SuDocu: Summarizing Documents by Example

Anna Fariha Matteo Brucato Peter J. Haas Alexandra Meliou
College of Information and Computer Sciences
University of Massachusetts, Amherst
{afariha, matteo, phaas, ameli}@cs.umass.edu

ABSTRACT

Text document summarization refers to the task of producing a brief representation of a document for easy human consumption. Existing text summarization techniques mostly focus on generic summarization, but users often require *personalized* summarization that targets their specific preferences and needs. However, precisely expressing preferences is challenging, and current methods are often ambiguous, outside the user's control, or require costly training data. We propose a novel and effective way to express summarization intent (preferences) via *examples*: the user provides a few example summaries for a small number of documents in a collection, and the system summarizes the rest. We demonstrate SuDocu, an example-based personalized DOCUment SUmmarization system. Through a simple interface, SuDocu allows the users to provide example summaries, learns the summarization intent from the examples, and produces summaries for new documents that reflect the user's summarization intent. SuDocu further explains the captured summarization intent in the form of a *package query*, an extension of a traditional SQL query that handles complex constraints and preferences over answer sets. SuDocu combines topic modeling, semantic similarity discovery, and in-database optimization in a novel way to achieve example-driven document summarization. We demonstrate how SuDocu can detect complex summarization intents from a few example summaries and produce accurate summaries for new documents effectively and efficiently.

PVLDB Reference Format:
A. Fariha, M. Brucato, P. J. Haas, and A. Meliou. SuDocu: Summarizing Documents by Example. *PVLDB*, 13(12): 2861-2864, 2020.
DOI: https://doi.org/10.14778/3415478.3415494

1. INTRODUCTION

Document collections, such as Wikipedia, contain a wealth of information that can assist in many tasks. Yet, finding the right information quickly and easily is still a big challenge, despite all the advances in search engine technology, natural language processing, and machine learning. Consider the following scenario:

Example 1 (Trip planning)**.** *Arnob wants to plan visits to interesting places around the USA. She wants to know interesting locations and typical weather conditions for each state, but finding this information on the Web for 50 states is tedious and time-consuming. She knows that Wikipedia contains all the information she needs, but each page is large and full of facts that are not relevant to her intent (e.g., demographics, law, etc.). Arnob can manually extract relevant summaries of at most 3 pages, by selecting a small set of sentences that correspond to her specific information needs (interesting places and weather). But to thoroughly research her options, she needs an automated way to do this for the remaining 47 states.*

Proceedings of the VLDB Endowment, Vol. 13, No. 12
ISSN 2150-8097.
DOI: https://doi.org/10.14778/3415478.3415494

Surprisingly, today's technology cannot help Arnob! A search engine, like Google, is good at finding which web pages are likely to contain relevant information, but it would require many queries and Arnob would need to be very thoughtful about search keywords in order to collect the relevant information for all 50 states. Arnob tried to use Natural Language Processing (NLP) and Machine Learning (ML) techniques and found that *text summarization* tools may be helpful. However, most text summarization tools are "generic": they produce summaries that are not tailored for her personal preferences and specific information needs. The summaries she obtained from these tools did not cover all important aspects of her task, but rather provided general information about the state's politics, law, education, etc. Arnob found that some summarization tools can be tailored with a user intent, and require a natural language question to express it. She picked a question answering system, like Alexa, and issued the following question: "What are some interesting places in Massachusetts and how extreme is the weather there?" Unfortunately, the system could not understand what Arnob meant by "interesting places"—since interestingness is a very personal concept—and returned her sentences about places of general interest: MIT, Harvard Square, and Boston Library.

Arnob is interested in natural sites: parks, lakes, mountains, seas, etc. While particular preferences may be hard to express precisely with a query, it is easy for Arnob to identify relevant sentences within a document. For example, Arnob selected the following sentences from Utah's Wikipedia page as most relevant to her needs:

Example 2 (Personalized summary of Utah)**.** *The state of Utah relies heavily on income from tourists and travelers visiting the state's parks and ski resorts. Today, Utah State Parks manages* 43 *parks and several undeveloped areas totaling over* $95,000$ *acres of land and more than* $1,000,000$ *acres of water. With five national parks (Arches, Bryce Canyon, Canyonlands, Capitol Reef, and Zion), Utah has the third most national parks of any state after Alaska and California. Temperatures dropping below* $0\ ^\circ F$ *should be expected on occasion in most areas of the state most years.*

She would like to extract something similar to the summary of Example 2 for each of the 50 states. Luckily, she can now use SuDocu, a personalized DOCUment SUmmarization system, that enables users to specify their summarization intent by a few *example summaries* and produces *personalized* summaries for new documents. SuDocu is an instance of a *query-by-example* system [5], tailored for text document summarization. The key motivation of SuDocu is that asking a user to provide examples of their desired answers, rather than vague questions, is a more effective way to learn the true intent, especially for a complex summarization intent involving multiple topics, e.g., interesting places *and* weather.

We demonstrate SuDocu [1], an end-to-end system that achieves *example-driven personalized* document summarization. The key idea is to view summarization as a combinatorial optimization problem where we want to extract an optimal set of sentences to form the summary, subject to the constraint that the summary's overall topic coverage should be *close* to that of the examples. We model topics of the documents using a standard LDA approach [3], adapt our prior work for example-driven semantic similarity discovery [5] to create the constraints, and solve the resulting integer linear program using our prior techniques for scalable package queries [4].

In our first demonstration scenario, the attendees impersonate Arnob. They observe first-hand how SuDocu detects their summarization intent from only a few example summaries of a few documents, and then efficiently produces summaries of new documents matching their intents. Participants are free to specify their own intent by choosing different example summaries. We proceed to discuss how SuDocu's personalized summarization differs from prior art (Section 2), provide a solution sketch (Section 3), and conclude with a detailed outline of our demonstration (Section 4).

2. CONTRAST WITH PRIOR ART

We focus on producing a personalized *extractive* summary of each document within a collection of documents. Such a summary directly selects sentences from the document to form the summary. (In contrast, *abstractive* summarization, which synthesizes new sentences that embody a holistic understanding of the document, is a much harder task; even state-of-the-art deep learning methods struggle to produce human-readable summaries [10].) The key issues are: how to (1) express the user's intent, and (2) select the set of sentences that, collectively, best satisfy the user's intent.

In *query-based* summarization, users specify their intent in the form of an unstructured query—typically, a natural language question. For example, the question “What are some interesting places?” is very subjective, as different people consider different places as interesting. For a nature enthusiast, parks, lakes, oceans, and mountains are interesting; for an art enthusiast, museums, concerts, and plays are interesting. Some approaches use hints that represent user interest. Such hints take different forms, such as user-provided annotations [8], vision-based eye-tracking [11], user history and collaborative social influences [9], and so on. SuDocu allows the user to provide precise and concrete examples of the type of summaries they want, and does not require large training data. A possible way to adapt query-based summarization for example-driven summarization is to infer the underlying natural-language query from the example summaries, and then use an existing tool. However, computers understand structured queries with clear semantics, which can easily be constructed from examples, much better than natural language queries, so an example-based approach is both simpler and more accurate.

Early approaches to sentence selection would score each sentence based on some criteria and return the top-k sentences as a summary. This would often lead to the inclusion of redundant sentences. To tackle the issue of redundancy, later work [6] followed an ad hoc iterative greedy approach, leading to suboptimal summaries. Alternative approaches based on topic modeling identify a set of topics that the user cares about (perhaps extracted from examples) and then pick the best sentence per topic to construct the summary. However, such a summary can also be suboptimal, as sentences often cover multiple topics; a sentence that is not top-scoring in any single topic, but covers multiple topics well, might be excluded from the summary. While some approaches try to iteratively refine the summary quality [2], they are mostly based on heuristic approaches, e.g., A^* search, that still do not guarantee optimality.

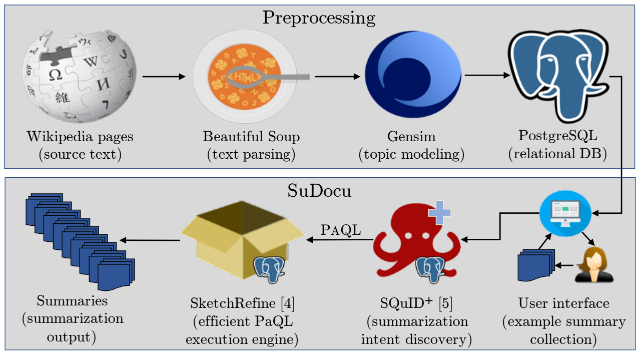

Preprocessing
Wikipedia pages (source text)
Beautiful Soup (text parsing)
Gensim (topic modeling)
PostgreSQL (relational DB)
SuDocu
Summaries (summarization output)
SketchRefine [4] (efficient PaQL execution engine)
PaQL
SQuID+ [5] (summarization intent discovery)
User interface (example summary collection)

Figure 1: The SuDocu architecture. SuDocu combines SQuID$^+$ and SketchRefine in a novel way to summarize documents by example.

A shortcoming of the foregoing sentence-selection approaches is that they consider candidate sentences in isolation, rather than trying to select a set of sentences that collectively form a good summary. The problem of selecting the best *set* of sentences can be formulated as an integer program. Lin and Bilmes [7] provide an integer programming formulation with constraints and objectives involving general sentence score, diversity, and summary length, but with no connection to the user-provided examples. In contrast, our formulation can capture the summarization intent from the example summaries using constraints on how much each topic should be “covered” by the summary; roughly speaking, the coverage should resemble that of the user-provided examples. Also, because of the combinatorially large number of possible summaries, the formulation in [7] cannot generally scale to large dataset sizes. We use our previously-developed SketchRefine algorithm [4] to scale the resulting integer linear program to very large datasets.

3. SOLUTION SKETCH

We now provide a solution sketch for SuDocu. Figure 1 depicts SuDocu's end-to-end pipeline. SuDocu pre-processes a corpus of documents by extracting all the sentences, automatically identifying all the topics, and assigning topic scores to each sentence. After preprocessing, the user can interact with SuDocu's interface and issue example summaries to specify their intent. We first describe how we model the user intent, and then discuss preprocessing, summarization intent discovery, and summary generation.

Modeling personalized extractive summaries. Following prior work on text summarization [7], we model the personalized summarization as an optimization problem. Given the example summaries, we define the optimal summary as the one that maximizes a user-defined merit score (discussed later) such that the topic-coverage of the summary is similar to that of the example summaries. In SuDocu, we construct a linear constraint on topic-coverage for each topic, allowing scalable solution methods.

We express the optimization problem as a *package query* [4]. A *package* (summary) is a collection of tuples (sentences) from a relation (document) that (a) individually satisfy base predicates (traditional SQL selection predicates), and (b) collectively satisfy global predicates (package-specific predicates). A package query

Topic	Intuitive meaning	**Top related words and their associated weight** (ordered by decreasing weight)
1	*politics*	(governor, 0.015), (election, 0.013), (vote, 0.011), (democratic, 0.011), (majority, 0.009), (presidential, 0.008)
2	*legislature*	(century, 0.012), (passed, 0.011), (legislature, 0.010), (constitution, 0.009), (created, 0.007), (law, 0.006), (political, 0.006)
3	*urbanization*	(population, 0.077), (largest, 0.052), (city, 0.029), (percent, 0.019), (metropolitan, 0.012), (capital, 0.011), (people, 0.011)
4	*economy*	(major, 0.027), (economy, 0.018), (largest, 0.013), (industry, 0.013), (billion, 0.011), (production, 0.011), (oil, 0.009)
5	*demography*	(american, 0.029), (people, 0.021), (native, 0.018), (french, 0.015), (century, 0.015), (settlers, 0.012), (tribes, 0.010)
6	*climate*	(climate, 0.017), (feet, 0.011), (temperature, 0.010), (rail, 0.010), (forests, 0.009), (summer, 0.009), (winter, 0.009)
7	*location*	(north, 0.035), (west, 0.033), (south, 0.030), (east, 0.029), (southern, 0.022), (eastern, 0.020), (region, 0.020)
8	*taxes*	(tax, 0.056), (income, 0.030), (rate, 0.029), (ranked, 0.021), (nation, 0.021), (sales, 0.017), (average, 0.015), (capita, 0.014)
9	*education*	(government, 0.039), (school, 0.029), (county, 0.025), (public, 0.025), (federal, 0.023), (schools, 0.022), (law, 0.016)
10	*general*	(national, 0.007), (major, 0.006), (popular, 0.005), (system, 0.004), (founded, 0.004), (home, 0.004), (construction, 0.004)

Figure 2: Topics of Wiki pages of 50 states (extracted using topic modeling), their intuitive meaning, and top related words with associated weights.

comprises base and global predicates that define the set of feasible packages and an objective function that defines a preference ranking among them. The *Package Query Language* (PaQL) is a simple extension to SQL that allows for the easy specification of global constraints and objectives.

Preprocessing. The first step of SuDocu involves extracting sentences from documents. In our implementation, we use `Beautiful Soup`, a library for extracting content from HTML pages. We then identify all of the topics in the extracted sentences.

We use the well-known *Latent Dirichlet Allocation (LDA)* topic model [3], in which a learned (latent) topic is represented as a set of weights assigned to the words in the vocabulary, and a sentence is viewed as a set of weights assigned to the topics. (Sentences here play the role of documents in [3].) The weight of a word (resp., topic) represents its relative importance to the topic (resp., sentence). For our implementation, we used `Gensim`, a standard NLP library that offers LDA-based topic modeling. Figure 2 shows the topics learned from the Wikipedia pages of 50 US states. In general, we can plug in any topic modeling technique into SuDocu.

The LDA topic weight of a sentence scores the relevance of a particular sentence to a particular topic. For example, the first sentence from the Example Summary 2, "The state of Utah relies heavily on income from tourists and travelers visiting the state's parks and ski resorts", would score high on "economy" and low on "education". Once sentences are encoded into the topic space, a sentence s (within document d) and its merit and topic-wise scores form a tuple of the form $\langle d, s, m\_score, s.T_1, s.T_2, \dots, s.T_m\rangle$, where m_score is the merit score of s (see below) and $s.T_j$ denotes the score of s against topic T_j. We store these tuples into a PostgreSQL database.

Summarization intent discovery. To discover the summarization intent from example summaries, we extend the example-driven semantic similarity discovery approach of SQuID [5]; we call our extension SQuID$^+$. Whereas SQuID synthesizes SQL selection queries to retrieve tuples that are similar to user-specified example tuples, SQuID$^+$ synthesizes package queries to retrieve summaries (i.e., sets of tuples) that are similar to the user-specified example summaries. SQuID would treat a single sentence as an example tuple; in contrast, SQuID$^+$ considers a *set* of sentences (summary) as an example package. Further, it aims to retrieve the summary with the highest utility (maximizing total m_score) among these similar summaries. To discover similarities among example summaries, we compute the topic-wise aggregate score for each example summary by summing the topic-wise scores of its sentences. That is, the score of example E_i against topic T_j is $E_i.T_j = \sum_{s\in E_i} s.T_j$. Now we specify the global topic-coverage predicate for T_j, given a set of examples $\{E_1, E_2, \dots\}$, as follows:

$$\texttt{SUM}(T_j)\ \texttt{BETWEEN}\ \min_i E_i.T_j\ \texttt{AND}\ \max_i E_i.T_j$$

Thus the aggregate score for each topic T_j in the summary must lie between the minimum and maximum aggregate scores in the examples; i.e., viewing each E_i as a point in topic space, the summary must lie within the bounding hyperrectangle of the examples.

One can further fine-tune the above constraint bounds: e.g., if most examples scored very high against a topic and only a few scored low, increase the minimum score threshold for that topic. In general, SuDocu can accept any package constraint derivation mechanism and is not limited to SQuID$^+$.

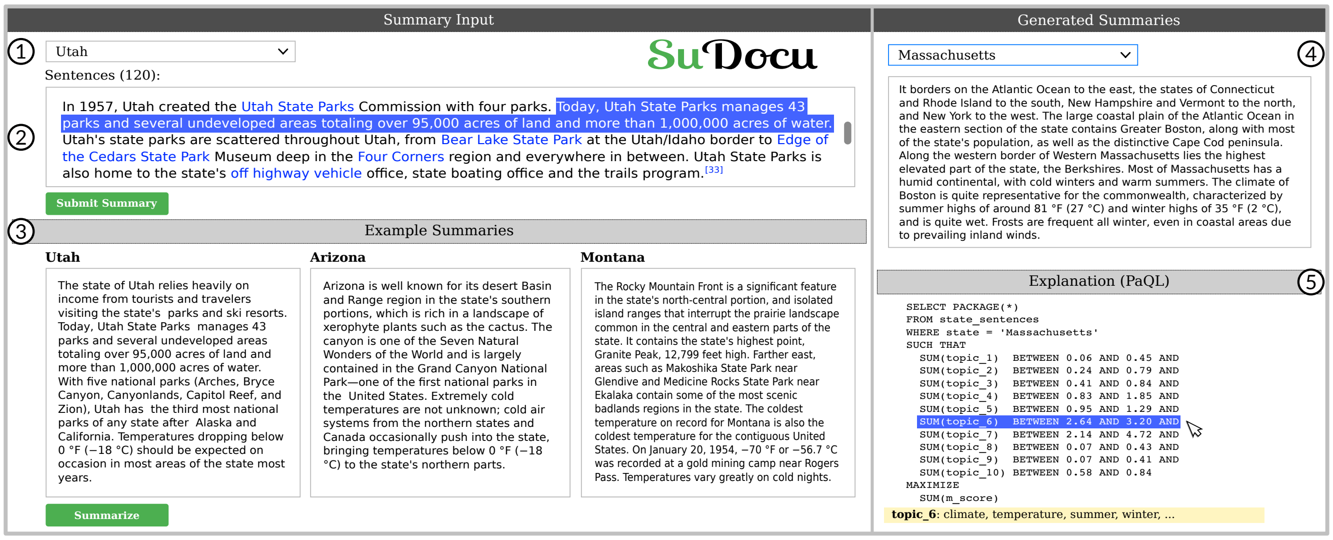
From the set of "feasible" summaries that satisfy the topic constraints, we want to select the "best" one. More precisely, we aggregate a per-sentence, user-defined "merit" score over the sentences in a summary to obtain the summary's merit score; we then seek the feasible summary having the highest merit score. Different definitions of merit are possible. If, e.g., the merit score of every sentence is -1, then maximizing the merit is equivalent to finding the shortest feasible summary. In our implementation, the merit score $m\_score(s)$ of a sentence $s = (w_1, \dots, w_J)$ comprising J words (with stop words excluded) is defined as $m\_score(s) = \sum_{j=1}^{J} F(w_j)$, where $F(w)$ is the normalized frequency of word w in the corpus. Thus the more "important" (high corpus-frequency) words that a sentence contains, the higher its merit score.

The complete PaQL query is formulated as in Figure 3. Each tuple of the input relation corresponds to a sentence, and the attributes comprise the sentence and document IDs, along with the merit and topic-wise scores. The objective function to be maximized is the summary merit score `SUM(m_score)`, and the `WHERE` clause ensures that only sentences from the document of interest are considered.

Efficient summary generation. Once the PaQL formulation of a package query is completed, the last step is to execute it. Package queries are combinatorial in nature, and solving them in general is NP-hard. If the problem is small enough, we can translate a package query directly into an equivalent integer linear program that can be solved with off-the-shelf softwares. For each tuple t_i in the input relation, the translation assigns a binary decision variable x_i corresponding to the inclusion/exclusion of t_i in the answer package. When there are so many candidate sentences that the solver either cannot load the problem in main memory or fails to find a solution, we apply the SketchRefine algorithm [4], a divide-and-conquer approach that returns a near-optimal solution of the PaQL query having a provable approximation guarantee. SuDocu then presents the optimal set of sentences as the summary to the user, along with the PaQL query that encodes the summarization intent.

4. DEMONSTRATION

We demonstrate SuDocu on the Wikipedia pages of 50 US states to show how it can accurately detect the user's summarization intent and efficiently produce effective personalized summaries. We describe the user's interaction through five steps (Figure 3), first impersonating Arnob (a nature enthusiast) and then Bruno (an economics student). We annotate each step with a circle in Figure 3.



In 1957, Utah created the Utah State Parks Commission with four parks. Today, Utah State Parks manages 43 parks and several undeveloped areas totaling over 95,000 acres of land and more than 1,000,000 acres of water. Utah's state parks are scattered throughout Utah, from Bear Lake State Park at the Utah/Idaho border to Edge of the Cedars State Park Museum deep in the Four Corners region and everywhere in between. Utah State Parks is also home to the state's off highway vehicle office, state boating office and the trails program.[33]

Utah

The state of Utah relies heavily on income from tourists and travelers visiting the state's parks and ski resorts. Today, Utah State Parks manages 43 parks and several undeveloped areas totaling over 95,000 acres of land and more than 1,000,000 acres of water. With five national parks (Arches, Bryce Canyon, Canyonlands, Capitol Reef, and Zion), Utah has the third most national parks of any state after Alaska and California. Temperatures dropping below 0 °F (−18 °C) should be expected on occasion in most areas of the state most years.

Arizona

Arizona is well known for its desert Basin and Range region in the state's southern portions, which is rich in a landscape of xerophyte plants such as the cactus. The canyon is one of the Seven Natural Wonders of the World and is largely contained in the Grand Canyon National Park—one of the first national parks in the United States. Extremely cold temperatures are not unknown; cold air systems from the northern states and Canada occasionally push into the state, bringing temperatures below 0 °F (−18 °C) to the state's northern parts.

Montana

The Rocky Mountain Front is a significant feature in the state's north-central portion, and isolated island ranges that interrupt the prairie landscape common in the central and eastern parts of the state. It contains the state's highest point, Granite Peak, 12,799 feet high. Farther east, areas such as Makoshika State Park near Glendive and Medicine Rocks State Park near Ekalaka contain some of the most scenic badlands regions in the state. The coldest temperature on record for Montana is also the coldest temperature for the contiguous United States. On January 20, 1954, −70 °F or −56.7 °C was recorded at a gold mining camp near Rogers Pass. Temperatures vary greatly on cold nights.

It borders on the Atlantic Ocean to the east, the states of Connecticut and Rhode Island to the south, New Hampshire and Vermont to the north, and New York to the west. The large coastal plain of the Atlantic Ocean in the eastern section of the state contains Greater Boston, along with most of the state's population, as well as the distinctive Cape Cod peninsula. Along the western border of Western Massachusetts lies the highest elevated part of the state, the Berkshires. Most of Massachusetts has a humid continental, with cold winters and warm summers. The climate of Boston is quite representative for the commonwealth, characterized by summer highs of around 81 °F (27 °C) and winter highs of 35 °F (2 °C), and is quite wet. Frosts are frequent all winter, even in coastal areas due to prevailing inland winds.

```
SELECT PACKAGE(*)
FROM state_sentences
WHERE state = 'Massachusetts'
SUCH THAT
  SUM(topic_1)  BETWEEN 0.06 AND 0.45 AND
  SUM(topic_2)  BETWEEN 0.24 AND 0.79 AND
  SUM(topic_3)  BETWEEN 0.41 AND 0.84 AND
  SUM(topic_4)  BETWEEN 0.83 AND 1.85 AND
  SUM(topic_5)  BETWEEN 0.95 AND 1.29 AND
  SUM(topic_6)  BETWEEN 2.64 AND 3.20 AND
  SUM(topic_7)  BETWEEN 2.14 AND 4.72 AND
  SUM(topic_8)  BETWEEN 0.07 AND 0.43 AND
  SUM(topic_9)  BETWEEN 0.07 AND 0.41 AND
  SUM(topic_10) BETWEEN 0.58 AND 0.84
MAXIMIZE
  SUM(m_score)
```

topic_6: climate, temperature, summer, winter, ...

Figure 3: The SUDOCU demo: ① the user selects a document for manual summarization, ② the user selects sentences from the document to construct an example summary, ③ the user views the example summaries, edits them if necessary, and submits them to request for summarization intent discovery, ④ the user specifies a new document to summarize and SUDOCU produces a personalized summary of it, ⑤ PaQL query that captures the summarization intent.

Impersonating Arnob. In our first demonstration scenario, the user impersonates Arnob of Example 1.

Step ① (Document selection for manual summarization): First, the user selects a state to manually summarize. Selecting a state displays all of the sentences from its Wikipedia page. In our screenshot, the user first selects Utah.

Step ② (Manual summarization): The user adds relevant sentences to the summary (by highlighting them) or removes previously selected sentences to refine the summary. Since Arnob is a nature enthusiast, the user picks sentences that mostly talk about parks, ski resorts, plants, canyons, etc. Moreover, since Arnob wants to know about the state's climate, the user also selects a few sentences about temperature.After summarizing Utah, the user repeats steps ① and ② for Arizona and Montana.

Step ③ (Summary submission): After manual summarization, the user can view the example summaries, editing them if needed. Once the user is satisfied, they request SUDOCU to discover their summarization intent. SUDOCU processes the example summaries and generates a PaQL query that encodes this intent.

Step ④ (New document summarization): Since Arnob plans to visit the east coast, the user selects Massachusetts. SUDOCU executes the PaQL query, adding `state = 'Massachusetts'` to the `WHERE` clause. SUDOCU shows the returned package as the summary of Massachusetts. Massachusetts is by the Atlantic Ocean, and has no canyons or big mountains. Although the user never provided example sentences about oceans, SUDOCU was still able to figure out that oceans would be among the most interesting places in Massachusetts based on topic similarity to canyons and parks. The summary also contains a few lines about temperature, cold winter, and warm summer, just as Arnob wants.

Step ⑤ (Summarization intent explanation): In the explanation panel, SUDOCU shows the PaQL query—the underlying mechanism to produce new summaries. The user can edit the PaQL query directly to further refine their summarization intent. The query gives the insight that the user is mostly interested in `topic_6` (climate) and `topic_7` (location). Since the topic name is not clear from the query, the user hovers on `topic_6`, revealing the most related words for that topic: climate, temperature, summer, etc.

Impersonating Bruno. Bruno wants to write a report summarizing the economy of all 50 US states. Bruno is smart. So, instead of doing all the work by himself, he decides to use SUDOCU. In our second demonstration scenario, the user impersonates Bruno. The steps are identical to the first scenario, but the summarization intent is completely different. The user now selects sentences that represent the state's economy. For example, for Utah, the user picks the sentence "The state has a highly diversified economy, with major sectors including transportation, education, . . . ". On completion of this demonstration, participants can observe how Bruno gets a completely different summarization result than Arnob.

Demonstration engagement. Besides our guided demonstration, impersonating Arnob and Bruno, participants can also issue their own summarization intent using different example summaries. Further, they can also plug their own datasets into SUDOCU.

Our demonstration showcases the ease and effectiveness of summarization by example. By formulating summarization intent as an optimization problem gleaned from a small set of user-provided examples, SUDOCU can efficiently compute concise and informative personalized summaries.

Acknowledgements: This material is based upon work supported by the NSF under grants IIS-1453543, IIS-1943971, and CCF-1763423. We thank Genglin Liu for helping us in developing the user interface of SUDOCU.

5. REFERENCES

[1] Sudocu: http://sudocu.cs.umass.edu/demo/, 2020.
[2] A. Aker, T. Cohn, and R. Gaizauskas. Multi-document summarization using A* search and discriminative training. In *EMNLP*, pages 482–491, 2010.
[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.
[4] M. Brucato, J. F. Beltran, A. Abouzied, and A. Meliou. Scalable package queries in relational database systems. *PVLDB*, 9(7):576–587, 2016.
[5] A. Fariha and A. Meliou. Example-driven query intent discovery: Abductive reasoning using semantic similarity. *PVLDB*, 12(11):1262–1275, 2019.
[6] A. Haghighi and L. Vanderwende. Exploring content models for multi-document summarization. In *NAACL*, pages 362–370, 2009.
[7] H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*, pages 912–920, 2010.
[8] R. Móro et al. Personalized text summarization based on important terms identification. In *DEXA*, pages 131–135, 2012.
[9] Z. Ren, S. Liang, E. Meij, and M. de Rijke. Personalized time-aware tweets summarization. In *SIGIR*, pages 513–522, 2013.
[10] T. Shi, Y. Keneshloo, N. Ramakrishnan, and C. K. Reddy. Neural abstractive text summarization with sequence-to-sequence models. *arXiv preprint arXiv:1812.02303*, 2018.
[11] S. Xu, H. Jiang, and F. C. M. Lau. User-oriented document summarization through vision-based eye-tracking. In *IUI*, pages 7–16, 2009.