno svjetlo svojim građanima da mogu na ljetovanja u hrvatsku i ostali dio Balkana.2000 godine dolazi za ministricu turizma gospođa Župan - Rusković . Ta godina pokazuje se za turizam dobra i to se pripisuje SDP -u i gospođi ministarki . Ove godine sunce jaće i duđe sije pa eto to se pripisuje ministru Bajsu . Ja ču im samo poručiti . Ne bacajte pare na \" promocije \" jer svijet zna za nas , radije te novce ulažite u izobrazbu turističkoga i ugostiteljskoga osoblja . To bi bio naš največi uspjeh .	This is how it is for you: in 1999, the Americans settled the situation in Kosovo and the whole of Europe, and other countries gave the green light to their citizens to go on vacation to Croatia and the rest of the Balkans. In 2000, Ms. Župan - Rusković came to be Minister of Tourism. That year proves to be a good thing for tourism and it is attributed to the SDP and the Minister. This year the sun is shining stronger and longer, so that is attributed to Minister Bajs. I'll just tell them. Don't waste money on \"promotions \" because the world knows about us, rather invest that money in the training of tourism and catering staff. That would be our greatest success.
54311 0.0448	Sezona je ove godine bila iznad prosjeka i normalno da je Bajs ponosan	This season has been above average and it's normal for Bajs to be proud
54412 score: 0.0534	slazem se sa Somelier , a po izjavama i komentarima sto daje ministar Bajs vidi se nema veze s turizmom , HR je konkurentna samo u o dredjenim vrstama turizma (nauticki turizam) i trebalo bi se fokusirati upravo na njih koji usput najvise i trose , a ne slusati ove gluposti Bajsa da je sezona uspjesna zato sto je dozvolio onim krsevima od aviona da slijecu ili zato sto je dao 20 miliona € za reklamu na googlu i eurosportu	I agree with Somelier, and according to the statements and comments given by Minister Bajs, there is nothing to do with tourism, HR is competitive only in o dredged types of tourism (nautical tourism) and we should focus on those who spend the most, and not listen to this nonsense of Bajs that the season was successful because he allowed those breaches of planes to land or because he gave 20 million € for advertising on google and eurosport
54413 score: 0.0582	Bajs , kaj nas briga kak su turistički tržili u Austriji , Italiji , Francuskoj ili Grčkoj ? Raci ti nama zakaj je u Hrvatskoj bilo manje turistof neg lani iako ti tvrdiš da mi imamo kakti prednost kao auto destinacija ? Zakaj i u onom jednom jadnom mesecu kad je bilo više turistof nek lani ima manje lovice ? Zakaj se inšpekcije i dalje zezaju sa boravišnim taksama vikendaša dok ugostitelji premlaćuju goste , ne izdaju račune i jasno , ne plačaju poreze , uključujući i PDV ?	Bajs, do we care how they marketed tourism in Austria, Italy, France or Greece? Tell us why there were fewer tourists in Croatia than last year, even though you claim that we have some advantage as a car destination? Why, even in that poor month when there were more tourists, let there be less money last year? Why do the inspections continue to mess with the weekend taxes of the weekenders while the caterers beat the guests, do not issue invoices and clearly do not pay taxes, including VAT?
56141 0.0376	Nakon ove kostatacije sa zadovoljstvom mogu kostatirati da je Bajs napredovao sa jedne na dvije litre dnevno.	After this casting, I am pleased to say that Bajs has progressed from one to two liters a day.
ref.	Hrvatski turizam u porastu , uspješna sezona . Vlada je problem i ne ostaje dovoljno novca . Ne bacajt pare ne promocije već ulažite u izobrazbu turističkoga i ugostiteljskoga osoblja . Baj ponosan na sezonu iznad prosjeka . HR je konkurentna samo u određenim vrstama turizma i trebalo bi se fokusirati na njih . Zakaj je manje turista nego lani i nanje novca . Inspekcije se zezaju sa boravišnim taksama a ugostitelji premlaćuju goste , ne izdaju račune i ne plaćaju poreze .	Croatian tourism on the rise, successful season. The government is a problem and there is not enough money left. Don't waste money on promotions, but invest in the training of tourism and catering staff. Bajs proud of the above average season. HR is competitive only in certain types of tourism and should focus on them. Why are there fewer tourists than last year and money for them. Inspections mess with sojourn taxes and caterers beat guests, do not issue invoices and do not pay taxes.

Table 4: Visualization of the most important sentences in each user comment (in red). The original comments are on the left-hand side and their machine translations on the right-hand side. The reference score is at the bottom. Two sentences with the highest score are shown in green.

Comments, which was preprocessed and evaluated in the same manner as the example from Table 4. The selected sentences capture both prevalent themes (artistic freedom and racial questions) but exhibit the problem of redundancy. More examples from English, along with German, can be found on our source code repository[6].

6 Conclusion

We developed a multilingual unsupervised approach to user commentary summarization and tested it on a less-resourced Croatian language. Our models are based on cross-lingual neural sentence encoders, which make them easily applicable to many languages with little or no preprocessing. We tested several sentence representations and assessed the effect of dimensionality reduction. We used clustering and graph-based ranking algorithms to select sentences that form the final summaries. The results were promising both on the news articles dataset and the user comments evaluation dataset. The source code of our approach is freely available under the open-source licence.

The presented approach has several limitations. It only works within extractive summarization approaches, which do not allow sentence modification. With abstraction techniques, e.g., sentence compression, we could further distill the important information. We only tested sentence representation methods, while paragraph or document embeddings would also be sensible. We also did not exploit the information contained in the threading structure of the commentaries and possible relation of comments with the text of an original article.

In further work, we intend to exploit additional information in comments which was not used in the present study. The number of likes that a comment received could be used to weight sentences. Instead of working on a sentence-level, we could take a comment as a whole and embed it as a document. We plan to extend the work on visualization since it showed promising results, especially in the interactive exploration mode, inaccessible in the paper format.

Acknowledgments

The work was partially supported by the Slovenian Research Agency (ARRS) core research programme P6-0411 and the Ministry of Culture of Slovenia through project RSDO (Development of Slovene in a Digital Environment – Language Resources and Technologies). This paper is supported by European Union's Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media).

[6]`https://github.com/azagsam/xl-user-comments`

References

Abdullah Alharbi, Qaiser Shah, Hashem Alyami, Muhammad Adnan Gul, M Irfan Uddin, Atif Khan, and Fasee Ullah. 2020. Sentence embedding based semantic clustering approach for discussion thread summarization. *Complexity*, 2020:1–11.

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. 2017. Text summarization techniques: a brief survey. *arXiv preprint arXiv:1707.02268*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Rishi Bommasani and Claire Cardie. 2020. Intrinsic evaluation of summarization datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8075–8096.

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. 2020. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258.

Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4106–4118.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. 2020. Multi-fact correction in abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9320–9331.

Id	Text
24107006 score: 0.0282	This art is all about perception . It is about the point the artist is trying to make and how the viewer sees it . This art should not be limited because it is attached to an emotion these moments being recorded through art of a society that claims to be post racial opens the eyes of those who do not want to see and forces them to . This illustration does that and in my opinion that makes it so much more valuable because it does not just sit in silence it sends a message .
23235619 score: 0.0283	Artists should n't be limited or restricted in what they can do as an artist . Everyone should have a voice or take on a matter no matter how unpopular or offensive the opinion is . Censoring art defeats the creativity and free expression in art . Censorship perverts the message the artist try 's to convey .
22099108 score: 0.0273	I believe that all subjects should be fair game for an artist . It should n't matter if they are depicting a murder , or even if it 's " black subject matter " , every artist has a voice that deserves to be heard . As Smith writes " We all encounter art we do n't like , that upsets and infuriates us . " (1) I understand that some topics are difficult to talk about and that some art is can cause anger but I think that it is irrational to make topics off - limits because people do n't agree with it .
22098876 score: 0.0264	I personally believe that artists should be able to write about anything they want , drive to the studio , then turn those words into beautiful music . Music is an art and in art there are no limits so honestly whatever they feel is relevant to write about , they should have the freedom to do so . Regardless of peoples personal opinions artist should be comfortable to talk about what they want to talk about . " We all encounter art we do n't like , that upsets and infuriates us . " (Gilpin , 1) I understand that some subjects are very sensitive , but most of the things people do n't like to hear are usually cold hard facts about the dark side of society . A few examples would be , hate crimes against all races , racism in america , people killing other people . It s just the sad truth that a lot of people hate to hear . Music is a powerful - subject that can really impact a person .
22075721 0.0258	nothing should be in limited to artist . they should have the freedom to do what they pleased .
22054073 0.0252	I believe there is n't a problem when a white artist draws a topic that is related to discrimination against the Blacks . This artist may want to show that killing black people is wrong . It does n't matter if she 's white or black .
22041906 score: 0.0280	I do n't think that any topic is out of bounds to an artist . That is the idea of an artist , is n't it ? To talk about subjects that they think should be talked about , or that they feel motivated to bring attention to . I do n't think it is right to throw blame and anger towards one group because they are creating art about a different group . I understand why there is anger , but demanding that a work be destroyed is just absurd to me . Could the artist have done something differently ? Possibly , but demanding empathy and understanding from a group different than your own , and then saying their act of trying to do so is inappropriate just does n't make sense . I do n't think any one group " owns " history . History is a human experience . People as a collective own the histories that shaped the world they live in . That is the point of the exhibition . The exhibition description on the Whitney site says , " Throughout the exhibition , artists challenge us to consider how these realities affect our senses of self and community . " Instead of focusing on the color of the artists skin , we should be focusing on the point of the show .. how the painting makes us feel about ourselves and our communities , because I am sure that everyone could say that there is room for improvement when it comes to both .
22031632 score: 0.0219	The question of whether or not any group " owns " a portion of history is not the issue . It is about how that imagery is used , if it is used intelligently , and that it mimics an aspect of white racism : the historic practice of whites displaying the mutilated corpses of black people . To make the issue about censorship is to miss the point . Instead students should be asked to consider how a white person might have better handled her desire to show empathy .

Table 5: Visualization of the most important sentences in each user comment for a sample from the New York Times Comments dataset. Since the conversation is very long, we show here only a part of it. The green color stresses the best two sentences.

Günes Erkan and Dragomir R Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT sentence embedding. *arXiv preprint arXiv:2007.01852*.

Jing Gu and Zhou Yu. 2020. Data Annealing for Informal Language Understanding Tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3153–3159.

Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring network structure, dynamics, and function using networkx. In *Proceedings of the 7th Python in Science Conference*, pages 11 – 15.

Chiao-Fang Hsu, James Caverlee, and Elham Khabiri. 2011. Hierarchical comments-based clustering. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 1130–1137.

Kuldeep Kaur and Anantdeep Kaur. 2017. A survey on email summarisation techniques and its challenges. *Asian Journal of Computer Science And Information Technology*, pages 40–43.

Elham Khabiri, James Caverlee, and Chiao-Fang Hsu. 2011. Summarizing user-contributed comments. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 5, pages 534–537.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors.

In *Advances in Neural Information Processing Systems*, pages 3294–3302.

Clare Llewellyn, Claire Grover, and Jon Oberlander. 2014. Summarizing newspaper comments. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8, pages 599–602.

Zongyang Ma, Aixin Sun, Quan Yuan, and Gao Cong. 2012. Topic-driven reader comments summarization. In *21st ACM International Conference on Information and Knowledge Management*, pages 265–274.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. UMAP: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 404–411.

Katarina Milačić. 2020. Summarization of web comments. Master's thesis, University of Ljubljana, Faculty of Computer and Information Science.

Gabriel Murray and Giuseppe Carenini. 2008. Summarizing spoken and written conversations. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 773–782.

Gerard van Oortmerssen, Stephan Raaijmakers, Maya Sappelli, Erik Boertjes, Suzan Verberne, Nicole Walasek, and Wessel Kraaij. 2017. Analyzing cancer forum discussions with text mining. *Knowledge Representation for Health Care Process-Oriented Information Systems in Health Care Extraction & Processing of Rich Semantics from Medical Texts*, pages 127–131.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Martin Potthast, Benno Stein, Fabian Loose, and Steffen Becker. 2012. Information retrieval in the commentsphere. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4):1–21.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, 65(1):569–630.

Dietmar Schabus and Marcin Skowron. 2018. Academic-industrial perspective on the development and deployment of a moderation system for a newspaper website. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)*, pages 1602–1605, Miyazaki, Japan.

Beaux Sharifi, Mark-anthony Hutton, and Jugal Kalita. 2010. Automatic summarization of Twitter topics. In *Proceedings of National Workshop on Design and Analysis of Algorithm*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Ziyi Yang, Yinfei Yang, Daniel Cer, Jax Law, and Eric Darve. 2020. Universal sentence representation learning with conditional masked language model. *arXiv preprint arXiv:2012.14388*.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208.

Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2019. Make lead bias in your favor: Zero-shot abstractive news summarization. *arXiv preprint arXiv:1912.11602*.

Appendix B: API Definition of the Report Generation Service

- `GET` **/api/health** - Used to check version and health of service
- `GET` **/api/languages** - List supported languages
- `POST` **/api/report** - Produce a report from `multipart/form-data` input
- `POST` **/api/report/json** - Produce a report from `application/json` input

`GET` /api/health

Allows for monitoring of service health. A response with a status code other than HTTP 200 indicates the service is not healthy. If healthy, returns service version number.

Parameters

None

Example Response

```
{
  "version": "1.0.0"
}
```

`GET` /api/languages

Describes the languages supported by the Report Generator. All languages in the response are valid to be used as the `language` parameter in the `POST` **/api/report** request.

Parameters

None

Example Response

```
{
  "languages": [
    "en"
  ]
}
```

POST /api/report

Produces a natural language report from the comments in the request body. The response consists of two mandatory fields: `language`, which describes the language of the report and `body`, which contains the output text as a string of HTML content. The response can also contain an additional `errors` field, which describes any errors encountered during the generation process.

This endpoint assumes the body of the request has `Content-Type` of `application/json` and contains the following fields:

Parameters

Field	Description
`output_language`	The language the report should be written in. Valid values are those returned by the GET **/api/languages** endpoint.
`comment_language`	The language of the comments, if known. If present, language-specific resources (such as topic modeling) can be used if available for the indicated language. If not present (or set to ‘‘`all`’’), only multilingual models are used.
`comments`	A JSON list of comments, each comment expressed as a JSON string.

Example Response

```
{
  "language": "en",
  "report": "<p>...</p><p>...</p>",
}
```