Liṅ: Unsupervised Extraction of Tasks from Textual Communication

Parth Diwanji, Hui Guo, Anup K. Kalia*, Munindar P. Singh
North Carolina State University, Raleigh, NC
*IBM Thomas J. Watson Research Center, Yorktown Heights, NY
{pdiwanj, hguo5}@ncsu.edu, anup.kalia@ibm.com, mpsingh@ncsu.edu

Abstract

Commitments and requests are a hallmark of collaborative communication, especially in organizational settings. Identifying specific tasks being committed to or requests from emails and chat messages can enable important downstream tasks, such as producing to-do lists, reminders, and calendar entries. State-of-the-art approaches for task identification rely on large annotated datasets, which are not always available, especially for domain-specific tasks. Accordingly, we propose Liṅ, an unsupervised approach of identifying tasks that leverages dependency parsing and VerbNet. Our evaluations show that Liṅ yields comparable or more accurate results than supervised models on domains with large training sets, and maintains its excellent performance on unseen domains.

1 Introduction

In organizational settings where team members interact with each other, *tasks* are constantly exchanged through communications. For example, a team leader may request a team member to accomplish a task via email. Team members may chat with each other to make commitments about how tasks are assigned within a team. Efficient team collaborations, including creating to-do lists and meetings, presume that the tasks are clear. How can natural language processing (NLP) support such practices?

We define a *task* as a verb phrase that specifies a single action to be carried out. One or more tasks could arise in a message from emails or chats, from a sender to a receiver. We consider the root verb of such a verb phrase as its *main verb*. Therefore, we assume identifying main verbs is essential to identifying tasks. Consider the following sentences indicating commitments or requests:(1) *I will* ***send*** *the QA later today* and (2) *please* ***reschedule*** *the meeting to next week.* Each sentence contains only one verb, which can easily be identified as a main verb. However, in some cases, main verbs could be difficult to identify. For example, consider the following sentences containing multiple verbs, and thus, the main verb of a task (in bold) is not obvious (other verbs are underlined): (1) ***let*** *me know what I need to do to be ready*; (2) *go ahead and* ***start*** *working on this*; (3) *before you arrange a meeting, we should* ***think*** *about a few things.*; and (4) *I would like to* ***meet*** *to discuss or appeal to Greg.*

Several contributions exist on identifying tasks from emails and chats. Bennett and Carbonell [2005] provide supervised machine learning classifiers for detecting whether a sentence includes a task. Lampert et al. [2010] provide a binary classifier to detect requests. Kalia et al. [2013] propose binary classifiers to detect the operational forms of business tasks such as commitments of different types. Wang et al. [2019] categorize commitments into three types—Request Information, Schedule Meeting, and Promise Action—and train a deep learning model to identify them. Lin et al. [2018] identify actions such as *reply-yesno*, *reply-ack*, and *investigate*, using a reparametrized long short-term memory (LSTM) network. Mukherjee et al. [2020] apply a sequence-to-sequence model to generate to-do lists based on commitments expressed in emails.

There are two important limitations to the existing contributions. First, the majority of them are limited to detecting whether a sentence contains a task (binary classification) without identifying the

specific task. Second, existing studies leverage supervised approaches to identify tasks, which may not be easily generalized to domain-specific task detection where manually annotated datasets are not available. Manual annotation of tasks can be cumbersome, especially for domain-specific datasets.

To this end, we propose Liṅ, an unsupervised approach to identify specific tasks from sentences. To extract specific tasks, Liṅ identifies the main verb present in a sentence by jointly modeling the syntactic and semantic information in it. We have evaluated Liṅ on an email dataset and a chat dataset. Liṅ achieves an F1-score of 80% for the email dataset and 89% for the chat dataset. The results show a significant improvement over the state-of-the-art supervised approaches.

2 Method

Liṅ identifies tasks by jointly modeling the syntactic and semantic information in sentences. We observe that tasks can be identified based on the combination of thematic role of the performer of a task, tenses, and other syntactic and semantic features. To extract syntactic features, we consider dependency parsing Chen and Manning [2014]. Dependency parsing provides diverse relations for verbs, which helps us define more syntactic rules for verbs appearing in different parts of the sentence. To extract semantic features, we consider VerbNet Schuler [2005], a structured lexicon focused on verbs.

2.1 Syntactic Features

We composed rules based on the following syntactic features to identify tasks from sentences.

1. Typed Dependency Relations. We adopt the typed dependency parsing for applying syntactic rules to sentences. For a given sentence, we first create a dependency parse tree and then traverse the tree from its root to other words to find a valid task. We consider dependency relations that a main verb can take on, such as ROOT and COMP. The clausal complement (COMP) relation is one of the best indicators for main verbs, which occur in 41% of sentences containing tasks in our annotated Email Dataset. For adjectives and verbs, COMP behaves as their object. When COMP is a verb, it provides detailed information about the action. For example, in *I would like to **call** you tomorrow*, the verb *call* has a COMP dependency on the verb *like*. It is evident from this example that *call* is the main verb representing a task.

2. Tense. We assume a verb tagged with a part-of-speech (POS) VB or VBG identifies a task. For example, *I will **send** you the details* represents a task since *send* is tagged VB whereas *I **sent** you the details* does not represent a task since *sent* is tagged VBD. We further filter out VBG verbs in the past tense using a rule on their AUX dependency For example, *I **was sending*** is eliminated.

3. Task Performer. We assume a task has a sender and a receiver. Senders are referred as the first person, while receivers are referred as the second person. Task performers can be identified using dependency relations, such as NSUBJ of a verb

4. Questions and Negations. We assume that questions do not indicate tasks. In questions, words like *When* and *Where* have a dependency of ADVMOD with verbs, and can be leveraged to filter out questions. Similarly, a sentence such as *Do **not call** me tomorrow* indicates a negative intent and hence, does not represent a task. The ADVMOD dependency of *not* on the verb identifies such cases.

5. Verb Association. If a verb is associated with an AGENT but is not a valid task, we skip its descendants except its COMPs. Consider the example *Before you arrange a meeting, we should **think** about a few things.* This sentence does not have a commitment or request and the verb *arrange* should not be considered a task. The semantic meaning of the sentence revolves around *think* which is an ancestor of *arrange*. When a verb is not associated with AGENT, it can often mean desire, need, or intent with a task as its descendant. Hence, we restrict skipping descendants to verbs associated with an AGENT.

6. Identifying multiple tasks. Whenever we find a valid task, we skip its descendants, since we are interested in only identifying distinct tasks. To extract multiple tasks in sentences, we focus on the CONJ dependency relation.

2.2 Semantic Features

We obtain semantic information for detecting tasks from VerbNet. VerbNet includes multiple verb classes, and each class is associated with syntactic structures and semantic information. VerbNet pro-

vides semantic information of each class of verbs using *semantic predicates* and *thematic roles*.

1. Thematic Roles. Thematic roles describe the participants involved and their relation with the action. VerbNet specifies the following thematic roles: AGENT, LOCATION, and THEME. An AGENT is an actor who carries out the event intentionally. We focus on this role in our algorithm. *2. Semantic Predicates*. Predicates indicate how the participant is involved in the event. *3. Agent Predicates*. We refer to a predicate associated to the AGENT of a verb as an *agent predicate*. There are 81 such agent predicates in VerbNet, which indicate how an AGENT is involved in an action. *4. Theme Predicates*. THEME role is the object of the action and has no control over the event. We refer to a predicate associated to the THEME role of a verb as *theme predicates*. These predicates provide useful information about the actions taking place, such as transfer of information, motion, and desire.

Here are some examples of verbs and their associated agent and theme predicates. For the verb **send**, the agent predicate is **cause**, and the theme predicate is **motion**. Similarly for the verb **work**, the agent predicate and theme predicate are **work** and **cooperate**, respectively. We examined the usage of each predicate and shortlisted 29 agent and 38 theme predicates as valid representations of a task. Note that the two lists are not mutually exclusive, since in some cases a predicate can be associated with the AGENT and in some cases it can be associated with the THEME of a verb.

To identify a valid task, a verb must be associated with an AGENT, and either one of its agent predicates is from our agent predicates list, or one of its theme predicates is from our theme predicates list. We check every class associated with that verb in VerbNet and, if any of the classes satisfy our conditions, the verb represents a task.

2.3 Task Inference

We infer tasks from sentences as follows. First, with a dependency parser, we create a parse tree representation of a sentence. Then, we start traversing from root and apply all rules mentioned for each node. If current word is an adjective, we use the COMP relation associated with the adjective to extract the verb. Then, we apply rest of the syntactic rules, such as checking the POS, tense, and task performers. If a verb satisfies these syntactic rules, we check for semantic validation using VerbNet. If the word is not a valid task but carried out by the agent, we skip all its descendants except COMPs. If the word is a valid task, we extract it and skip all its descendants including COMP. While skipping descendants from a node, we do not skip words that have a CONJ relationship with current node. In this way, we can handle multiple possible tasks in one sentence.

3 Experimental Setup

To evaluate Liṅ, we consider two different datasets, an email dataset and a chat dataset. The email dataset comprises of 1,000 emails with a total of 6,418 sentences extracted from the Enron corpus Klimt and Yang [2004]. We extracted 14,132 verb phrases from these sentences, of which 1,910 are labeled as tasks. The labeling was performed by two independent annotators, who achieved an inter-rater agreement (Cohen's Kappa) of 0.76, indicating a substantial agreement. They resolved their disagreements by discussion to produce the final dataset. The chat dataset comprises of 114 dialogues with a total of 300 sentences extracted from a task-oriented chatbot dialogue dataset Eric et al. [2017]. The same two annotators annotated the dataset with near perfect agreement (Cohen's Kappa=0.93).

We evaluate Liṅ as well as its syntax and semantics modules as separate models. For baselines, we adopt the Universal Sentence Encoder Cer et al. [2018] with an SVM classifier, pretrained BERT Devlin et al. [2019], and FastText Joulin et al. [2017] models. For BERT, we used the pretrained 12-layer uncased version and fine-tuned it for our labeled dataset. Since we want the models to find the exact verbs of tasks, a simple binary classification at sentence level would not be a good baseline. To avoid the sparsity of output classes, we separate the distinct verb phrases (VPs) from sentences using dependency parsing and then train these models for binary classification of all VPs (whether the current VP represents a task or not). To make sure that each VP contained only one main verb, we constructed VPs by including only the immediate descendants of a verb in a parse tree. As a context, we provided the whole sentence along with current VP as inputs. We split the dataset using five-fold cross

validation such that 80% was used for training and 20% for testing. The code and dataset can be found at `https://github.com/Parth27/Lin`.

4 Evaluation

We adopt accuracy, precision, recall, and F1 score as our metrics. We posit that F1 score is the most reliable metric here since both the datasets are imbalanced and include a preponderance of negatives. Table 1 shows the results of the above baselines and Liṅ, as well as two ablations of Liṅ comprising the syntactic and semantic reasoning alone.

	Email Dataset				**Chat Dataset**			
	Accuracy	*Precision*	*Recall*	*F1 score*	*Accuracy*	*Precision*	*Recall*	*F1 score*
SVM + USE	89.35%	54.40%	82.42%	65.51%	83.42%	70.09%	72.11%	71.09%
FastText	69.53%	69.95%	68.62%	69.25%	78.80%	71.66%	41.34%	52.43%
BERT	89.17%	74.82%	82.85%	78.58%	92.68%	85.32%	89.42%	87.32%
Liṅ Syntax	93.34%	74.48%	69.80%	72.06%	92.12%	87.12%	84.61%	85.85%
Liṅ Semantics	91.08%	58.62%	93.36%	72.01%	89.40%	74.07%	96.15%	83.68%
Liṅ	**95.36%**	**83.82%**	**77.29%**	**80.42%**	**94.85%**	**94.73%**	**86.53%**	**90.45%**

Table 1: Results on our datasets. Each measure is expressed as a percentage.

The results show that Liṅ outperforms the baselines on both datasets. Liṅ outperforms BERT by a small margin. The difference may not be significant. BERT, trained on the email dataset, performs relatively well on the chat dataset. One major reason could be the presence of simple sentences in chats with some similarities to those in emails. BERT may not work as well on a different domain.

Qualitative analysis. We observe a low recall for the Liṅ Syntax model. One major reason is that this model marks incorrect verbs as tasks, and no long considers their descendants, which may include the correct tasks. For example, in this sentence, *I think we can send the details tomorrow and then arrange a meeting*, the Syntax model marks *think* as task and does not identify its descendants *send* and *arrange*, which are the actual tasks. This way, the number of false negatives keeps increasing as the Syntax model keeps ignoring descendants. We observe that Liṅ Semantics achieved a higher recall trading off for a lower precision. Our syntactic rules help to provide a structure to the entire sentence. Lacking these rules, Liṅ Semantics considers verbs without considering the context in which they appear, yielding many false positives. Liṅ Semantics fails on sentences like ***Try*** *to* ***complete*** *the analysis*, where it marks *try* and *complete* both as tasks even though it represents a single task where *complete* is the main verb. Note that *try* could be a task, as in *Please* ***try*** *this and get back to me.*

We evaluated Liṅ with the trained supervised baseline models on a separate annotated chatbot dialogue dataset and observed that our unsupervised approach outperforms these baselines. Despite being trained on the Email dataset, SVM+USE and BERT perform well on the chat dataset. The main reason is that the chat dataset has very simple sentences with simpler structure. Since we adopt USE for encoding sentences and VPs, the SVM+USE model does not rely on the vocabulary of training data, and neither does pre-trained BERT.

Based on the above results, it is clear that Liṅ works well in multiple domains. Liṅ is able to perform comparably to or better than the state-of-the-art supervised baselines in both the domains we targeted.

5 Conclusion

Identifying the main verbs of tasks in sentences of email or chat can facilitate important downstream tasks. Our unsupervised approach achieves comparable or more accurate results than supervised baselines in domains with available training data. Evaluation on a chat dataset shows that our unsupervised approach can extend well to unseen domains, which can save time and effort of manual annotations.

References

Paul N. Bennett and Jaime Carbonell. Detecting action-items in e-mail. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 585–586, Salvador, Brazil, August 2005. Association for Computing Machinery.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *CoRR*, abs/1803.11175:1–7, 2018.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christoper Manning. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49, Saarbrücken, Germany, January 2017.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.

Anup K. Kalia, Hamid R. Motahari Nezhad, Claudio Bartolini, and Munindar P. Singh. Monitoring commitments in people-driven service engagements. In *Proceedings of the 10th IEEE International Conference on Services Computing (SCC)*, pages 160–167, Santa Clara, California, June 2013. IEEE Computer Society. doi: 10.1109/SCC.2013.62.

Bryan Klimt and Yiming Yang. Introducing the Enron corpus. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, pages 1–2, Mountain View, California, USA, July 2004.

Andrew Lampert, Robert Dale, and Cecile Paris. Detecting emails containing requests for action. In *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 984–992, Los Angeles, California, June 2010. Association for Computational Linguistics.

Chu-Cheng Lin, Dongyeop Kang, Michael Gamon, Madian Khabsa, Ahmed Hassan Awadallah, and Patrick Pantel. Actionable email intent modeling with reparametrized RNNs. In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 4856–4864, New Orleans, Louisiana, USA, February 2018. AAAI Press.

Sudipto Mukherjee, Subhabrata (Subho) Mukherjee, Marcello Hasegawa, Ahmed Hassan Awadallah, and Ryen W. White. Smart To-Do: Automatic generation of to-do list from emails. In *Annual Conference of the Association for Computational Linguistics (ACL)*, pages 8680–8689, July 2020.

Karin Kipper Schuler. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. PhD dissertation, University of Pennsylvania, USA, 2005. AAI3179808.

Wei Wang, Saghar Hosseini, Ahmed Hassan Awadallah, Paul Bennett, and Chris Quirk. Context-aware intent identification in email conversations. In *Proceedings of the 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1–10, Paris, France, July 2019. Association for Computing Machinery.