A Tale of Two Revisions: Summarizing Changes Across Document Versions

Santosh T.Y.S.S[1*] and **Natwar Modani**[2] and **Apoorv Saxena**[2]
[1]Technical University of Munich, Germany
[2]Adobe Research, India
santosh.tokala@tum.de, {nmodani, apoorvs}@adobe.com

Abstract

Document revision is a crucial aspect of the writing process, particularly in collaborative environments where multiple authors contribute simultaneously. However, current tools lack an efficient way to provide a comprehensive overview of changes between versions, leading to difficulties in understanding revisions. To address this, we propose a novel task of providing thematic summary of changes between document versions, organizing individual edits based on shared themes. We assess capabilities of LLMs on this task and further introduce three strategies to tackle this task: (i) representing the input of two documents along with edits in the 'diff' format (ii) a two-stage task decomposition with individual edit description generation as an intermediate task and (iii) clustering based chunking and subsequent merging techniques for handling longer documents. Our experiments demonstrate the effectiveness of our approach in improving the model's capacity to handle this complex task. Additionally, we introduce ChangeSumm, a curated dataset comprising human-written thematic summaries for pairs of document versions, to facilitate evaluation and further research in this direction.

1 Introduction

Writing, inherently, is a dynamic and iterative process marked by the continual evolution of ideas, the assimilation of feedback and the collaborative efforts that shape the narrative (Flower and Hayes, 1980, 1981; Fitzgerald, 1987). In this journey, text revision emerges as a pivotal aspect—an adaptive process that involves discerning disparities between intended and realized text, determining the necessary edits, and deciding how to implement those changes (Scardamalia, 1986; Bridwell, 1980; Faigley and Witte, 1981). In the contemporary landscape, with the proliferation of collaborative document authoring systems introduces an additional layer of complexity with simultaneous edits by multiple authors turning the revision process into a labyrinth, challenging the comprehension of the document's evolution and resulting in inconsistencies (Oliver et al., 2018). While collaborative authorship fosters creativity and diverse perspectives, it presents the challenge of efficiently managing changes (Noël and Robert, 2004).

Consider a scenario where multiple authors or a single author make edits to a document using the track-change feature in document authoring and version management tools. Eventually, an author or an approver need to review these changes to incorporate them. Current tools display individual edit operations like 'insert' or 'delete', necessitating users to skim through each edit one by one to understand the overall changes across versions. This process makes it challenging to discern the cumulative effect of edits made over the whole document. Therefore, it becomes essential to reimagine the presentation of change logs, aiming for a design that facilitates swift human consumption.

While there has been prior work on understanding revisions, their main focus has been identifying the intent behind each atomic edit (Faruqui et al., 2018; Yang et al., 2017; Zhang et al., 2017; Anthonio et al., 2020; Spangher and May, 2021; Jiang et al., 2022; Mita et al., 2022; Rajagopal et al., 2022; Kashefi et al., 2022; Lee and Webster, 2012). Moreover, those intents deal with categorizing surface level edits at fine-grained level such as grammaticality, fluency, clarity, readability etc and associate addition of any new information to a broad intent of content level change. Though this can be viewed as some grouping of edit operations based on intent, it predominantly focuses on each edit in isolation, without considering the entire context of the document. Moreover, they lack an easy navigable way for users to understand various topical theme edits present in these content-level changes.

Our proposition centers around providing a condensed bird's-eye view of edits by offering a sum-

[*]Work done during internship at Adobe Research

Findings of the Association for Computational Linguistics: ACL 2024, pages 3195–3211

Figure 1: Overview of our proposed task of generating thematic summary of changes between two versions of a document. Colours beside each theme in the summary indicate attribtution to individual edits.

mary of changes between versions of the document organized based on shared common themes. Each theme in the summary is detailed with attributions to individual edits, providing a navigable way to review. For example, in Figure 1, we display two versions of a document with edits scattered throughout, but by providing overarching themes such as 'adding details of new dataset, unsupervised methods, and expert-based manual analysis', we alleviate the burden on the user to figure out the themes of the edits. More than a mere tracking tool, this assistant empowers approvers to effortlessly view and assess groups of edits emerging from shared themes with a single click, eliminating the need to review individual changes one by one. Such a method could further support the development of a system that can automatically identify inconsistencies across the document, suggest edits to maintain a cohesive narrative, and generate a response elucidating how feedback has been incorporated, thereby enhancing collaboration productivity.

To the best of our knowledge, our work represents the first attempt to define this problem and curate an evaluation-only dataset ChangeSumm containing 45 high-quality human-written thematic summaries along with attribution to individual edits. These versions, spanning both pre-review and camera ready revisions, from NLP-related scientific papers of the NLPEER dataset (Dycke et al., 2022). Such thematic summaries would greatly assist reviewers to comprehend how their feedback is incorporated into the document, fostering a more collaborative reviewing experience.

We evaluated the performance of GPT-4 and GPT-3.5 on our proposed task and found that they fail to correctly spot the edits between two versions exhibiting edit-hallucination, generating a span of text as an edit, when in fact, it may not represent an actual modification, and, at times, produce text that is not present in the original document. To tackle this complex task, we explore three strategies: (i) Providing input in the 'diff' format, explicitly indicating edits, alleviating the edit-hallucination problem. (ii) Two-stage task decomposition approach which involves generating edit description for each edit in the first stage, which are further used to form thematic clusters and summaries. (iii) A simple chunking-merging strategy for the first stage task of edit description generation and clustering-based chunking based on an external embedding model, to pack related edit descriptions belonging to each cluster into chunks to obtain a thematic summary in second stage, to handle the challenge of longer documents.

Our experiments validate that incorporating diff formatted input, two-stage task decomposition and clustering-based chunking strategies improve the LLM's capability to handle this complex task. In summary, our contributions are: (a) We propose the novel task of generating a thematic summary of changes between two document versions, along with attributions to individual edits. (b) We curate ChangeSumm, an evaluation-only dataset of 45 high-quality human-written thematic summaries for NLP-related scientific paper versions. (c) We explore three strategies to tackle this complex task using LLMs. Our annotation guidelines and dataset are available at https://github.com/natwar-modani/ChangeSumm.

2 Related Work

Edit understanding: : Early work by Faigley and Witte 1981 laid the foundation for edit intent classification, introducing a taxonomy that categorizes edits into surface-level meaning-preserving changes or text-based meaning-altering changes. However, this coarse-grained approach faced limitations in capturing nuanced intent. Subsequent efforts, such as Bronner and Monz 2012, introduced similar categorizations like factual versus fluency, yet fine-grained intent categorizations emerged from Wikipedia revisions corpus studies (Pfeil et al., 2006; Jones, 2008; Liu and Ram, 2011; Daxenberger and Gurevych, 2012). These fine-grained taxonomies included categories like vandalism, paraphrase, markup, spelling/grammar, reference, information etc. Notably, Zhang et al. 2017 integrated argumentative writing features and surface changes to construct eight categories of revision purposes, including claims/ideas, warrant/reasoning/backing, rebuttal/reservation, organization, clarify, etc., which tried to capture the purpose behind such an edit rather than the syntactic or semantic actions captured in earlier works.

Advancing on prior works, Yang et al. 2017 proposed a semantic taxonomy of edit intentions for Wikipedia, including clarification, elaboration, and fact updates etc. Subsequently, Wikipedia revisions have become a focal point in research on edit intent identification (Faruqui et al., 2018; Rajagopal et al., 2022; Du et al., 2022). These revisions have also served as valuable corpora for a spectrum of NLP tasks (Nelken and Yamangil, 2008; Ganter and Strube, 2009; Yamangil and Nelken, 2008; Zanzotto and Pennacchiotti, 2010), covering areas such as textual entailment (Zanzotto and Pennacchiotti, 2010), sentence simplification (Yatskar et al., 2010), and addressing spelling errors and paraphrases (Max and Wisniewski, 2010).

Moreover, diverse corpora have been curated for edit intent identification from domains such as Wikihow (Anthonio et al., 2020; Bhat et al., 2020), argumentative essays (Zhang et al., 2017; Zhang and Litman, 2015; Kashefi et al., 2022; Zhang and Litman, 2016; Afrin and Litman, 2019), news articles (Spangher and May, 2021), and scientific documents (Mita et al., 2022; Du et al., 2022; Jiang et al., 2022). These corpora served as valuable resources for various tasks, including writing quality assessment (Louis and Nenkova, 2013; Daudaravicius et al., 2016; Afrin and Litman, 2023; Liu and Ram, 2011), text simplification and compression (Xu et al., 2015), style transfer (Krishna et al., 2020), and hedge detection (Medlock and Briscoe, 2007), within their respective domains.

Beyond edit understanding, some works leverage feedback/comments/reviews to address the alignment task, aiming to identify the mapping between edits and the corresponding triggering comments (Kuznetsov et al., 2022; D'Arcy et al., 2023; Zhang et al., 2019). Furthermore, Zhang et al. 2019; D'Arcy et al. 2023 delve into edit suggestion based on the comment, proposing tasks like Edit Anchoring (Zhang et al., 2019), identifying the locations in a document that are likely to undergo edit as the result of a specific comment. D'Arcy et al. 2023 further advances it with the task of edit generation, generating a revised version of the document based on comments and the pre-review document.

In our work, we carve a distinct niche by targeting the generation of thematic summaries for edits given the document versions, an aspect not explored in prior works. While existing tasks on edit understanding primarily concentrated on each word/sentence-level edit in isolation, our primary focus is on the global context of the whole document, considering related edits comprehensively.

Document-level Long Text Tasks: Recent efforts, exemplified by Shaham et al. 2022, introduced SCROLLS, a suite of tasks (summarization (Huang et al., 2021; Chen et al., 2021; Zhong et al., 2021), question answering (Dasigi et al., 2021; Kočiskỳ et al., 2018; Bowman et al., 2022), and natural language inference (Koreeda and Manning, 2021)) requiring reasoning over long texts. Subsequently with focus on LLM evaluation, Shaham et al. 2023 proposed ZeroSCROLLS, an evaluation-only zero-shot benchmark derived from SCROLLS and and two other aggregation tasks (Angelidis et al., 2021; Kryściński et al., 2021), addressing long text reasoning. Such benchmark creation for long context capability evaluation include LongBench Bai et al. 2023 (both English and Chinese), L-Eval (An et al., 2023) and BAMBOO (Dong et al., 2023). Wang et al. 2023 comprehensively evaluated LLMs for document-level machine translation. In our work, we evaluate whether LLMs possess the capabilities to understand and cluster edits between versions, providing a thematic-level summary. We believe that this task can serve as a challenging testbed for evaluating LLMs in handling long contexts.

3 Task Definition & Dataset Curation

The task of the thematic summary generation is to provide a concise overview of edits between two versions of a document, organized based on common themes. Given the original (source) and revised (target) versions of a document, the task involves clustering the individual edits into groups, where edits within the same group share common themes and generating a descriptive summary for each thematic group, providing an overview of the underlying theme and associated changes in that thematic group. However, defining what constitutes a theme depends on individual interpretation. We define a theme as the collective set of edits deemed to be stemming from a specific comment or feedback from a reviewer, grouped under a single concept. While acknowledging the potential subjectivity, our definition provides a pragmatic framework for operationally tackling this task. Notably, thematic groups form a many-to-many mapping: one thematic group includes several edits and one edit can be associated with multiple thematic groups.

To construct a dataset with source and target paper and corresponding thematic summaries with attribution to edits under each theme, we first need to define the unit of edit. While phrase-based difference offers a fairly precise definition of the scope of a edit, it may be difficult to achieve consistent annotations and would also turn out laborious resulting in too many edits. We define our unit of revision to be at the sentence level. In other words, even if a sentence contains multiple edits, the entire sentence will be annotated as one edit unit (resulting in many-to-many mapping between edit units and themes). To extract the edit unit, we initially use a character-based text comparison algorithm first and then build edit units at sentence-level based on the differences extracted (Bronner and Monz, 2012).

Edit Segmentation: We use the Neil Fraser's google-diff-match-patch library [1] based on Myer's diff algorithm (Myers, 1986) enhanced with pre-diff speedups and post-diff semantic cleanups to produce the list of differences as a contiguous sequence of deleted, inserted or equal words. The difference is represented by a sequence of edit segments where each edit segment is a pair, where first element represents the operation {deleted, inserted, equal} and and second element in a substring which belongs to either or both documents.

Edit unit construction: To distill the edit segments into sentence-level edit units, we perform a linear traversal on the list of edit segments to identify the start and end indices for each target sentence. Simultaneously, we keep track of the substring of the source text that is covered within these index bounds of the edit segments list, creating an alignment pair between both the target sentence and the substring of source text covered in those bounds. If the target-source alignment pair covers all 'equal' edit segments (i.e., the same sentence in both source and target), we mark it as a non-edit unit. Otherwise, we mark the pair as an edit unit. During this traversal process, if there is a span of source text deleted between two target sentences, we include it as an edit unit with no corresponding target sentence in the alignment pair. Conversely, if no source text appears within the bounds of the target sentence, the corresponding target text is marked as an edit unit with the alignment complement in the source set to null. Detailed algorithm is presented in Appendix 1. Within each source-target alignment pair in the edit-units, we further detect and mark changes at the word level (i.e., deletions, insertions, and modifications). We achieve this by utilizing the difflib library[2], which tokenizes the sentences using the Spacy tokenizer[3] and processes the input as a list of words. This approach enables us to efficiently highlight the differences within those pairs. For a visual representation of the edit units obtained through this method, refer to left pane in Figure 2 where every edit unit is followed by 'Assign to group' button.

Revision Corpus: We leverage the NLPEER dataset (Dycke et al., 2022), comprising NLP-related scientific papers submitted for review and their corresponding camera-ready versions as source and target versions for our task. These papers originated from submissions to ACL-17 and CONLL-16 conferences collected as part of the Kang et al. 2018, with newly curated additions from COLING-20 and ARR-22 submissions. We obtain the parsed PDF documents from NLPEER dataset. Subsequently, we tokenize the obtained paragraphs into sentences using the scispacy tokenizer (Neumann et al., 2019). Utilizing the segmentation and edit unit algorithms described earlier, we extract sentence-level edit units from these

[1] https://github.com/google/diff-match-patch

[2] https://docs.python.org/3/library/difflib.html

[3] https://spacy.io/api/tokenizer

document versions. We select document versions based on the total number of edit-units where the total count of edit-units falls within the 30-70 percentile range across the corpus.

Annotation Workflow: Figure 2 illustrates the annotation interface developed for creating the ChangeSumm corpus, which involves writing thematic summaries for the revision corpus collected above. The left side of the interface presents the target document interspersed with the source document in the form of edit units. Each edit unit is marked and equipped with a button to assign it to thematic groups, which are dynamically populated as the user creates them. The right-hand pane serves as the workspace for annotators to create new thematic groups, displaying edit units already assigned to each group. Once all the edits are attributed to thematic groups, annotators can provide a final summary for each group, highlighting the theme and edits associated with it. To alleviate the annotation burden for common edits like spelling corrections and grammar updates, we automatically detect them using heuristics based on edit distance and root word analysis, assigning them to a thematic group of surface edits. However, annotators have the flexibility to make changes as needed.

Annotator Recruitment: Due to the time-consuming nature and the need for high annotation quality given the complexity of the task, we opted not to use Amazon Mechanical Turk. Instead, we recruited a small group of annotators from the Upwork freelancing platform. We deliberately refrained from imposing rigid guidelines that might stifle the creativity and opted for hands-on training sessions with annotators with discussions and providing examples illustrating effective and ineffective clustering for themes. They are provided a pre-annotation exercise for which they received detailed feedback to align their understanding with the meta-definition of theme. We release all our annotation guidelines and examples to encourage broader collaboration. Finally, they were assigned the task only if they successfully passed the pre-annotation exercise. Ultimately, we hired 7 Upwork freelancers and obtained a total of 95 responses. Given the subjective nature of the task, we then reviewed the responses and selected only those responses which were of good quality, and discarded the ones with quality issues (e.g., having clusters which categorized changes as insertion/deletion, etc., instead of theme or topic based grouping). Please note that the review of responses was done completely obviliously to the model outputs and only based on the quality of annotations only. Upwork workers were compensated with a base rate of $7-10 for each response (irrespective of their response getting selected or not).

ChangeSumm Dataset Inspection On average, each version pair consists of 75 edits, with the number of edits per pair varying between 25 and 172. Thematic summaries produced by annotators comprise an average of 14.5 themes per pair, ranging from 5 to 28 themes. The average number of edits attributed to each theme is 5.33, with the range spanning from 1 edit per cluster to 60. Around 25% of total edits are associated with multiple (more than 1) themes (13% mapped to 2 different themes and rest more than more than 2) and the rest assigned to a single theme.

4 Method

We propose a two stage task decomposition approach which takes the input of two documents in the diff format and produces description of each edit in first stage which are then passed as input into second stage to produce the clusters of edits based on shared theme and respective thematic description for each cluster. To account for longer inputs exceeding the context size of LLMs, we follow divide and conquer approach in both stages of the task. In the first stage, we chunk the input to pass into LLM and simply concatenate the obtained edit descriptions from each chunk. For longer input (edit descriptions obtained from first stage) in the second stage, we employ clustering based chunking approach wherein we cluster the edit descriptions using an embedding model and pack edit descriptions belonging to each of the clusters into chunks based on context length of LLM and and then concatenate the obtained chunk-level thematic summaries to obtain the final thematic summary. Detailed prompts for our method are presented in Appendix C.

Input Format Design: We present the input to the LLM in the textual diff format using XML tags. The input consists of the target document interspersed with the source document in the form of edit units. Each edit unit is enclosed within <edit x> </edit x> tags, where x is a specific number identifying the edit used for attributing each edit to thematic groups. Furthermore, word-level edits are represented with <add> </add> and <del></del>

tags to specify insertion and deletion segments within edit units. We keep the unit of text as paragraph to provide sufficient context to LLM for generating the description for diffs (so if a 5 sentence paragraph has change in one sentence, we still feed the entire paragraph to LLM, but identify the edits at the sentence level).

Two stage Task Decomposition: We postulate that an effective thematic summary generator should demonstrate the capability to comprehend each edit in isolation and subsequently group them based on shared themes. To achieve this, we introduce a two-stage task decomposition, with the intermediate task of edit description generation.

In the first stage, we present the model with input in the diff format, prompting the LLM to describe each edit in free-form text, capturing the underlying intent behind the modification. This initial stage focuses on the local context of individual edits. In the second stage, we provide the generated edit descriptions and prompt the LLM to generate a thematic summary. This involves clustering the edits into groups based on shared themes (referred to as attributing edits to thematic groups) and providing a theme description for each cluster. We instruct the model to produce output with attributions in inline citation format, where each theme description is followed by edit numbers to represent the attribution. In this stage, the emphasis shifts to establishing shared themes across all edits globally within the document.

Handling Longer Documents: When the length of the diff input exceeds the context window size of the LLM in first stage, we divide it into smaller, non-overlapping chunks. Chunking is performed at the paragraph level, ensuring that edit units do not get separated at the boundaries of chunks and we only pass those paragraphs with atleast an edit unit. Subsequently, all edit descriptions from each chunk are concatenated directly to obtain the first stage output. For the second stage process, in order to handle concatenated edit descriptions that exceed the maximum context size of LLMs, we employ a clustering-based chunking. We convert each edit description into embeddings using the Sentence-BERT model (Reimers and Gurevych, 2019) and then cluster these embeddings using affinity propagation clustering (Frey and Dueck, 2007) . We then pack edit descriptions from the same cluster into the same chunk and pack as many clusters as possible into each chunk based on the context length of the LLM. Each of these chunks is passed into the second stage LLM, resulting in thematic summaries for each chunk which are then concatenated into a final summary. As an alternative, we also pack edit descriptions in order of their occurrence in the LLM calls and make an LLM call to consolidate them into final summary clusters (using prompt shown in Appendix C.5.3).

5 Experiments & Discussion

We use GPT-4 and GPT-3.5 models (OpenAI, 2023) with 32k and 16k context window respectively, temperature=0.7, top-p=0.95 for all our experiments. We employ `all-MiniLM-L6-v2` model to obtain embeddings for clustering. We use pydantic [4] and jxnl instructor [5] library to enforce structured format of output for easy parsing.

5.1 Evaluation Metrics

We evaluate the model-generated thematic summaries along with attributions to the individual edits covered under each theme against curated reference data using three types of metrics. Additionally, we compute coverage as the fraction of edits covered by model-generated attributions to the total number of actual edits.

(i) Quality of Edit Groups: This metric assesses the model's ability to understand themes across edits, resulting in better clusters. We compute omega index (Murray et al., 2012)[6] which calculates the index of resemblance for overlapping clusters. To address incomplete coverage in the model generated clusters (this metric assumes complete coverage), we extend model-generated clusters with additional clusters, one for each missing edit number.

(ii) Quality of theme descriptions without edit attributions: For each theme description in reference, we find the maximum aligned one in model generated theme descriptions based on the ROUGE-L F-score[7] and BERT similarity score (Reimers and Gurevych, 2019) using `all-MiniLM-L6-v2` model and report the averaged similarity across all reference theme descriptions.

(iii) Quality of theme descriptions along with edit attribution: For each theme in the reference

[4] https://github.com/pydantic/pydantic

[5] https://github.com/jxnl/instructor

[6] https://cdlib.readthedocs.io/en/latest/reference/eval/cdlib.evaluation.omega.html

[7] https://lightning.ai/docs/torchmetrics/stable/text/rouge_score.html

	Cov.	Omega	RL w/o A.	BS w/o A.	RL w. A.	BS w. A.
GPT 4						
a. Text Input			0.08	0.35		
b. Text Input CoT			0.10	0.30		
c. Diff Input	0.91	0.16	0.11	0.36	0.08	0.26
d. Diff Input with Edit desc.	0.94	**0.19**	0.12	0.39	0.08	0.28
e. (d) in Two separate calls	**0.96**	0.17	0.12	0.38	0.08	0.30
f. (e) + Clustering-chunking	0.93	0.16	0.14	0.42	0.09	**0.31**
GPT-3.5						
a. Text Input			0.07	0.27		
b. Text Input CoT			0.08	0.27		
c. Diff Input	0.73	0.11	0.15	0.34	0.07	0.19
d. Diff Input with Edit desc.	0.72	0.03	0.20	0.44	0.09	0.24
e. (d) in Two separate calls	0.85	0.02	**0.21**	**0.47**	**0.10**	0.29
f. (e) + Clustering-chunking	0.79	0.03	0.17	0.46	0.09	0.28

Table 1: Performance comparison among different methods on ChangeSumm. A., RL, BS denote attribution, ROUGE-L and BERT similarity score respectively. Best values across the board and each model are bolded and underlined respectively.

summary, we compute the ROUGE-L F-score and Sentence-BERT similarity score with every generated theme description and compute weighted average of similarity scores with weights as the jaccard index of overlapping ratio between edit attributions covered under them. Finally, we report the average across all the reference theme descriptions.

5.2 Comparison with baselines

We consider the following two baselines: (a) Text Input which takes two documents in text format as input to the LLM. To address length limitations, we chunk the inputs and fill each chunk with task instruction and use half of the remaining space for the first version and the other half for the second version of the document. We chunk these documents at paragraph level to allow for better alignment. The model is instructed to generate thematic summaries directly. (b) Text Input with CoT Prompt: Similar to (a), but prompted in a chain of thought style, instructing the model to first derive the edit groups explicitly and then use them in attributions while generating thematic summaries.

Please note that for these two methods, since the edits are either not explicitly identified (in (a)), or may not correspond to the actual edits in the document, it is not possible to compute the coverage, ROUGE and BERT similarity based scores with attribution for these two methods. While we do not provide full numerical values, we checked the edits generated by CoT method, and there were several errors, both of false-negatives and false-positives types to render the output of these methods unsuitable to compute the other metrics.

From Table 1, we observe that both the *Text Input* models underperform compared to rest which employs Diff format for the input. This can be primarily attributed to limited coverage of *text input* as these methods require model to implicitly spot differences, but they fabricate sentences as insertion/deletion edits even when they are verbatim in the source documents. This problem is exacerbated with longer documents due to suboptimal chunking strategies, which may introduce various edits being recognized merely because the content is misaligned and require sophisticated alignment based strategies to deal with longer documents. Inputting in diff format with explicit edit unit boundaries and unique identifier numbers enclosed in XML tags explicitly addresses edit hallucination challenge and further simplifies the attribution of model-designated edits back to the actual text.

5.3 Ablation Study

To assess the impact of edit description generation task at the intermediate stage, we also evaluate a variant, which does not generate the descriptions for the diff input. This is identified as method c in Table 1 and uses the prompt as given in Appendix C.3. On the other hand, method identified as d in the table generates the edit descriptions first and in the same LLM call, also generates the the-

matic clusters. We introduce method (e) where edit description generation and thematic summary generation is handled in two separate calls. Method (f) in Table 1 involves clustering based chunking strategy for second stage.

From Table 1, we observe that this intermediate task helps the model generate better clusters and thematic summaries, along with attribution as well. This assists the model in understanding each edit in isolation, leading to better thematic clusters conditioned on edit descriptions. Further, we observe that using two separate calls with edit description generation as intermediate task in (e) yields better thematic summaries than a single call method (d) mostly across all metrics, in both models. This is due to generation of edit descriptions and thematic clusters in a single call results in longer text generation leading to low quality output towards the end of generation, which is also consistent with prior works of long-form text generation (Dong et al., 2023; An et al., 2023). We observe clustering based chunking strategy leads to better summaries in case of GPT-4, while remains slightly lower in GPT-3.5. We attribute this decrease in GPT-3.5 to the decrease in coverage compared to (e). As clustering based method reorders the edit sequence based on clusters obtained from external embedding model, this might have confused GPT-3.5 model to cover all the edits as opposed to ordered edit descriptions in (e). While we see a smaller drop in coverage in GPT-4, this reordering helped to provide better thematic summaries.

5.4 Effect of different LLMs

Overall, GPT 4 achieved higher coverage compared to GPT 3.5 by about $10 - 15\%$, which means that much fewer edits are left unassigned to themes by GPT 4 compared to GPT 3.5. We observe that GPT-3.5 tended to produce single edit based clusters in all methods except (c), resulting in notably low omega values. This suggests that GPT-3.5 struggled to produce effective clusters, particularly when tasked with multiple tasks in the task decomposition approach ((d)-(f)). These single-edit-based cluster descriptions often detailed each edit individually, leading to greater alignment at the summary level but without improved clustering (omega) scores, creating a spurious inflation of summary alignment scores.

	CQ	TS w/o A	TS w/ A
GPT-4			
c. Diff Input	3.2	2.8	2.8
f. (e) + Clustr. chunk	3.2	3.4	3.2
GPT-3.5			
c. Diff Input	2.6	2	2
f. (e) + Clustr. chunk	1.2	1.4	1.2

Table 2: Human evaluation of model generated thematic summaries.

5.5 Qualitative Evaluation

We randomly select 5 samples and their outputs generated from method (c) *Diff Input* and (f) *Diff Input with Edit Description in two separate calls with clustering based chunking* from GPT-4 and GPT-3.5, resulting in 20 input-output pairs. One of the author is tasked to rate the model generated thematic summaries across three granularities on a likert scale of 1 to 5: (i) Cluster Quality (CQ): How well are different edits clustered together? (ii) Thematic Summary without Attribution (TS w/o A): Does the summary reflect an overall understanding of the changes undergone between the versions of the document? (iii) Thematic Summary with Attribution (TS w/ A): Do the theme descriptions better reflect the overall changes while remaining faithful to the attributed edits under each theme?

From Table 2, we observe that our method yields better summaries for GPT-4 (indicated by high TS w/o A, TS w/ A), while being faithful to edits clustered under it. This improvement can be attributed to the generation of edit descriptions as an intermediate task, which helps the model focus on each edit by generating edit descriptions. This, in turn, translates to producing faithful summaries conditioned on them. Additionally, clustering-based chunking further assists in faithfulness by packing relevant edit descriptions into one chunk, alleviating the burden on the model to merge the related descriptions across the chunks.

On the other hand, we observe that GPT-3.5 summaries are scored lower than GPT-4 in Table 2, while the reverse trend is observed in Table 1. We attribute this mainly to the single sparse clusters produced by GPT-3.5, resulting in many themes, making them less effective for human consumption. This also aligns with the inflation effect of automated RL and BS scores for GPT-3.5 in Table 1, which do not reflect the summaries' lower utility, as we described earlier.

Error Pattern	Themes in the model generated summary	Edit Groups
Too Few Themes Dense Clusters	Changes related to the release of the code and model, introduction and functionalities of the SpeechT5 framework, deletion of certain details about pre-training techniques, and the broader context of the SpeechT5 model in the field of large-scale pre-training models.	2, 7, 12, 16, 17, 18, 19, 29, 30, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44
Too many themes Sparse Clusters	A conclusion is made that subtree substitution is a simple yet effective compositional data augmentation method for semantic parsing.	28
Generic Summaries	Influence of STILTs and its comparison to MTL	18, 19, 20, 21, 22, 23, 24, 25
Ignoring word level tags leading to misunderstanding edits	The second cluster is about the cross-modal vector quantization approach proposed to align the textual and speech information in a unified semantic space. <Group_2>To align the textual and speech information into this unified semantic space , we propose a <del>crossmodal </del><add>cross-modal </add> vector quantization approach that randomly mixes up speech/text states with latent units as the interface between encoder and decoder . </Group_2>	2, 12
Repeating sentence as summary	Following the pre-training techniques in NLP, self-supervised speech representation learning has also been investigated and shown promising results, benefiting from richly learned representations such as wav2vec 2.0 and HuBERT.	6, 8, 9

Table 3: Pattern of errors noticed in model generated thematic summaries. Detailed Edit group in textual format is only displayed when necessary.

5.6 Error Analysis

We outline various error patterns in the model-generated outputs in Table 3. The Diff Input approach with GPT-4 tends to produce very few themes with dense clusters, where multiple edits are grouped under each theme (First row). While these summaries encompass different aspects of the changes, they can be overwhelming for humans to comprehend. To address this, future research could focus on generating hierarchical themes to provide more granularity. On the other hand, GPT-3.5 often generates single-edit-based clusters, resulting in numerous clusters that are difficult to digest (Second row) while these summaries remain faithful to the actual edits.

In some cases, the summaries provide a generic description of the concept without specifying the changes made (Third row). Additionally, the model misunderstands the actual edit, considering an entire sentence as a single edit, leading to inaccurate thematic summaries (Fourth row). For example, a minor hyphenation addition is interpreted as a significant sentence change, attributing the edit to an overarching theme. This issue is mitigated by incorporating an edit description as an intermediate task, helping the model to understand each edit before proceeding to the actual task. Finally, there are instances where the model simply repeats entire sentences in the cluster without providing a concise summary of the underlying theme (Fifth row).

6 Conclusion

We introduced the task of providing thematic summaries of changes between document versions, aimed at offering users a comprehensive overview of edits organized around shared themes. To facilitate evaluation, we curated the ChangeSumm dataset, consisting of human-written thematic summaries for changes between document versions. Our experiments with LLMs revealed the challenging nature of this task and underscored its potential as a benchmark for assessing LLM performance on long-context document-level tasks. To address the complexities involved, we proposed three strategies: (1) Using the 'diff' format to represent document versions and edits, mitigating edit hallucination. (2) Introducing a two-stage task decomposition approach with edit description generation as an intermediate step. (3) Employing clustering-based chunking and merging techniques to address the limited context length of LLMs. Overall, we hope our work fuel further research in this direction to foster collaboration productivity in document revision processes.

Limitations

One notable limitation of our dataset and method is its restriction to text content exclusively. It does not involve other modalities such as images, graphs, or tables commonly found in documents. Our dataset is derived from documents from NLP related scientific papers, using the submissions for review and camera ready versions from conferences, in English. These can offer limited coverage in terms of the types of documents and revisions included, potentially limiting its generalizability to other types of documents. The effectiveness of our method may vary depending on the language and genre of the documents being analyzed.

The human-written thematic summaries in the ChangeSumm dataset may suffer from annotation bias, as annotators may interpret and summarize document revisions differently based on their perspectives, introducing inconsistencies. Given the inherent creative aspect of our task, the interpretation of what constitutes a theme can be subjective based on individual perspectives and biases. We provided extensive training sessions and supplemented with practical exercises for annotators to analyze and discuss. By actively involving annotators in the process and providing constructive feedback on their pre-annotation phase exercise, we aimed to align their understanding with a "meta-definition" of what constitutes a theme. Despite our efforts, achieving consensus among annotators on complex thematic summaries remains a daunting challenge. Furthermore, the quality of a thematic summary may vary significantly depending on its intended use and audience, suggesting the need for broader human evaluation to understand the utility and required characteristics of themes.

We are constrained by the extensive annotation effort required to gather labeled data for this challenging task, necessitating demanding cognitive effort to ensure high-quality annotations, limiting the quantity of our ChangeSumm dataset. We further filtered annotated data to enhance high quality of the dataset, prioritizing quality over quantity.

In this study, we compared model-generated edit attributions (clusters) and thematic cluster descriptions against human-annotated ones using established evaluation metrics like omega, ROUGE, and BERT similarity scores. While these metrics offer valuable insights, we recognize their limitations and are open for suggestions to create a more robust evaluation framework. To improve the evaluation process, we could explore the development of novel metrics tailored specifically to thematic summarization tasks. For instance, we could devise metrics that assess the likelihood of generating the revised content based on a provided thematic description. Additionally, we could consider metrics that evaluate the accuracy of identifying edit locations in the source document given the edit description or theme description, and then compare these against the target revised version.

Ethics Statement

In sourcing and curating the ChangeSumm dataset, we acquired the revision corpus from the NLPEER dataset (Dycke et al., 2022), ensuring compliance with all ethical standards and guidelines. Proper consent and permissions were obtained from the authors of NLPEER to utilize their dataset. Throughout the annotation process, we prioritized anonymity and confidentiality, safeguarding the identity of all participants involved and annotators were compensated with mutually agreed fair wage price. To minimize potential biases, we adopted a diverse pool of annotators and implemented rigorous quality control measures to ensure the accuracy and reliability of the annotations.

References

Tazin Afrin and Diane Litman. 2019. Identifying editor roles in argumentative writing from student revision histories. In *Artificial Intelligence in Education: 20th International Conference, AIED 2019, Chicago, IL, USA, June 25-29, 2019, Proceedings, Part II 20*, pages 9–13. Springer.

Tazin Afrin and Diane Litman. 2023. Predicting desirable revisions of evidence and reasoning in argumentative writing. *arXiv preprint arXiv:2302.05039*.

Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *arXiv preprint arXiv:2307.11088*.

Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. 2021. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.

Talita Anthonio, Irshad Bhat, and Michael Roth. 2020. wikihowtoimprove: A resource and analyses on edits in instructional texts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5721–5729.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Irshad Bhat, Talita Anthonio, and Michael Roth. 2020. Towards modeling revision requirements in wikihow instructions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8407–8414.

Samuel R Bowman, Angelica Chen, He He, Nitish Joshi, Johnny Ma, Nikita Nangia, Vishakh Padmakumar, Richard Yuanzhe Pang, Alicia Parrish, Jason Phang, et al. 2022. Quality: Question answering with long input texts, yes! *NAACL 2022*.

Lillian S Bridwell. 1980. Revising strategies in twelfth grade students' transactional writing. *Research in the Teaching of English*, pages 197–222.

Amit Bronner and Christof Monz. 2012. User edits classification using document revision histories. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 356–366.

Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2021. Summscreen: A dataset for abstractive screenplay summarization. *arXiv preprint arXiv:2104.07091*.

Mike D'Arcy, Alexis Ross, Erin Bransom, Bailey Kuehl, Jonathan Bragg, Tom Hope, and Doug Downey. 2023. Aries: A corpus of scientific paper edits made in response to peer reviews. *arXiv preprint arXiv:2306.12587*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*.

Vidas Daudaravicius, Rafael E Banchs, Elena Volodina, and Courtney Napoles. 2016. A report on the automatic evaluation of scientific writing shared task. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 53–62.

Johannes Daxenberger and Iryna Gurevych. 2012. A corpus-based study of edit categories in featured and non-featured wikipedia articles. In *Proceedings of COLING 2012*, pages 711–726.

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. *arXiv preprint arXiv:2309.13345*.

Wanyu Du, Vipul Raheja, Dhruv Kumar, Zae Myung Kim, Melissa Lopez, and Dongyeop Kang. 2022. Understanding iterative revision from human-written text. *arXiv preprint arXiv:2203.03802*.

Nils Dycke, Ilia Kuznetsov, and Iryna Gurevych. 2022. Nlpeer: A unified resource for the computational study of peer review. *arXiv preprint arXiv:2211.06651*.

Lester Faigley and Stephen Witte. 1981. Analyzing revision. *College composition and communication*, 32(4):400–414.

Manaal Faruqui, Ellie Pavlick, Ian Tenney, and Dipanjan Das. 2018. Wikiatomicedits: A multilingual corpus of wikipedia edits for modeling language and discourse. *arXiv preprint arXiv:1808.09422*.

Jill Fitzgerald. 1987. Research on revision in writing. *Review of educational research*, 57(4):481–506.

Linda Flower and John R Hayes. 1980. The cognition of discovery: Defining a rhetorical problem. *College composition and communication*, 31(1):21–32.

Linda Flower and John R Hayes. 1981. A cognitive process theory of writing. *College composition and communication*, 32(4):365–387.

Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science*, 315(5814):972–976.

Viola Ganter and Michael Strube. 2009. Finding hedges by chasing weasels: Hedge detection using wikipedia tags and shallow linguistic features. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 173–176.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*.

Chao Jiang, Wei Xu, and Samuel Stevens. 2022. arxivedits: Understanding the human revision process in scientific writing. *arXiv preprint arXiv:2210.15067*.

John Jones. 2008. Patterns of revision in online writing: A study of wikipedia's featured articles. *Written Communication*, 25(2):262–289.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018. A dataset of peer reviews (peerread): Collection, insights and nlp applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1647–1661.

Omid Kashefi, Tazin Afrin, Meghan Dale, Christopher Olshefski, Amanda Godley, Diane Litman, and Rebecca Hwa. 2022. Argrewrite v. 2: an annotated argumentative revisions corpus. *Language Resources and Evaluation*, pages 1–35.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Yuta Koreeda and Christopher D Manning. 2021. Contractnli: A dataset for document-level natural language inference for contracts. *arXiv preprint arXiv:2110.01799*.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. *arXiv preprint arXiv:2010.05700*.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*.

Ilia Kuznetsov, Jan Buchmann, Max Eichler, and Iryna Gurevych. 2022. Revise and resubmit: An intertextual model of text-based collaboration in peer review. *Computational Linguistics*, 48(4):949–986.

John SY Lee and Jonathan J Webster. 2012. A corpus of textual revisions in second language writing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 248–252.

Jun Liu and Sudha Ram. 2011. Who does what: Collaboration patterns in the wikipedia and their impact on article quality. *ACM Transactions on Management Information Systems (TMIS)*, 2(2):1–23.

Annie Louis and Ani Nenkova. 2013. What makes writing great? first experiments on article quality prediction in the science journalism domain. *Transactions of the Association for Computational Linguistics*, 1:341–352.

Aurélien Max and Guillaume Wisniewski. 2010. Mining naturally-occurring corrections and paraphrases from wikipedia's revision history. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Ben Medlock and Ted Briscoe. 2007. Weakly supervised learning for hedge classification in scientific literature. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 992–999.

Masato Mita, Keisuke Sakaguchi, Masato Hagiwara, Tomoya Mizumoto, Jun Suzuki, and Kentaro Inui. 2022. Towards automated document revision: Grammatical error correction, fluency edits, and beyond. *arXiv preprint arXiv:2205.11484*.

Gabriel Murray, Giuseppe Carenini, and Raymond Ng. 2012. Using the omega index for evaluating abstractive community detection. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, pages 10–18.

Eugene W Myers. 1986. An o (nd) difference algorithm and its variations. *Algorithmica*, 1(1-4):251–266.

Rani Nelken and Elif Yamangil. 2008. Mining wikipedia's article revision history for training computational linguistics algorithms.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. Scispacy: Fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*. Association for Computational Linguistics.

Sylvie Noël and Jean-Marc Robert. 2004. Empirical study on collaborative writing: What do co-authors do, use, and like? *Computer Supported Cooperative Work (CSCW)*, 13:63–89.

Samantha K Oliver, C Emi Fergus, Nicholas K Skaff, Tyler Wagner, Pang-Ning Tan, Kendra Spence Cheruvelil, and Patricia A Soranno. 2018. Strategies for effective collaborative manuscript development in interdisciplinary science teams. *Ecosphere*, 9(4):e02206.

R OpenAI. 2023. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2:13.

Ulrike Pfeil, Panayiotis Zaphiris, and Chee Siang Ang. 2006. Cultural differences in collaborative authoring of wikipedia. *Journal of Computer-Mediated Communication*, 12(1):88–113.

Dheeraj Rajagopal, Xuchao Zhang, Michael Gamon, Sujay Kumar Jauhar, Diyi Yang, and Eduard Hovy. 2022. One document, many revisions: A dataset for classification and description of edit intents. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5517–5524.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Marlene Scardamalia. 1986. Research on written composition. *Handbook of research on teaching*, pages 778–801.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *arXiv preprint arXiv:2305.14196*.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. 2022. Scrolls: Standardized comparison over long language sequences. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 12007–12021.

Alexander Spangher and Jonathan May. 2021. Newsedits: A dataset of revision histories for news articles (technical report: Data processing). *arXiv preprint arXiv:2104.09647*.

Longyue Wang, Chenyang Lyu, Tianbo Ji, Zhirui Zhang, Dian Yu, Shuming Shi, and Zhaopeng Tu. 2023. Document-level machine translation with large language models. *arXiv preprint arXiv:2304.02210*.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Elif Yamangil and Rani Nelken. 2008. Mining wikipedia revision histories for improving sentence compression. In *Proceedings of ACL-08: HLT, Short Papers*, pages 137–140.

Diyi Yang, Aaron Halfaker, Robert Kraut, and Eduard Hovy. 2017. Identifying semantic edit intentions from revisions in wikipedia. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2000–2010.

Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from wikipedia. *arXiv preprint arXiv:1008.1986*.

Fabio Massimo Zanzotto and Marco Pennacchiotti. 2010. Expanding textual entailment corpora fromwikipedia using co-training. In *Proceedings of the 2nd Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 28–36.

Fan Zhang, Homa B Hashemi, Rebecca Hwa, and Diane Litman. 2017. A corpus of annotated revisions for studying argumentative writing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1568–1578.

Fan Zhang and Diane Litman. 2015. Annotation and classification of argumentative writing revisions. In *Proceedings of the tenth workshop on innovative use of NLP for building educational applications*, pages 133–143.

Fan Zhang and Diane Litman. 2016. Using context to predict the purpose of argumentative writing revisions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1424–1430.

Xuchao Zhang, Dheeraj Rajagopal, Michael Gamon, Sujay Kumar Jauhar, and ChangTien Lu. 2019. Modeling the relationship between user comments and edits in document revision. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5002–5011.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938*.

A Constructing input in Diff format

Algorithm 1 describes the pseudo code for converting the source and target text into diff-formatted input.

B Annotation Interface

Fig. 2 displays the annotation interface created to collect our ChangeSumm corpus.

C Prompts

C.1 (a) Text Input

System prompt:

```
You are a helpful assistant
tasked with analyzing the
differences between two documents
and providing a summary of the
broad thematic changes between
the versions.  Your input is
the two documents.  The start
of documents is marked by three
newlines followed by <Document
i> where i is 1 or 2.  Given the
input in the form of (potentially
truncated) two documents, your
task is to understand the changes
made between the two versions and
cluster these changes together
based on common themes.  For
each cluster, you will provide
a brief description of the shared
theme across the changes they
contains.  Produce the output
in the following format.  Start
the description of each cluster
in a new line and put <Cluster i
Desc> at the start of the line
and </Cluster i Desc> at the end,
where i is the index of Cluster
you are describing.  Also, list
the change group numbers for
every cluster at the end of
cluster description in brackets.
Separate the description of
clusters by two new lines.
```

User prompt:

```
Here are the two documents (they
may be truncated at the end of
each document):  <Document 1>
<Document 2>
```

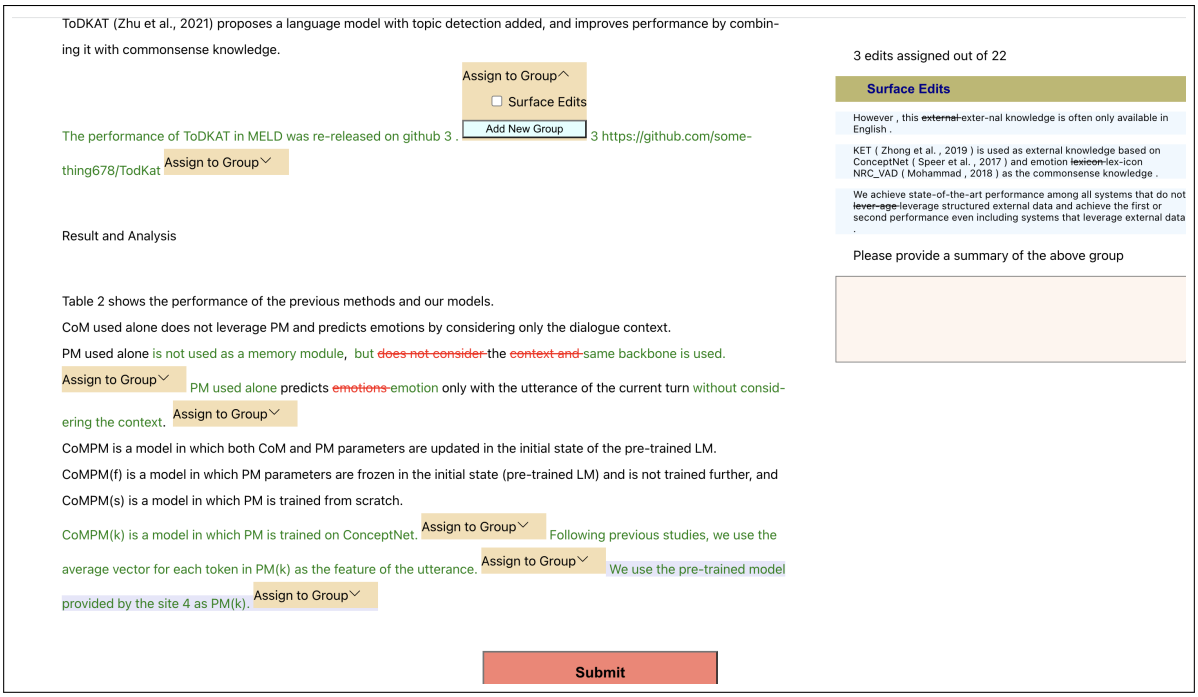

Figure 2: Annotation Interface for curating ChangeSumm Corpus.

C.2 (b) Text Input CoT

System prompt:

```
You are a helpful assistant tasked with analyzing the differences between two documents and providing a summary of the broad thematic changes between the versions. Your input is the two documents. The start of documents is marked by three newlines followed by <Document i> where i is 1 or 2. Given the input in the form of (potentially truncated) two documents, your task is to understand the changes made between the two version and cluster these changes together based on common themes. Let's think about this task in a step-by-step manner. So, first create a list of changes between the two documents assigning a number to each change. Then, cluster these changes together based on common themes. For each cluster, you will provide a brief description of the shared theme across the changes they contains and also list the change numbers (generated earlier) that are included in the cluster. Produce the output in the following format. Start the description of each cluster in a new line and put <Cluster i Desc> at the start of the line and </Cluster i Desc> at the end, where i is the index of Cluster you are describing. Also, list the change group numbers for every cluster at the end of cluster description in brackets. Separate the description of clusters by two new lines.
```

User prompt:

```
Here are the two documents (they may be truncated at the end of each document): <Document 1> <Document 2>
```

C.3 (c) Diff Input

System prompt:

```
You are a helpful assistant tasked with analyzing the differences between two documents and providing a summary of the broad thematic changes between the versions. Your input is
```

change group descriptions. Text between <Group i> and </Group i> tag describes i th change group. Note that the complete text in the group is not changed. Only the text between <add> </add> and <del> </del> are added and deleted respectively. Given the input in the form of change groups, you need to identify topic based themes in the change groups. Cluster the groups with similar topic and provide a concise summary for each cluster of change groups. Produce the output in the following format. Start the description of each cluster in a new line and put <Cluster i Desc> at the start of the line and </Cluster i Desc> at the end, where i is the index of Cluster you are describing. Also, list the change group numbers for every cluster at the end of cluster description in brackets. Separate the description of clusters by two new lines.

User prompt:

Here are the differences between two versions:

C.4 (d) Diff Input with Edit Description

System prompt:

You are a helpful assistant tasked with analyzing the differences between two documents and providing a summary of the broad thematic changes between the versions. Your input is change group descriptions. Text between <Group i> and </Group i> tag describes i th change group. Note that the complete text in the group is not changed. Only the text between <add> </add> and <del> </del> are added and deleted respectively. Given the input in the form of change groups, you need to provide a description for each change group. Describe only the exact change. Use the neighboring or whole document to derive the context of the change. Then, using the change group descriptions, you need to identify topic based themes in the change groups. Cluster the groups with similar topic and provide a concise summary for each cluster of change groups. First produce the change group description in the following format. Start the description of each group in a new line and put <Group i Desc> at the start of the line and </Group i Desc> at the end, where i is the index of group you are describing. Then, produce the cluster description in the following format. Start the description of each cluster in a new line and put <Cluster i Desc> at the start of the line and </Cluster i Desc> at the end, where i is the index of Cluster you are describing. Also, list the change group numbers for every cluster at the end of cluster description in brackets. Separate the description of clusters by two new lines.

User prompt:

Here are the differences between two versions:

C.5 (e) Diff Input with Edit Description in two separate calls and (f) with clustering based chunking

C.5.1 First Call

System prompt:

You are a helpful assistant tasked with analyzing the differences between two documents and providing a summary of the broad thematic changes between the versions. In the first stage, you will describe each change group. Describe only the exact change. Use the neighboring or whole document

to derive the context of the change. Your input is change group descriptions. Text between <Group i> and </Group i> tag describes i th change group. Note that the complete text in the group is not changed. Only the text between <add> </add> and <del> </del> are added and deleted respectively. Given the input in the form of change groups, you need to provide a description for each change group. Describe only the exact change. Use the neighboring or whole document to derive the context of the change. Produce the output in the following format. Start the description of each group in a new line and put <Group i Desc> at the start of the line and </Group i Desc> at the end, where i is the index of group you are describing.

User prompt:

Here are the differences between two versions:

C.5.2 Second Call

System prompt:

You are a helpful assistant tasked with analyzing the differences between two documents and providing a summary of the broad thematic changes between the versions. In this stage, you will cluster the change group descriptions to form topical themes.Your input is descriptions of change groups. Text between <Group i> and </Group i> tag describes the changes in i th group. Given the change group descriptions, you need to identify topic based themes in the change groups. Cluster the groups with similar topic and provide a concise summary for each cluster of change groups. Produce the output in the following format. Start the description of each cluster in a new line and put <Cluster i Desc> at the start of the line and </Cluster i Desc> at the end, where i is the index of Cluster you are describing. Also, list the change group numbers for every cluster at the end of cluster description in brackets. Separate the description of clusters by two new lines.

User prompt: Here are the descriptions of change groups:

C.5.3 Consolidation of clusters

System prompt:

You are a helpful assistant tasked with carrying out a part of the process of analyzing the differences between two documents and providing a summary of the broad thematic changes between the versions. If the input documents are too long, they will be chunked into smaller parts to fit in the LLM context. The LLM will produce clusters on the chunks that they see. Specifically, your task is to take the cluster descriptions produced by previous LLM calls on individual chuncks of documents and consolidate them if the themes covered by those clusters are similar or highly related. Note that the cluster numbers may be repeated for outputs of different calls to LLMs in previous steps. Do not merge clusters based on their cluster numbers. After consolidating the clusters, please generate description for them and also provide the mapping from the new clusters to the change groups they are representing. Keep the format of output the same as the format of input. Do not put any explanation of how you arrived at the output.

User prompt:

Here is the description of clusters:

Algorithm 1: Constructing Input in Diff Format

Input : Source text, Target Text

Output : Final Output string representing Target interspersed with source text in the diff format with edit units at the sentence level

Algorithm:
Sentence Tokenize Source and Target Text;
Initialize an empty list to add alignment pairs;
Obtain edit Segments as contiguous list of pairs from diff-match-patch between source text, target text (Fist element in pair represent the operation (deleted, inserted, equal) and second element represent substring);
Mark pointers to the start of source sentences, target sentences ;
Set Current Aligned text of source, target to Null ;
foreach *pair in edit segments* **do**
    Traverse the pointers of source sentences, target sentences based on the text string and operation in edit segment;
    Update Current Aligned source text, target text by concatenating edit segment text to it;
    if *Target Pointer reaches the end of sentence* **then**
        create a new alignment pair between existing current aligned source and target text;
        Set Current Aligned source and target text to Null;
    end
    if *Source Pointer reaches the end of sentence and Current Aligned target text is Null* **then**
        create a new alignment pair between existing current aligned source and empty target text;
        Set Current Aligned source and target text to Null;
    end
end
Initialize final output string;
foreach *aligment pair in aligment pairs* **do**
    if *source or target text is empty* **then**
        Add non-empty part to current output string marking with the corresponding edit operation;
    end
    else
        if *source and target text in aligment pair is same* **then**
            Append either of text to final output string;
        end
        else
            Detect word-level changes using difflib on both source and target text and append to final output string;
        end
    end
end