









Extracting Workflows from Natural Language Documents: A First Step

Leslie Shing¹(✉) , Allan Wollaber¹(✉) , Satish Chikkagoudar²(✉),
Joseph Yuen^{3,4}(✉) , Paul Alvino⁴(✉) , Alexander Chambers⁴(✉) ,
and Tony Allard⁴(✉) 

¹ MIT Lincoln Laboratory, Lexington, MA, USA
{leslie.shing,allan.wollaber}@ll.mit.edu

² Naval Research Laboratory, Washington, D.C., USA
satish.chikkagoudar@nrl.navy.mil

³ Commonwealth Bank of Australia, Sydney, Australia
joseph.yuen@cba.com.au

⁴ Defence Science and Technology Group, Edinburgh, SA, Australia
{paul.alvino,alexander.chambers,tony.allard}@dst.defence.gov.au

Abstract. Business process models are used to identify control-flow relationships of tasks extracted from information system event logs. These event logs may fail to capture critical tasks executed outside of regular logging environments, but such latent tasks may be inferred from unstructured natural language texts. This paper highlights two workflow discovery pipeline components which use NLP and sequence mining techniques to extract workflow candidates from such texts. We present our Event Labeling and Sequence Analysis (ELSA) prototype which implements these components, associated approach methodologies, and performance results of our algorithm against ground truth data from the Apache Software Foundation Public Email Archive.

Keywords: Workflow discovery · Natural language · Sequence mining

1 Introduction

Visibility into organizational workflows via business process models enables the completion of critical tasks in a predictable and measurable way, and is especially relevant for organizations which must manage risk and prioritize their tasks in a contested environment. However, it is difficult to properly reconstruct workflows whose tasks are executed outside of regular logging environments. We hypothesize that latent tasks can be learned, at least in part, from unstructured natural language documents (NLDs) (e.g. emails, chats and blogs) using a technique that discovers topics and recurrent event sequences in an automated fashion. Approaches exist for workflow extraction from semi-structured NLDs [6, 14]. Our work parallels MailOfMine [4], which aims to build workflow models from unstructured NLDs (i.e. email). The objective of this paper is to highlight approaches for two of the components in the workflow discovery

pipeline, namely: (1) keyword extraction and topic clustering of events using semantic analysis of the NLDs, and (2) sequence rule mining to identify partial workflow candidates given these keywords and topic clusters.

1.1 Related Work

Email, one type of unstructured NLD (i.e. user-generated written content that lacks a predefined data model), is often a means by which informal tasks are carried out [4] and is the subject of our initial evaluation due to its availability and accessibility. Many approaches have been developed to infer activities from emails and other NLDs [5, 6, 14]. As our use case requires automated, unsupervised information extraction from unstructured text, we seek topic extraction and clustering approaches similar to [4].

Work in [6, 14] use the verbiage inherent in procedural texts to sequence tasks. Di Ciccio and Mecella describe a declarative approach for mining control-flow constraints between tasks [4]. In addition to these methods, process mining is a viable alternative approach for task sequencing. Process mining, such as Sequential Pattern Mining [9], is used to extract exemplar *cases* from information system event logs. It is widely used by organizations to identify patterns and trends in business workflows and gain other insights into workflow processes.

This paper is structured as follows. Section 2 introduces our approach for components of workflow discovery and its implementation. Section 3 provides an overview of a few experiments we conducted and their performance measures on a benchmarking dataset. Section 4 details the results attained from our experiments. Section 5 discusses the current performance of our prototype, as well as the opportunities, challenges, and lessons learned for this first iteration of our prototype, and Sect. 6 concludes the paper.

2 Methods and Approach

We have created a prototype system, Event Labeling and Sequence Analysis (ELSA), that integrates techniques from both natural language processing (NLP) and sequence process mining to automatically generate partial workflow candidates for events inferred from NLDs.

The NLP text processor component clusters NLDs based on topics inferred from related documents. We use latent semantic indexing (LSI), an unsupervised technique widely used in NLP research, and density-based spatial clustering of applications with noise (DBSCAN) [7], an algorithm that determines the number of clusters dynamically. The email subject lines and text bodies are preprocessed, transformed into a TF-IDF weighted document-term matrix for LSI, and input to DBSCAN as a k -dimensionally reduced matrix of document vectors. DBSCAN parameters are initialized with $\varepsilon = 0.2$ and $minpts = 5$ based on empirical evaluation. Each email is either labeled with a cluster ID or discarded as noise by DBSCAN, removing emails that are not representative of tasks in the overarching workflows. The top N keywords of each cluster are extracted to identify the main topics of each cluster.

The sequence process mining component uses sequence rules to extract partial workflow candidates from document clusters. Sequence rule mining identifies temporal rules $A \rightarrow B$ (A followed by B , where A and B are itemsets) from input sequences, support (i.e. the fraction of the sequences that contain the rule), and confidence (i.e. the fraction of sequences that contain the rule out of the sequences that contain the ‘trigger’ of the rule). The Top-K Non-Redundant Sequential rules (TNS) algorithm [9] is a sequence mining technique that prunes the search space to avoid redundant generation of sequential rules and determines the minimum support for a rule dynamically. This algorithm was selected due to its success in several logistical domains relevant to our research. TNS was configured to keep the top 30 rules with a minimum confidence threshold of 0.5 and a Δ value of 2, based on [9]. Input sequences for TNS are temporally ordered emails grouped by cluster ID as identified by DBSCAN. The output of the TNS algorithm is a set of the top 30 sender sequence rules and their associated support and confidence values.

ELSA’s modular design consists of the following key modules: data ingestor, NLP text processor, sequence database transformer, and sequence process miner. Each component uses generalized APIs for communication to allow for the maturation and development of modules within minimal constraints. ELSA is written primarily in Python and uses a Python wrapper for the Java implementation of the TNS algorithm [8]. All data, including the output from pipeline components are stored in a SQLite database.

3 Experimental Design

To verify ELSA’s performance, we compare output at each step of the analysis pipeline against a known ground truth, curated from open source data from the Apache Camel project [1]. The process for this is described in [2]. The ground truth consists of a total of 250 manually-evaluated emails, each tagged with five keywords and assigned to one of 65 email traces.

For each email, ELSA produces a vector of terms and associated weights, whereas the ground truth identifies five keywords. For standard set-based metrics like the Jaccard similarity (J) and Sørensen-Dice coefficient/ F_1 score, we take only the top five ELSA keywords. To employ vector-based metrics such as the generalized Jaccard (GJ), cosine (C) and soft cosine (\tilde{C}) similarities, we assume equal weighting of the ground truth keywords. Since we only require that the keywords be semantically similar for clustering, we use ‘soft’ versions of the Jaccard and Sørensen-Dice similarities (denoted with a tilde) analogous to the soft cosine,

$$\tilde{J}(A, B) \equiv \frac{\sum_{a \in A, b \in B} S(a, b)}{\sum_{a, a' \in A} S(a, a') + \sum_{b, b' \in B} S(b, b') - \sum_{a \in A, b \in B} S(a, b)}, \quad (1)$$

$$\tilde{F}_1(A, B) \equiv \frac{2 \sum_{a \in A, b \in B} S(a, b)}{\sum_{a, a' \in A} S(a, a') + \sum_{b, b' \in B} S(b, b')}, \quad (2)$$

where S is a similarity measure between pairs of keywords, chosen to be the Wu-Palmer (WP) path-similarity [15] calculated through WordNet [12]. We choose WordNet senses for the keywords to maximize the WP path-similarity between each keyword pair.

Whereas the ground truth identifies event traces, ELSA produces sender rules. To compare these, we use the concepts of support and confidence introduced in Sect. 2, additionally defining ‘soft’ versions given by

$$\text{soft support}(X \rightarrow Y) \equiv \frac{\tilde{N}(X \rightarrow Y)}{|S|}, \quad (3)$$

$$\text{soft confidence}(X \rightarrow Y) \equiv \frac{\tilde{N}(X \rightarrow Y)}{N(X)}. \quad (4)$$

Here $|S|$ is the total number of traces, $N(X)$ is the number of traces where the ‘trigger’ X of the rule is seen, and \tilde{N} is a count of partially-observed rules,

$$\tilde{N}(X \rightarrow Y) \equiv \sum_S \frac{|\Delta Y|}{|Y|} \quad \text{for the largest } \Delta Y \subset Y \text{ after } X \text{ in } S. \quad (5)$$

4 Results

The left table in Fig. 1 shows the proportion of emails with at least n matching keywords between the ground truth and ELSA’s top five. This matching is either direct (keyword-to-keyword) or soft (through the soft Jaccard index). ELSA’s performance in this area is promising, with 75% of the emails having at least one keyword directly matching, and 60% having at least three keywords softly matching. A comparison between the direct and soft matchings shows that the semantic meaning is often similar in many cases where keywords do not match exactly. The right plot in Fig. 1 shows the fraction of emails with similarity scores of at least s between the ELSA and ground truth keywords, for the metrics discussed in Sect. 3.

The average support and soft support for ELSA’s sender rules are relatively low, 0.8% and 1.2% respectively, indicating that the rules are very specific. The table and plot in Fig. 2 show the proportion of sender rules found to have a

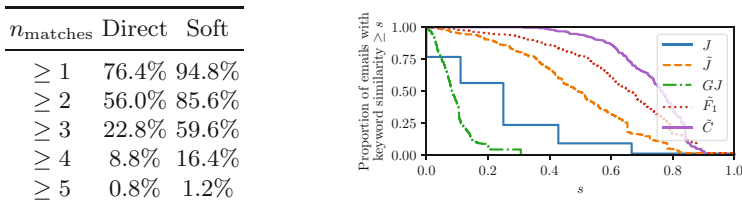


Fig. 1. Proportion of emails with (left) at least n keywords matching (directly or softly) and (right) similarity scores of at least s between ELSA and the ground truth.

Confidence	Direct	Soft
≥ 0.2	45.5%	54.5%
≥ 0.4	27.3%	40.9%
≥ 0.6	13.6%	13.6%
≥ 0.8	13.6%	13.6%
≥ 1.0	13.6%	13.6%

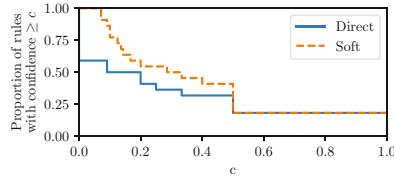


Fig. 2. Proportion of rules with a (direct or soft) confidence of at least c when measured against the ground truth traces.

(direct or soft) confidence of at least c , where we note that 40% of the rules have a soft confidence greater than 0.4. That is, we are seeing the rules (which we are considering to be candidate trace fragments) represented to some extent in the ground truth traces, indicating an important step towards trace construction.

5 Discussion

ELSA experiences several shortcomings that we will address in future iterations. Firstly, LSI uses a bag-of-words method, which does not consider word ordering or associations, and hence polysemy and synonymy are not captured. This can lead to inaccuracies in topic clustering. Additionally, LSI does not perform well on short documents. To address these issues, we aim to add Word Sense Disambiguation techniques, such as [13] which use external knowledge bases like Wordnet [12] to determine the intended word senses and enrich the documents with contextual data. Additionally, we would like to remove Subject Matter Expert (SME) input for initializing algorithm parameters.

Some components of ELSA’s pipeline are as yet unimplemented. ELSA has no additional layer of abstraction (metalabels) from the keywords, which provide additional context for the types of actions completed. We will apply techniques such as Explicit Semantic Analysis [10] and lexical graph similarity metrics to extract these metalabels. ELSA also lacks a trace assignment step between the NLP processing and sequence mining components, with input to the TNS algorithm simply being temporal sequences sorted by topic. We will use a combination of Allen’s logic [3] and straightforward ‘group-by-conversation’ rules to generate trace instances. Finally, we require a method to better extract workflow instances from sequences of events. One possibility is to use recurrent neural networks, owing to their effectiveness in learning rules from sequential data [11].

6 Conclusion

In this work, we have highlighted approaches for two components in the workflow discovery pipeline. We have used quantitative metrics to assess the extent to which our software prototype (ELSA) was able to successfully cluster emails,

extract keywords, and discover repeated patterns in a ground truth dataset. Although ELSA performed fairly well at keyword labeling and was acceptable at discovering sender-sequences in traces, it is only the first step towards discovering workflows, as discussed in Sect. 5. This approach will now drive the development for the second version of ELSA, which will use further abstractions beyond traces to discover workflows and be tested against the same ground truth dataset, as well as a system with more mission context.

Acknowledgment. This material is based upon work supported under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U. S. Air Force.

References

1. Apache camel. <https://camel.apache.org/>. last accessed 23 Jan 2018
2. Allard, T., Alvino, P., Shing, L., Wollaber, A., Yuen, J.: A novel dataset to facilitate automated workflow analysis. PLOS ONE (2018) (submitted)
3. Allen, J.F., Ferguson, G.: Actions and events in interval temporal logic. *J. Log. Comput.* **4**(5), 531–579 (1994)
4. Di Ciccio, C., Mecella, M., Scannapieco, M., Zardetto, D., Catarci, T.: MailOfMine – analyzing mail messages for mining artful collaborative processes. In: Aberer, K., Damiani, E., Dillon, T. (eds.) SIMPDA 2011. LNBIP, vol. 116, pp. 55–81. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34044-4_4
5. Dredze, M., Lau, T., Kushmerick, N.: Automatically classifying emails into activities. In: Proceedings of the 11th International Conference on Intelligent User Interfaces, pp. 70–77. ACM (2006)
6. Dufour-Lussier, V., Le Ber, F., Lieber, J., Nauer, E.: Automatic case acquisition from texts for process-oriented case-based reasoning. *Inf. Syst.* **40**, 153–167 (2014)
7. Ester, M., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd* **96**, 226–231 (1996)
8. Fournier-Viger, P., Gomariz, A., Gueniche, T., Soltani, A., Wu, C.W., Tseng, V.S.: SPMF: a Java open-source pattern mining library. *J. Mach. Learn. Res.* **15**(1), 3389–3393 (2014)
9. Fournier-Viger, P., Tseng, V.S.: TNS: mining top-k non-redundant sequential rules. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, pp. 164–166. ACM (2013)
10. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 1606–1611 (2007)
11. Lipton, Z.C., Berkowitz, J., Elkan, C.: A critical review of recurrent neural networks for sequence learning. arXiv preprint [arXiv:1506.00019](https://arxiv.org/abs/1506.00019) (2015)
12. Miller, G.A.: Wordnet: a lexical database for English. *Commun. ACM* **38**(11), 39–41 (1995)
13. Navigli, R., Lapata, M.: An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(4), 678–692 (2010)

14. Schumacher, P., Minor, M., Walter, K., Bergmann, R.: Extraction of procedural knowledge from the web: a comparison of two workflow extraction approaches. In: Proceedings of the 21st International Conference on World Wide Web, pp. 739–747. ACM (2012)
15. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, pp. 133–138 (1994). <https://doi.org/10.3115/981732.981751>